*e*Trust™ CA-ACF2® Security for z/OS

Administrator Guide

r8

Computer Associates®

H00053-4E

Contents

Chapter 1: Overview of eTrust CA-ACF2

Chapter 2: Controlling System Entry

Chapter 3: Maintaining Logonid Records

Chapter 4: Maintaining Scope Records

Chapter 5: Understanding SAF

Chapter 6: Maintaining Access Rules

Chapter 7: Maintaining Resource Rules

Chapter 8: Maintaining Entry Source and Source Group Records

Chapter 9: Maintaining Cross-Reference Records

Chapter 10: Maintaining Shift and Zone Records

Chapter 11: Maintaining Structured Infostorage Records

Chapter 12: Maintaining Cache Records

Chapter 13: Using the Command Propagation Facility

Chapter 14: Maintaining Global System Options Records

Chapter 15: Maintaining Profile Records

Chapter 16: Implementing DFSMS Support

Chapter 17: Maintaining Field Records

Chapter 18: Maintaining Identity Records

Chapter 19: Special eTrust CA-ACF2 Procedures

Chapter 20: JES Security Overview

Chapter 21: z/OS Unix System Services Support

Chapter 22: Controlling Access to the Hierarchical File System

Chapter 23: Using the SYSPLEX Coupling Facility

Chapter 24: LDAP Directory Services (LDS)

Chapter 25: Digital Certificate Support

Appendix A: Operator Identification Card Support

Appendix B: IBM-Supplied Resource Classes

Appendix C: Protecting Operator Commands

Appendix D: Implementing Member-Level Protection

Appendix E: National Language Support

Appendix F: RACF to eTrust CA-ACF2 Translation

Index

Chapter 1

Overview of eTrust CA-ACF2

eTrust™ CA-ACF2® Security for z/OS (eTrust CA-ACF2) provides protection against unauthorized destruction, disclosure, or modification of data and resources at your site. It operates as an extension of your operating system.

eTrust CA-ACF2 protects **all** data by default. That is, a user who is not the owner of the data can access the data only if the owner or the security administrator authorizes access using a rule.

This chapter describes:

- eTrust CA-ACF2 components
- The ACF command
- ACF subcommands common to all settings
- The ACF command using ISPF panels
- The ACF command in batch
- The ACFM CICS transaction
- The ACF IMS transaction

eTrust CA-ACF2 Components

From an administrative standpoint, eTrust CA-ACF2 is tailored to individual users, data sets, and resources through the features described in this section.

You can use eTrust CA-ACF2 TSO commands, ISPF panels, or batch utilities to dynamically update these components. In addition, you can use the ACFM CICS transaction and the ACF IMS transaction. These are described later in this chapter.

Note: You can also use the eTrust CA-ACF2 WorkStation (eTrust CA-ACF2/WS) product to maintain eTrust CA-ACF2 database records. eTrust CA-ACF2/WS simplifies enterprise-wide security management with advanced graphical administration, auditing, reporting, and monitoring facilities. eTrust CA-ACF2/WS combines a fast and easy-to-use Windows-based GUI for single-point administration of all eTrust CA-ACF2 systems with centralized monitoring and reporting of security events throughout a heterogeneous, multi-vendor environment.

Detailed information on each of the following components is provided in this guide. These components are all contained in the eTrust CA-ACF2 databases. These databases and other components are briefly described in the Other CA-ACF2 Components section later in this chapter.

Logonid Records

Logonid records define each system user in terms of general identification, status, privileges, access history, attributes related to TSO, CICS, IMS, VM, VSE, violation statistics, and so forth. See the "Maintaining Logonid Records" chapter.

Access Rules

Access rules describe the conditions (environment) for accessing particular data sets, and determine whether access is permitted or prevented for a user or group of users. See the "Maintaining Access Rules" chapter.

Resource Rules

Resource rules describe conditions (environment) for accessing particular resources. Resources include TSO accounts, TSO procedures, IMS transactions, commands, PSBs, and AGNs, CICS files, programs, transactions, transient data, temporary storage, SYSIDs, and DL/I calls, system resources protected by SAF calls, or any resource a site wants to define. See the "Maintaining Resource Rules" chapter

Entry Records

Entry records let a site define a source or groups of sources associated with system entry access. These records are obsolete, but still supported. XREF records provide much more flexibility then Entry records and should be used in place of Entry Records. For more information related to XREF records, see the "Maintaining Entry Source and Source Group Records" chapter.

Field Records

Field records let you specify access to records based on information in a field. You define the record to eTrust CA-ACF2 in a RECORD definition record and specify the test you want eTrust CA-ACF2 to perform to validate access in an EXPRESSN record. Then you create resource rules to define the validation. The $RECNAME control statement points eTrust CA-ACF2 to the RECORD definition record and the RECCHECK parameter points eTrust CA-ACF2 to check the EXPRESSN record. This is a CICS-only facility; DB2 record level protection is not supported. See the "Maintaining Field Records" chapter for more information.

Cross Reference Records

Cross-reference records let a site define groups of sources or groups of resources for eTrust CA-ACF2 validation processing. See the "Maintaining Cross-Reference Records" chapter.

Scope Records

Scope records limit the authority a specially privileged user has over logonid records, access rules, and other eTrust CA-ACF2 records. See the "Maintaining Scope Records" chapter.

Shift and Zone Records

Shift and zone records define periods of time when access is permitted or prevented. Zone records offset the user's local time from the executing CPU time. Zone records are only used during system entry validation. See the "Maintaining Shift and Zone Records" chapter.

Cache Records

Cache records let a site set options for the eTrust CA-ACF2 cache facility. See the "Maintaining Cache Records" chapter.

CPF Records

CPF records let a site set command propagation facility options and define how nodes in the CPF network can communicate. See the "Using the Command Propagation Facility" chapter.

GSO Records

GSO records specify a site's eTrust CA-ACF2 global system options. See the "Maintaining Global System Options Records" chapter.

Profile Records

Profile records contain security-related information about users and resources that can be requested by the system. See the "Maintaining Profile Records" chapter.

NET Records

NET records specify distributed database options. See the *Distributed Database Support Guide* for details.

Identity Records

Identity records contain extended user authentication information. See the chapter entitled, "Maintaining Identity Records."

Other eTrust CA-ACF2 Components

eTrust CA-ACF2 Databases

The eTrust CA-ACF2 databases contain the components previously described in the eTrust CA-ACF2 Components section. These databases are:

- Logonid database—contains logonid records for all users on the system.
- Rule database—contains all data set access rules.
- Infostorage database—contains resource rules, entry records, cross-reference records, scope records, shift and zone records, global system options (GSO) records, identity records, field records, and profile records.

eTrust CA-ACF2 Field Definition Record (ACFFDR)

The ACFFDR is made up of Assembler language macros that:

- Define and establish controls for each field of data in the logonid record. The logonid record contains the same fields for all users.
- Specify system options related to the logonid record and to the operation of eTrust CA-ACF2.

Changes to the information maintained in the ACFFDR are typically made only periodically. To make ACFFDR changes, you must reassemble and relink the ACFFDR. Once relinked, you must then make the new module available for use. This can be done using one of the following methods:

- At IPL time using the CLPA option to rebuild the link-pack area (LPA)
- Dynamically on a running system by using the F ACF2,NEWMOD(ACFFDR) command

ACF Command and Subcommands

Lets you create and maintain the major components of eTrust CA-ACF2. We also provide ISPF panels that perform the same functions as the TSO ACF command. The ACF command is also available for eTrust CA-ACF2 batch processing. A HELP subcommand is available and provides instructions on the use of commands and descriptions of various fields.

ISPF Panels

ISPF panels let you perform the following eTrust CA-ACF2 administration online:

- Add, change, delete, and list logonid records
- Compile, decompile, and delete access and resource rules and field records.
- Add, change, delete, and list structured infostorage records such as global system options (GSO) records and cross-reference records (XREF).
- Add, change, delete, and list unstructured infostorage records such as shift, entry, and xref.
- Display eTrust CA-ACF2 system processing options
- Run eTrust CA-ACF2 reports at the terminal
- Execute eTrust CA-ACF2 utility programs at the terminal.

Report Generators and Utilities

Report generators and utilities assist with security maintenance, administration, and auditing. The eTrust CA-ACF2 report generators produce reports and audit trails. Use these reports and audit trails to implement and maintain security and to monitor certain access and security violations. Most reports use data produced by eTrust CA-ACF2 and recorded with the IBM System Management Facility (SMF).

eTrust CA-ACF2 utilities provide tools for maintaining and enhancing security functions at your site. These report generators and utilities are described in the *Reports and Utilities Guide*.

You can also use Advantage™ CA-Earl® to run the eTrust CA-ACF2 reports. This approach gives you the capability of generating customized reports to accommodate local installation requirements. For details, see the *Reporting with Advantage™ CA-Earl® Guide*.

Your Password

Your password is a unique string of characters. Enter it and your logonid to prove your identity to the system. Once entered, the password is one-way encrypted so that it is not stored as it was entered. eTrust CA-ACF2, however, cannot protect passwords outside the computer; such controls are the responsibility of the user.

User Identification String (UID)

Identifies the user and places each user within an eTrust CA-ACF2 related structure. Whereas eTrust CA-ACF2 uses the logonid record to verify a user's system access and privileges, eTrust CA-ACF2 uses the UID to verify a user's access to data and resources. Furthermore, while the logonid identifies a unique user, the UID can identify a user or a group of users in eTrust CA-ACF2 rules. The logonid record contains the fields that comprise the UID; however, the actual UID does not exist in the logonid record. The UID string is dynamically built at sign-on time.

Each site defines its own UID structure using the @UID macro in the ACFFDR. In the @UID macro, you specify the logonid record fields that you want to include in the UID. Your site can select which fields are used, but you must use the same UID format for all users. We strongly recommend that you specify the logonid at the end of your UID. This format lets you mask UIDs most effectively. For details on masking the UID parameter in rules, see the "Maintaining Access Rules" chapter.

These fields can include the fields supplied with eTrust CA-ACF2 or fields defined by your site. For example, a UID of MM0244MKTPTH consists of four site-defined fields plus the user's logonid. The layout of these fields follows:

Position	Value	Description
1	M	Site (Munich)
2	M	Division (Marketing)
3-4	02	Department (02)
5-6	44	Function code (44)
7-14	MKTPTH	Logonid (MKTPTH)

After you define your UID, you can use it to specify groups of users in access and resource rules. For more details, see the chapters "Maintaining Access Rules," and "Maintaining Resource Rules."

The ACF Command

The ACF command provides subcommands to process eTrust CA-ACF2 rules and records. This section describes the basic operation of the ACF command and its subcommands.

When the TSO READY message appears at the terminal, you can issue the ACF command:

```
acf
```

When the system responds with the message ACF, you are ready to process eTrust CA-ACF2 rule sets and records. If you are in ISPF, you can issue the TSO ACF command from the command line.

Note: If the command you enter ends with a string containing a dash (-) or a plus sign (+), enclose the string in single quotes. For example, to list the access rule with the high level qualifier +MASTER+:

```
LIST '+MASTER+'
```

ACF Command Settings

After you issue the ACF command, you must establish the ACF command setting. The ACF command setting determines the particular type of eTrust CA-ACF2 record you can process. The ACF command has the following settings:

Setting	Type of eTrust CA-ACF2 Record Processed
ACF	A combination of LID and RULE settings. This setting is the default when you issue the ACF command.
CONTROL	System control records including CAC, CPF, NET, GSO, SMS, and TSO.
ENTRY	Entry records.
FIELD	EXPRESSN and RECORD records for record-level protection rules.
IDENTITY	Extended user authentication records.
LID	Logonid records.
PROFILE	User profile records.
RESOURCE	Resource rule sets.
RULE	Data set access rule sets.
SCOPE	Scope records.
SHIFT	Shift and zone records.
XREF	Source group (SGP) and resource group (RGP) cross-reference records.

When the ACF command is active, you can establish the ACF command setting by entering a SET subcommand. For example, to process logonids, you enter SET LID:

```
acf
 ACF
set lid
```

For some settings, you must be more specific about the type of eTrust CA-ACF2 rule set or record you want to process. Therefore, you must specify the setting and a three-character type code:

```
acf
 ACF
set entry(src)
```

The ENTRY setting and type code of SRC are described in the “Maintaining Entry Source and Source Group Records” chapter.

Summary of ACF Subcommands Available

For a given command setting, a subcommand processes a certain type of eTrust CA-ACF2 record or rule set. In a particular setting, some subcommands might not be valid.

The following table lists each setting and the ACF subcommands that are valid under that setting:

Setting	eTrust CA-ACF2 Component	ACF Subcommands for the Setting
ACF	Logonid records and access rules.	INSERT, CHANGE, LIST, DELETE, COMPILE, TEST, STORE, DECOMP
CONTROL	CAC, CPF, NET, GSO, SMS, and TSO records.	INSERT, CHANGE, LIST, DELETE. SHOW operates differently (See the "Maintaining Structured Infostorage Records" chapter.)
ENTRY	Entry records (SRC and SGP).	INSERT, CHANGE, LIST, DELETE
FIELD	RECORD and EXPRESSN records.	COMPILE, DECOMPILE, DELETE, LIST, RECKEY
IDENTITY	Extended user authentication records.	INSERT, CHANGE, LIST, DELETE
LID	Logonid records.	INSERT, CHANGE, LIST, DELETE, SYNCH
PROFILE	Profile records.	INSERT, CHANGE, LIST, DELETE, COMPILE, DECOMPILE, RECKEY
RESOURCE	Resource rules.	COMPILE, TEST, STORE, DECOMP, DELETE, LIST, RECKEY
RULE	Access rules.	COMPILE, TEST, STORE, DECOMP, DELETE, LIST, RECKEY
SCOPE	Scope records.	INSERT, CHANGE, LIST, DELETE
SHIFT	Shift and zone records (SFT and ZON).	INSERT, CHANGE, LIST, DELETE
XREF	SGP and RGP cross-reference records.	INSERT, CHANGE, LIST, DELETE

ACF Subcommands Common to All Settings

After you enter the ACF command and establish the appropriate ACF command setting, you can issue the various ACF subcommands.

The ACF subcommands common to all settings include:

- ACCESS
- CHKCERT
- CONNECT
- END or QUIT
- EXPORT
- GENCERT
- GENREQ
- HELP
- MLSLABEL
- MLWRITE
- REKEY
- REMOVE
- ROLLOVER
- SET
- SHOW
- SN
- SYNCH

ACF Operator Commands from TSO

From a user TSO ACF session, you can issue any ACF operator command from any ACF setting. There are two requirements necessary to perform commands from the TSO ACF session. First, a user must have the TSO OPERATOR privilege set on their logonid record. Second, the user must also have the SAF OPERCMDS class authorization. There is a SAF AUTH call whenever a console command is issued. This SAF call is made whether the command is issued from the operator console or from the TSO ACF session. See the "Protecting Operator Commands" appendix for an explanation of how to protect these operator commands. **Note:** OPERCMDS must be active.

The following sections describe the ACCESS, CHKCERT, CONNECT, END, QUIT, EXPORT, HELP, MLSLABEL, MLWRITE, REMOVE, REKEY, ROLLOVER, SET, SHOW, and SN subcommands.

ACCESS Subcommand

The ACCESS subcommand lists each rule line that matches a given input resource and users that match the UID mask on the rule line. The ACCESS subcommand simulates validation for users that match UID masks on rule lines. If a user's UID matches the UID mask on a given rule line, the user is not listed after subsequent rule lines that contain a matching UID mask. The exceptions to this rule are the rule lines that contain "environment variables" such as, PROGRAM, SHIFT, RECCECK, etc. When these parameters are found on rule lines and a user has not matched a previous rule line that does not contain environment variables, the ACCESS subcommand's output will list the user after the matching rule lines, unit a matching rule line is encountered that does not contain environment variables. This rule line is the last rule line that the user will be listed under.

Note: The user issuing the ACCESS subcommand must have access to the rule sets that are processed by the subcommand. Only those rule records that the user normally has eTrust CA-ACF2 access to can be processed by the subcommand.

The ACCESS subcommand has the following syntax:

```
Access    [Dsname(dsname)]
          [Resource(resourcename) Type(typecode) [Class(c)] [Sysid(sysid)]]
```

Parameter Descriptions

Dsname(*dsname*)
Specifies the name of a data set. Masking is not allowed. The data set name can be with or without quotes. If quotes are not used, the prefix of the command issuer is used as the high-level qualifier of the data set. A data set name can have from 1 to 22 levels of qualifiers. Each level must begin with an alphabetic character or the character @, $ or #. You can specify up to eight characters per level. The entire data set name, including periods, can contain up to 44 characters.

Resource(*resourcename*)
Specifies the name of a generalized resource or DB2 resource. It follows the rules for resource names as described in the "Maintaining Resource Rules" chapter.

Type(*typecode*)
Specifies the one- to three-character resource type of the given resource.

Class(*class*)
Specifies the one character class of the given resource. The supported classes for the ACCESS subcommand are R for generalized resources and D for DB2 resources. The default is R.

Note: Class and Sysid are required when specifying a DB2 resource.

Sysid(*sysid*)
Specifies the one- to four-character DB2 subsystem ID.

Note: Setting the ACCESS field of the OPTS record, a refresh of the OPTS record, and a F ACF2,NEWUID operator command to build the LID/UID cross-reference table are required to activate the ACCESS subcommand dynamically while eTrust CA-ACF2 is active. eTrust CA-ACF2 normally builds the LID/UID cross-reference table at eTrust CA-ACF2 initialization, if the GSO OPTS ACCESS field is on.

Note: Class and Sysid are required when specifying a DB2 resource.

For data set requests:

```
ACF
access dsname('g41.ief.stdout')
ACCESS Subcommand Results For: G41.IEF.STDOUT

Key: G41

Ruleline: IEF.STDOUT UID(*) NEXTKEY(NEXT1)

Nextkey: NEXT1
Prefix: G41

Ruleline: -.IEF.- UID(*****USER)
Lids: USER      USER1     USER2

      Access denied

Ruleline: -.IEF.- UID(*) NEXTKEY(NEXT2)

Nextkey: NEXT2
Prefix: G41

Ruleline: IEF.- UID(*****ADMIN01)
Lids: ADMIN01

      Access denied

Ruleline: IEF.- UID(*) READ(A) WRITE(A) ALLOC(A) EXEC(A)
Lids: All other logonids

Return from NEXTKEY: NEXT2
Key: NEXT1

Return from NEXTKEY: NEXT1
Key: G41

Ruleline: - UID(*****SYSPRG) READ(A) EXEC(A)
Lids: SYSPRG    SYSPRG2

ACF
```

For generalized resource requests:

```
ACF

access resource(bpx.daemon) type(fac)
ACCESS Subcommand Results For: BPX.DAEMON

Key: RFACBPX.DAEMON

Ruleline: UID(*****FTPD) SERVICE(READ) ALLOW
Lids: FTPD


Ruleline: UID(*****INETD) SERVICE(READ) ALLOW
Lids: INETD


Ruleline: UID(*****OMVS) SERVICE(READ) ALLOW
Lids: OMVS      OMVSC     OMVSKERN  OMVSU     OMVSUSER  OMVSU1

ACF
```

For DB2 resource requests:

```
ACF

access resource(dsn.test) type(tbl) class(d) sysid(prod)
ACCESS Subcommand Results For: DSN.TEST

Key: DTBLPRODDSN.TEST

Ruleline: UID(SHS**SSDJLP) UNTIL(07/18/02) SERVICE(SELECT) LOG
Lids: No logonids found

Ruleline: UID(SHS**SSD) UNTIL(07/18/02) SERVICE(UPDATE) PREVENT
Lids: No logonids found

ACF
```

CHKCERT Subcommand

The CHKCERT subcommand is used to display information about an X.509 certificate in a CERTDATA Profile record or a z/OS data set, including whether it is registered with eTrust CA-ACF2. The certificate can be in DER-encoded or base-64-encoded form (PEM). It can also be a PKCS #12 certificate, which includes the private key associated with the certificate. The CHKCERT subcommand can be issued in any mode of the ACF command. See the "Digital Certificate Support" chapter for more information on the CHKCERT subcommand.

CONNECT Subcommand

The CONNECT subcommand is used to associate certificate information with a key ring. See the "Digital Certificate Support" chapter for more information on the CONNECT subcommand.

END or QUIT Subcommand

The END or QUIT subcommand terminates the ACF command and returns you to normal processing. These subcommands have the following syntax:

```
ENd
```

Or:

```
Quit
```

Under the RULE and RESOURCE settings, the END subcommand also terminates the TEST subcommand. When you terminate the TEST subcommand, the ACF command still remains active and in its current setting.

EXPORT Subcommand

The EXPORT subcommand is used to export an X.509 digital certificate from the eTrust CA-ACF2 database and put it into a z/OS data set. The data set can be used to insert the certificate in another system, or can be downloaded to a personal computer and installed in a web browser. See the "Digital Certificate Support" chapter for more information on the EXPORT subcommand.

GENCERT Subcommand

The GENCERT subcommand is used to generate a digital certificate and insert a CERTDATA profile record into the eTrust CA-ACF2 infostorage database. The GENCERT subcommand can be issued in any mode of the ACF command. See the "Digital Certificate Support" chapter for more information on the GENCERT subcommand.

GENREQ Subcommand

The GENREQ command is used to generate a certificate request to be sent to a Certification Authority. GENREQ extracts the subject's distinguished name and public key from an existing certificate, packages it in PKCS #10 format, signs it with the certificate's private key, base-64 encodes the result, and writes it to a data set. This request is sent to a Certification Authority, who will create a new certificate with the same distinguished name and public key, but issued and signed by the Certification Authority. See the "Digital Certificate Support" chapter for more information on the GENREQ subcommand.

HELP Subcommand

At any time while the ACF command is active, you can issue the HELP subcommand, except while compiling or testing a rule. The HELP subcommand provides online descriptions of the setting in which you are processing and the ACF subcommands under that setting.

The HELP subcommand has the following syntax:

```
Help [[ACCess]
     [ACFRSrc]
     [ACFRUles]
     [CHAnge]
     [CHKcert]
     [COMPile]
     [CONnect]
     [DEComp]
     [DELete]
     [ENd]
     [EXport]
     [F]
     [FIELDS]
     [GENCert]
     [GENReq]
     [GSO]
     [Help]
     [INsert]
     [List]
     [MLS]
     [MODE]
     [REKey]
     [RECkey]
     [REMove]
     [Rollover]
     [SEt]
     [SHow]
     [SN]
     [STore]
     [SYNch]
     [TEst]
```

For example, to view a description of an ACF subcommand under the setting you are in, enter HELP and then the subcommand name. For example, type:

```
help change
```

For a description of the HELP subcommand itself, enter the following while the ACF command is active:

```
help help
```

When the ACF command is not active, enter the following from TSO READY mode to provide online information about the ACF command:

```
help acf
```

The ACF HELP subcommand follows TSO conventions.

The ACF command does not support dynamic allocation of the SYSHELP ddname. With TSO 2.3.1, you can put HELP into SYS1.PARMLIB(IKJTSOxx) and remove //SYSHELP from the TSO logon proc to have concatenated TSO HELP data sets. This does not work for eTrust CA-ACF2. You must still have a //SYSHELP ddname; otherwise, you will receive the following message:

```
IEC130I SYSHELP DD STATEMENT MISSING
```

MLSLABEL Subcommand

The MLSLABEL subcommand is used to display your current active seclabel in a multilevel security (MLS) environment. This security label is the one with which you entered the system and which endures for the duration of your session. It cannot be used to display any user's security label other than your own. The MLSLABEL subcommand has no operands and can be issued in any mode of the ACF command. For more information on the MLSLABEL subcommand, see the *Multilevel Security Planning Guide*.

MLWRITE Subcommand

When MLS is active on a system, the MLWRITE subcommand can be issued by an authorized user to override global write-down protection by setting, resetting or querying the setting of the write-down mode for the user's address space. For more information on the MLWRITE subcommand, see the *Multilevel Security Planning Guide*.

REKEY Subcommand

The REKEY subcommand is used to create a new certificate from an existing certificate with a new public/private key pair. The REKEY command is the first step of a rekey rollover process to retire the use of an existing private key. The REKEY subcommand will also copy the subject's distinguished name, key usage and subject alternate name from the existing certificate. The new certificate is self-signed and saved under the same logonid or CERTAUTH or SITECERT.

```
REKey {logonid|logonid.suffix|CERTAUTH|CERTAUTH.suffix|SITECERT|SITECERT.suffix}
       [Label(existing-certificate-label)]
       [WITHLbl(new-certificate-label)]
       [WITHSfx(new-certificate-suffix)]
       [SIZe({key-size|1024})]
       [ICSF]
       [ACtive({date-or-date-time|current-date-000000|current-date-time})]
       [Expire({date-or-date-time|current-date-000000|current-date-time})]
```

***logonid*|logonid.suffix|CERTAUTH|SITECERT**

The logonid specifies the user associated with the certificate. It may be a one to eight-character logonid, or a logonid, a dot, and a one to eight-character suffix. If label is specified, logonid, rather than logonid.suffix, must be specified, and indicates the logonid with which the label is associated.

Label(*label*)
Specifies a 1 to 32-character label of the existing certificate. The label can contain blanks and mixed-case characters. **Note:** For every one apostrophe desired in the Label value, two consecutive apostrophes must be specified. For example, the Label value, Frank's Certificate, should be specified as, Frank''s Certificate. If a single apostrophe is specified in the Label value, the value will be considered invalid.

WITHLbl(*label*)
Specifies a 1 to 32-character label that the new certificate will have. The WITHLBL value can contain blanks and mixed-case characters. It must be unique to the logonid with which the new certificate is associated. If WITHLBL is not specified, the label field of the new certificate defaults to the upper-case version of the logonid or logonid.suffix that was specified. **Note:** For every one apostrophe desired in the WITHLBL value, two consecutive apostrophes must be specified. For example, the WITHLBL value, Frank's Certificate, should be specified as, Frank''s Certificate. If a single apostrophe is specified in the WITHLBL value, the value will be considered invalid.

WITHSufx(*record suffix*)
Specifies a 1 to 8-character suffix of the new certificate. The new suffix can contain mixed-case characters but will be folded to upper case. The new suffix must be unique to the logonid with which the new certificate is associated. The suffix will be appended to the record key with a dot (.) preceding the suffix. If a suffix is not specified, the suffix will be in the format of AUTOnnn, where nnn is a number from 001 to 999.

SIZe({*key-size*|*1024*})
Specifies the size of the private encryption key to be generated, in bits.

- 512—Specifies a low-strength key
- 768—Specifies a medium-strength key
- 1024—Specifies a high-strength key

ICSF
Indicates that the generated private key will be placed in ICSF. ICSF must be active and configured for PKA operations. If it is not an error message will be displayed when attempting to insert or use the private key.

ACtive({*date-or-date-time*|*current-date-000000*|*current-date-time*})
Indicates the date and time that the certificate becomes active, for example 04/30/01-154403. If no time is specified, it defaults to 000000. If no date is specified, it defaults to the current day and time. Note: The year specified must fall within the range, 1950 - 2049. If an expire date is not also specified, the active year specified must fall within the range, 1950 -2048, since the expire date defaults to the active day and time plus one year.

Valid date formats include: YY/MM/DD, MM/DD/YY, DD/MM/YY, and DD/MM/YYYY.

Expire({*date-or-date-time* | *current-date-000000* | *current-date-time*})
Indicates the date and time that the certificate expires, for example MM/DD/YY. If no time is specified, it defaults to 000000. If no date is specified, it defaults to the active day and time plus 1 year. Note: The year specified must fall within the range, 1950 - 2049.

Valid date formats include: YY/MM/DD, MM/DD/YY, DD/MM/YY, and DD/MM/YYYY.

Examples

```
REKEY CERTAUTH.LOCALCA WITHLBL(Local CA 2004) SIZE(1024) EXPIRE(12/31/14)

REKEY CERTAUTH LABEL(Local CA) WITHLBL(Local CA 2004) EXPIRE(12/31/19)

REKEY CERTAUTH LABEL(Local CA) WITHLBL(Local CA 2004) WITHSUFX(LOCAL04)
EXPIRE(12/31/19)
```

REMOVE Subcommand

The REMOVE subcommand is used to disassociate a certificate and a key ring. See the "Digital Certificate Support" chapter for more information on the REMOVE subcommand.

ROLLOVER Subcommand

The ROLLOVER subcommand is the final step in the rekey – rollover process. ROLLOVER specifies the old certificate that is to be superceded by the new certificate. The ROLLOVER subcommand will perform the following actions:

- Delete the private key of the old certificate (as specified by the LABEL keyword), so that it can no longer be used to sign or encrypt and documents or certificates.
- Replaces the old certificate with the new certificate (as specified by the NEWLABEL keyword) in every key ring that the old certificate is connected to.
- Copies the serial number base from the old certificate to the new certificate.

When the rollover is complete, the new certificate is used as if it were the old certificate. The old certificate will still be available to verify signatures and decrypt data, but can no longer be used to sign or encrypt.

```
ROllover{logonid|logonid.suffix|CERTAUTH|CERTAUTH.suffix|SITECERT|
SITECERT.suffix}
        [Label(old-certificate-label)]
        [NEWLabel(new-certificate-label)|NEWSufx(new-certificate-suffix)]
[Force]
```

***logonid*|logonid.suffix|CERTAUTH|SITECERT**
The logonid specifies the user associated with the certificate. It may be a one to eight-character logonid, or a logonid, a dot, and a one to eight-character suffix. If label is specified, logonid, rather than logonid.suffix, must be specified, and indicates the logonid with which the label is associated.

Label(*label*)
Specifies a 1 to 32-character label of the old (source) certificate. This is the certificate that will have it's private key deleted. The label can contain blanks and mixed-case characters. **Note:** For every one apostrophe desired in the Label value, two consecutive apostrophes must be specified. For example, the Label value, Frank's Certificate, should be specified as, Frank''s Certificate. If a single apostrophe is specified in the Label value, the value will be considered invalid.

NEWLabel(*label*)
Specifies a 1 to 32-character label of the new (target) certificate. This is the certificate that will replace the old certificate in all the key rings that had the old certificate connected. The NEWLABEL value can contain blanks and mixed-case characters. It must be unique to the logonid with which the new certificate is associated. If a NEWLABEL is not specified, then NEWSUFX must be specified. **Note:** For every one apostrophe desired in the NEWLABEL value, two consecutive apostrophes must be specified. For example, the NEWLABEL value, Frank's Certificate, should be specified as, Frank''s Certificate. If a single apostrophe is specified in the NEWLABEL value, the value will be considered invalid.

NEWSufx(*record suffix*)
Specifies a 1 to 8-character suffix of the new (target) certificate. The new suffix can be used in place of the NEWLABEL field.

Force
Specifies that eTrust CA-ACF2 should bypass the following checks and perform the rollover unconditionally.

1. The values of LABEL and NEWLABEL must be different. If NEWSUFX is specified instead of NEWLABEL, the label of the new certificate must be different than the LABEL value.
2. The certificates identified by LABEL and NEWLABEL (or NEWSUFX) must both have private keys associated with them.
3. The certificate identified by NEWLABEL (or NEWSUFX) must have never been the target of a previously issued ROLLOVER subcommand and never used to sign other certificates.

When the FORCE keyword is specified, the previous three checks are not performed.

Note: The ROLLOVER subcommand has a degenerative feature where the private key of the certificate is deleted if both LABEL and NEWLABEL are the same and the FORCE keyword is also used.

Examples

```
ROLLOVER CERTAUTH.LOCALCA NEWLABEL(Local CA 2004)

ROLLOVER CERTAUTH LABEL(Local CA) NEWLABEL(Local CA 2004)
```

SET Subcommand

The SET subcommand lets you establish the setting of the ACF subcommand. It also has some additional functions, such as modifying the operation of other subcommands. The SET subcommand has the following syntax:

```
SEt or T  [Acf]
          [Control(CAc|CPf|Gso|Net|SMs|Tso)]
          [DIvision(appldiv)|MDivision(?|divison-mask)]
          [Entry(SRC|SGP)]
          [Field(EXP|REC)]
          [Force|NOForce]
          [Identity(Aut)]
          [Lid]
          [MEmber(nnnnnnn)]
          [NOError]
          [NORule[(jobname|ALL)]]
          [Profile(Appclu|DAtaset|DLfclass|Group|Keysmstr|Ptktdata|Sysmview|User)]
          [Resource(typecode)]
          [Rule]
          [SCope(SCP)]
          [SHift(SFT|ZON)]
          [SYSid(?|sysid)|MSYSid(sysidmask)]
          [TArget(null|=|?node1,...,noden)]
          [TErse|Verbose]
          [TRivia|NOTrivia]
          [Xref(Sgp|Rgp)]
```

An alias of T is provided for the SET subcommand. This alias can be used in z/OS CLISTs to avoid conflict with the CLIST SET command.

Note: You must specify one of the previous parameters with the SET or T subcommand.

The SET subcommand parameters ACF, LID, RULE, RESOURCE, ENTRY, SCOPE, SHIFT, IDENTITY, CONTROL, PROFILE, and FIELD are summarized earlier in this chapter. See the "NET Records" chapter of the *Distributed Database Support Guide* for more information about the SET CONTROL(NET) subcommand.

The parameters of the SET subcommand are described in the following section. Once set, the FORCE|NOFORCE, TARGET, TERSE|VERBOSE, and TRIVIA|NOTRIVIA parameters remain in effect until you change them or end the ACF command session. Changing the setting of the ACF command does not affect these subcommand settings.

If a SET command fails for any reason, you will not be able to issue any additional database-related commands (for example, LIST, DECOMP, INSERT, DELETE, and so forth) until a valid SET command has been issued.

Parameter Descriptions

FORCE | NOFORCE

This parameter applies to the ACF, FIELD, and RULE settings. FORCE stores a rule set or field record regardless of whether it already exists. You might select this option if you are compiling a number of rules online or in batch and do not want to issue the STORE command after you finish compiling each rule. NOFORCE prevents you from storing a rule or field record if it already exists. You might want to use this option when you are first learning eTrust CA-ACF2 so that you do not inadvertently write a rule that prevents users from getting at data they are normally allowed to access. FORCE is the default.

MEMBER(*nnnnnnn*)

The MEMBER parameter determines eTrust CA-ACF2-generated member names for partitioned data sets. Whenever you decompile a rule set or field record into a partitioned data set (PDS) and do not specify a member name, eTrust CA-ACF2 uses the $MEMBER (for access and resource rule sets) and $KEY values to determine the member name.

If there is no $MEMBER parameter, eTrust CA-ACF2 uses the $KEY value, if possible, as the member name. However that key might form an invalid member name, particularly if the key is masked. To generate a member name, eTrust CA-ACF2 takes the rightmost five digits of the MEMBER parameter value, increments the value of these digits by one, and then precedes the result by an @ symbol. For example, if the value of this parameter is 00003, the eTrust CA-ACF2-generated name to replace the first invalid member name is @00004, the replacement for the second invalid member name is @00005, and so on.

The most recently used eTrust CA-ACF2-generated member name is stored. It is incremented by one to form the next eTrust CA-ACF2-generated member name unless the MEMBER parameter is respecified in the meantime. This parameter must be specified with a number from 0 to 9999999.

NOERROR

The NOERROR parameter resets the error indication so that the maximum return code of four is returned when you end the ACF command. Use this parameter when you issue ACF commands in batch.

NORULE(null | *jobname* | ALL)

The NORULE parameter clears currently held, locally resident rules. Locally resident rules are resource or access rules that you have specified as locally resident using the GSO INFODIR, RESDIR, or RESRULE record. For details on these records, see the "Maintaining Global System Options Records" chapter. After storing access or resource rules, a user can use this parameter to make newly stored rules effective for subsequent access validation by specifying the following values:

- **null**—Specify SET NORULE() to clear rules from your address space.
- ***jobname***—Specify SET NORULE(jobname) to clear rules for a specific job.
- **ALL**—Specify SET NORULE ALL to reset the locally resident rules for all address spaces or an operator can enter the following command:

```
F ACF2,SETNORUL(ALL)
```

The person issuing the command must have the SECURITY privilege. For more information about the F ACF2,SETNORUL(ALL) operator command, see the "Console Operator Commands" appendix in the *Systems Programmer Guide*.

SYSID(?|*sysid*)|MSYSID(*sysidmask*)

The SYSID parameter is a string of characters that groups various infostorage records for a specific system. It becomes a part of the infostorage record key. You can specify the SYSID when you process structured infostorage records, such as CONTROL(GSO) or IDENTITY(AUT). Subsequent subcommands affect the records belonging to that SYSID, unless the subcommands specify a different SYSID. For example, to set up a default SYSID for the CONTROL(GSO) command setting for a particular session, specify:

```
SET CONTROL(GSO) SYSID(cpu1)
```

To change the default SYSID without changing the command setting, you can enter the SET subcommand and the SYSID parameter:

```
SET SYSID(cpu2)
```

To change or display multiple records with different SYSIDs, you can use the MSYSID parameter, which lets you specify a mask of characters. For more information about SYSIDs, see the "Maintaining Structured Infostorage Records" chapter.

TARGET(null|=|?|*node1,...,node100*)

Identifies nodes where the ACF subcommands are processed. Valid values are:

- **null**—Specify SET TARGET() to process CPF commands on the HOME node specified in the CPF OPTIONS record.
- **=** (equal sign)—Specify SET TARGET(=) to process ACF subcommands at the home node in addition to any nodemasks defined in this parameter.
- **?** (question mark)—Specify SET TARGET(?) to process CPF commands on the DFTCMD nodes defined in the CPF OPTIONS record.
- ***node1,...,node100***—Specify the names of up to 100 nodes or masks where you want the ACF subcommands processed.

  For example:

```
acf
 ACF
set target(NY,CHI,LA)
```

All ACF subcommands entered after this command that update the eTrust CA-ACF2 databases, update the databases at the New York, Chicago, and Los Angeles nodes.

TERSE|VERBOSE

TERSE causes a shortened display of eTrust CA-ACF2 records and rule sets with the LIST or DECOMP subcommands.

With SET TERSE, the LIST subcommand displays only the first line of the LID record.

```
set terse
set scope(scp)
list payscope
  ACF60062 SCOPE PAYSCOPE STORED BY PAYSDH ON 07/15/98-11:43
```

The LIST and DECOMP subcommands display only the first and last lines of rules and field records. The fields of the @HEADER macro of the ACFFDR determine the TERSE output.

With SET VERBOSE, eTrust CA-ACF2 displays the full rule set or record:

```
set verbose
list payscope
  ACF60062 SCOPE PAYSCOPE STORED BY PAYSDH ON 07/15/98-11:43
  DSN(PAY,TEST,PAYSDH-) LID(PAY-) UID(TFINPAY) INF(SSCPPAY-)
```

VERBOSE is the default.

TRIVIA|NOTRIVIA

TRIVIA permits the full display of the logonid record (that is, if VERBOSE is in effect). NOTRIVIA causes eTrust CA-ACF2 to display only certain fields of the logonid record when you issue the LIST subcommand. These fields are determined by the FLAGS=LIMIT parameter of the @CFDE macro, described in the "eTrust CA-ACF2 Field Definition Record Generation" appendix of the *Getting Started* guide.

TRIVIA is the default. These parameters apply to the ACF and LID settings only.

SHOW Subcommand—Distributed Database Support

The SHOW NETOPTS, NETNODE, and LIDMAP subcommands are used for supporting the shipping of logonids across a network. For more information about these commands, see the *Distributed Database Support Guide*.

```
SHOW  NETOpts|NETNode(nodemask)|LIDmap[(-|lidmask)]
```

SHOW Subcommand—All Other Settings

The SHOW subcommand lists information about eTrust CA-ACF2 as it is currently running on your system. You cannot issue a SHOW subcommand to display data about a remote system. The SHOW subcommand has the following syntax:

```
SHow     [ACF2|ALl]
         [ACTive]
         [APpldef]
         [CAchesrv]
         [CErtmap]
         [CLasmap]
         [CPf]
         [CRitmap]
         [DB2]
         [DDsn]
         [DElrsrc]
         [EIM]
         [ETAudit]
         [Fields(recid)]
         [LDS]
         [LINKlst]
         [LINUx]
         [MLid]
         [MLS[STATUS|ALL|LEVELS|CATEGORIES|SECLABELS(ALPHA|LOW-HI|HI-LOW]]
         [MOde]
         [MUsass]
         [NJe(nodename)]
         [OMVS[ALL|GROUPS(mmmm[-nnnn])|SUPERUSERS|USERS(mmmm[-nnnn])]
              [DUplicates]]
         [PGms|PROGrams]
         [PROXY]
         [REAlm]
         [RESident]
         [RSRctype[(null|D|R)]]
         [RSVwords]
         [SAFdef]
         [STAte]
         [STC]
         [SYSPlex]
         [SYstems]
         [TNg]
         [TSo]
         [Unixopts]
        [Zerofld s]
```

The SHOW subcommand accepts only one parameter at a time. Samples of the SHOW subcommand with various parameters are shown on the following pages.

Parameter Descriptions

ACF2 | ALL

Gives you the comprehensive result of issuing separate SHOW subcommands with the parameters ACTIVE, APPLDEF, CLASMAP, CPF, DB2, DDSN, DELRSRC, LINKLST, MLID, MLS, MUSASS, NJE, PROGRAMS, RESIDENT, RSRCTYPE, RSVWORDS, SAFDEF, STATE, SYSPLEX, SYSTEMS, TNG, TSO, and ZEROFLDS. (These are all the possible parameters, except for FIELDS and MODE.)

ACTIVE

Displays the eTrust CA-ACF2 intercepts that have received control (denoted by YES). Also displayed are any local exits specified in the GSO EXITS record. This record is described in the "Maintaining Global System Options Records" chapter.

```
-- ACF2 INTERCEPTS THAT HAVE RECEIVED CONTROL --
DASD-OPEN(YES)          DASD-EOV(NO)            VSAM-OPEN(YES)
TAPE-OPEN(NO)           TAPE-EOV(NO)            CATALOG(NO)
USER CALL(NO)           EXTERNAL CALL(NO)       PROGRAM CALL(YES)
JOB/STEP TERM(YES)      TSO-MVS(YES)            CAT-CVOL(NO)
NONVSAM-ERASE(YES)      VSAM-ERASE(YES)         VTAM-OPEN(YES)

-- LOCAL EXITS SPECIFIED ON THIS SYSTEM --
DSN PRE-VALIDATE=NONE                   DSN POST-VALIDATE=NONE
DSN VIOLATION=NONE                      PSEUDO DSN GENERATE=NONE
RSRC PRE-VALIDATE=NONE                  RSRC POST-VALIDATE=NONE
STC VALIDATE=NONE                       SOURCE MODIFICATION=NONE
LOGON PRE-VALIDATE=NONE                 LOGON POST-VALIDATE=NONE
PASSWORD EXPIRATION=NONE                NEW PSWD VALIDATE=NONE
RULE PRE-PROCESS=NONE                   RULE POST-PROCESS=NONE
INFO PRE-PROCESS=NONE                   INFO POST-PROCESS=NONE
SVC INITIALIZATION=NONE                 TSO LOGON TERM TYPE=NONE
TSO LOGON PARM=NONE                     DDB LID NODE LOC=NONE
DDB USER INFO MOD=NONE                  LID PRE-PROCESS=NONE
LID POST-PROCESS=NONE                   SEV PRE-PROCESS=VACFSEVP (INACTIVE)
SEV POST-PROCESS=NONE                   PROGRAM OVERRIDE=NONE
CPF EXIT=NONE

-- ACF2 DDB FACILITY --
DDB = INACTIVE

-- ACF2 CACHE FACILITY --
DATABASE CACHE = INACTIVE
CACHE SYNCHRONIZER = INACTIVE

-- ACF2 CPF FACILITY --
CPF STATUS = INACTIVE

-- ACF2 SYSPLEX FACILITY --
XES STATUS = INACTIVE
XCF STATUS = INACTIVE
```

```
-- Multilevel Security (MLS) Facility --

    MLS Status:    ACTIVE
    MLS Mode:      QUIET
    WRITE-DOWN:    ALLOWED
    Current SYSID: XE41
    Seclabel by system ID: INACTIVE
    UNIX Files/Directories: SECLABELS NOT REQUIRED
    UNIX IPC Objects: SECLABELS NOT REQUIRED

-- AUTHENTICATION EXITS ON THIS SYSTEM: LIDFLD / PROCESS PROGRAM / INFOSTG --
NONE
```

APPLDEF

Displays all active site-defined structured infostorage applications.

```
show appldef
-- INSTALLATION DEFINED STRUCTURE INFO-STORAGE APPLICATIONS --

CLASS (SHORT / LONG): I / IDENTITY
TYPE (SHORT / LONG): AUT / AUTHSUP
SELECTION AUTHORIZATION: SECURITY, ACCOUNT, AUDIT, CONSULT, LEADER
DEFAULT DIVISION ROUTINE: ACF00DFT
ACTIVE DIVISION: AUTHSUP3
ASSOCIATED RSB / RECORD ID: ACF2RSB1 / ********

CLASS (SHORT / LONG): I / IDENTITY
TYPE (SHORT / LONG): AUT / AUTHSUP
SELECTION AUTHORIZATION: SECURITY, ACCOUNT
DEFAULT DIVISION ROUTINE: ACF00DFT
ACTIVE DIVISION: AUTHSUP5
ASSOCIATED RSB / RECORD ID: ACF2RSB2 / ********

CLASS (SHORT / LONG): C / CONTROL
TYPE (SHORT / LONG): SMS / SMS
SELECTION AUTHORIZATION: SECURITY, ACCOUNT
ASSOCIATED RSB / RECORD ID: ACFDRSMS / ********
```

CACHESRV

Displays the R_cacheserv definitions currently in use. **Note:** You must specify CACHESRV to see R_cacheserv information. To see eTrust CA-ACF2 cache information only, use the SHOW ACTIVE command.

```
show cachesrv

-- GSO CACHESRV DEFINITIONS for R_cacheserv –

R_cacheserv hardening is ACTIVE
```

The R_cacheserv file name is SSDRCM.RCACHE.

```
Cache Names Eligible for Hardening

---------------------------------

RCAC01

RCAC08

RCAC10

RSYS01

RSYS02
```

CERTMAP

Displays information contained in CERTMAP records as laid out in the internal CERTMAP table. The display first shows data from records that contain both IDNFILTER and SDNFILTER, followed by records with only SDNFILTER, and finally records with only IDNFILTER.

```
sho certmap

 -- CERTMAP FILTERING TABLES --

 IDN/SDN FILTERS
 ---------------
                                                IDN FILTER
                                                SDN FILTER
 Label                            TRUST   USER  CRITERIA
 ================================ ===== ======== ============================
 ACF2 DEVELOPMENT                   Y   ACF2DEVL CN=CAI CERT AUTHORITY.OU=ACF2 D
                                                 EVELOPMENT.O=COMPUTER ASSOCIATE
                                                 S.L=LISLE.ST=ILLINOIS.C=US

                                                 OU=ACF2.OU=DEVELOPMENT.OU=COMPU
                                                 TER ASSOCIATES.L=LISLE.ST=ILLIN
                                                 OIS.C=US


 JASMINE II DEVELOPMENT             Y   JASMINE  CN=CAI CERT AUTHORITY.OU=ACF2 D
                                                 EVELOPMENT.O=COMPUTER ASSOCIATE
                                                 S.L=LISLE.ST=ILLINOIS.C=US

                                                 OU=JASMINE II.OU=DEVELOPMENT.OU
                                                 =COMPUTER ASSOCIATES.L=ISLANDIA
                                                 .ST=NEW YORK.C=US


  UNICENTER TNG DEVELOPMENT          Y   TNG      CN=CAI CERT AUTHORITY.OU=ACF2 D
                                                 EVELOPMENT.O=COMPUTER ASSOCIATE
                                                 S.L=LISLE.ST=ILLINOIS.C=US

                                                 OU=UNICENTER TNG.OU=DEVELOPMENT
                                                 .OU=COMPUTER ASSOCIATES.L=ISLAN
                                                 DIA.ST=NEW YORK.C=US


 TOP SECRET DEVELOPMENT             Y   TSSDEVL  CN=CAI CERT AUTHORITY.OU=ACF2 D
                                                 EVELOPMENT.O=COMPUTER ASSOCIATE
                                                 S.L=LISLE.ST=ILLINOIS.C=US

                                                 OU=TOP SECRET.OU=DEVELOPMENT.OU
                                                 =COMPUTER ASSOCIATES.L=PRINCETO
                                                 N.ST=NEW JERSEY.C=US
```

```
SDN FILTERS
-----------

                                                          SDN FILTER
Label                              TRUST   USER           CRITERIA
================================== ===== ======== ==============================
ISLANDIA DEVELOPMENT                 Y   DEVL     OU=DEVELOPMENT.OU=COMPUTER ASSO
                                                  CIATES.L=ISLANDIA.ST=NEW YORK.C
                                                  =US


LISLE DEVELOPMENT                    N   DEVL     OU=DEVELOPMENT.OU=COMPUTER ASSO
                                                  CIATES.L=LISLE.ST=ILLINOIS.C=US


DALLAS DEVELOPMENT                   Y   DEVL     OU=DEVELOPMENT.OU=COMPUTER ASSO
                                                  CIATES.L=DALLAS.ST=TEXAS.C=US


IDN FILTERS
-----------

                                                          IDN FILTER
Label                              TRUST   USER           CRITERIA
================================== ===== ======== ==============================
PRIVATE USERS                        Y             OU=CLASS 2 PUBLIC PRIMARY CERTI
                                                  FICATION AUTHORITY.O=VERISIGN,
                                                  INC.C=US

                                                  APPLID=&APPLID.COMPANY=&COMPANY

PUBLIC USERS                         Y             OU=CLASS 1 PUBLIC PRIMARY CERTI
                                                  FICATION AUTHORITY.O=VERISIGN,
                                                  INC.C=US

                                                  APPLID=&APPLID
```

CLASMAP

Displays the internal (eTrust CA-ACF2-defined) and external (site-defined) CLASMAP records.

```
show clasmap

-- MERGED CLASMAP DEFINITIONS --

 MUSASS     RESOURCE    TYPE    ENTITY
   ID         CLASS     CODE    LENGTH     PROFINT     LOG  MIXED   EXTERNAL
 =======    ========    ===    ======     =======     ===  =====   ========
 ********   AC#CMD      SAF      8
 ********   ACAPPL      ACA      39
 ********   ACCBPROC    ACC      39
 ********   ACCTNUM     SAF      39
 ********   ACDIALOG    ACD      39
 ********   ACICSPCT    SAF      13
 ********   ACLIST      ACL      39
 ********   ACMSG       ACM      39
 ********   ACPANEL     ACP      39
 ********   ACREPORT    ACR      39
 ********   ACSQL       ACS      39
 ********   AIMS        SAF      8
 ********   AMARY       MAR      39                  LOG  MIX     EXT
 ********   APPCLU      ALU      35       PROF
 ********   APPCPORT    SAF      8
 ********   APPCSERV    SAF      73
 ********   APPCSI      SAF      26
 ***
```

CPF

Displays information about the OPTIONS record and the CPF network as defined in NODEDEF records. Here is the SHOW CPF display issued from the NYC node:

```
sho cpf

 -- COMMAND PROPAGATION FACILITY --

 CURRENT STATUS: INACTIVE

 CURRENT SYSID: NYC          JOURNALLING: YES
 CURRENT HOME NODE: NYC      LOGDAYS: 30
 PASSWORD SYNC: YES          COMMAND: YES
 EXTENDED CPF                CMDWAIT: YES
 UNDEFINED NODES: NO         JRNL QUICK START: NO

 DFTCMD: CHI LA NYC TNG1
 DFTPSW: CHI LA NYC TNG1
 JRNLRECV: NYC.JRNLRECV.CPFJFILE
 JRNLSEND: NYC.JRNLSEND.CPFJFILE

 -- NODE DEFINITIONS --
   NODE      RECEIVE    SEND    GATEWAY    UNICENTER    VM
   NAME        FROM      TO      NODE         NODE       NODE
   ----      --------   ----    -------    ---------    ----
   CHI         YES       YES      NO          NO          NO
   LA          NO        YES      NO          NO          NO
   NYC         YES       YES      NO          NO          NO
   TNG1        YES       YES      NO          YES         NO
   VMSYS1      YES       YES      NO          NO          YES
```

CRITMAP

Displays information contained in CRITMAP records as laid out in the internal CRITMAP table. The display shows the record id, SYSID, APPLID, USERID, and associated application variables.

```
sho critmap

 -- CRITERIA TABLE --

 Record key         SYSTEMID  APPLID   USERID   APPLICATION VARS
 ================== ======== ======== ======== ================================
 CRITMAP.PUBLIC2    *        CICSAPPL PUBLIC2

 CRITMAP.PLATINUM   *        HRAPPL   PLATUSER COMPANY=PLATINUM

 CRITMAP.UCCEL      *        HRAPPL   UCCUSER  COMPANY=UCCEL

 CRITMAP.PUBLIC1    *        WEBAPPL  PUBLIC1
```

DB2

Displays information about each DB2 subsystem protected by eTrust CA-ACF2 for DB2 that has been started since you started eTrust CA-ACF2. This information includes which exits are in use, the mode specified for each type of resource, the group ID specified, and any SAFELIST records that are active. See the eTrust CA-ACF2 for DB2 Administrator Guide for more information about these resource types. If you are not using eTrust CA-ACF2 for DB2 or no DB2 subsystems are running, no information is displayed.

```
show db2

eTrust CA-ACF2/DB2 RELEASE - 1.2 SP00

-- DB2 SUBSYSTEMS EXITS AND MODES –
DSN8 EXITS  -  DB2PRE()             DB2POST()
DSN8 MODES  -  BPL = ABORT          DBS = ABORT
               FNC = ABORT          JAR = ABORT
               PLN = ABORT          PRC = ABORT
               SCH = ABORT          SEQ = ABORT
               STG = ABORT          SYS = ABORT
               TBL = ABORT          TSP = ABORT
               TYP = ABORT

GROUP ID =
------- DB2 SAFELIST RECORDS -------
CLASS     SERVICE  COLUMN              RESOURCE
========  =======  ================  ========================
DB2PLAN   EXECUTE                     DSNESPRR
```

DDSN

Lists the data set names in use for the Rule, Logonid, and Infostorage databases. Also listed are the backup data sets of these databases, if allocated. If a dynamic data set name (DDSN) list was specified or defaulted at eTrust CA-ACF2 startup, any data set names preallocated but different from the name in the DDSN list are flagged with an asterisk and a note.

```
show ddsn
-- ACF2 DYNAMIC DATA SET NAMES SPECIFIED --
DDSNS PRIMARY DEFAULTED AT STARTUP. DSNS IN USE ARE:
         RULES= ACFSYS.ALTRULES
         LOGONIDS= ACFSYS.ALTLIDS
         INFOSTG= ACFSYS.ALTINFO
         BACKRULE= NOA NOT ALLOCATED
         BACKLID= PRE ACFSYS.BKLIDS
         BACKINFO= NOA NOT ALLOCATED

DDSN LISTS DEFINED IN FDR ARE:
PRIMARY  RULES= ACFSYS.ALTRULES
         LOGONIDS= ACFSYS.ALTLIDS
         INFOSTG= ACFSYS.ALTINFO
         BACKRULE= SYS1.ACF.BKRULES
         BACKLID= SYS1.ACF.BKLIDS
         BACKINFO= SYS1.ACF.BKINFO

ALT      RULES= SYS1.ACF.ALTRULES
         LOGONIDS= SYS1.ACF.ALTLIDS
         INFOSTG= SYS1.ACF.ALTINFO
         BACKRULE= SYS1.ACF.ABKRULES
         BACKLID= SYS1.ACF.ABKLIDS
         BACKINFO= SYS1.ACF.ABKINFO
```

The middle column in the section where "DSNS IN USE ARE:" is specified indicates where the data set is defined. Valid values are:

- **PRE**—preallocation by site.
- **NOA**—not allocated or defined before eTrust CA-ACF2 startup.

DELRSRC

Displays the eTrust CA-ACF2 delegated resources currently active in the system. Delegated resources are defined in the GSO DELRSRC records. The display shows the kind of resource (DB2 or generalized resource), resource type, SYSID (applicable only for DB2 resources), and the resource name.

```
show delrsrc
-- DELEGATED RESOURCES -

  Type   Sysid  Resource
  -----  -----  ------------------------------------------------------------
  D-TBL  BBMS   USER99.RACROUTE-
  R-FAC         USER99.RACROUTE.TESTDATA
```

EIM

Displays the current default PROXY settings used by Policy Director Authorization Services, and default EIM settings used by Enterprise Identity Mapping.

In the following example, no PROXY default had been set. The EIM default did not have all fields defined.

```
ACF
SHO EIM
--DEFAULT PROXY INFORMATION SUMMARY
--DEFAULTS DO NOT EXIST
--DEFAULT EIM INFORMATION SUMMARY

BIND DISTINGUISHED NAME:     cn=eim administrator, o=CA, st=Illinois, c=US
DISTINUGUISHED DOMAIN NAME:  NONE
LDAP SERVER URL AND PORT:    NONE
LOCAL REGISTRY:              NONE
OPTIONS:                     ENABLE
```

ETAUDIT

Displays the control options that are in effect for communicating security events to eTrust AUDIT.

If the GSO OPTS record field ETAUDIT is specified, the following output is displayed:

```
SHOW ETAUDIT

-- eTrust Audit Control Options --
Start of ACF2  (S ACF2):                 NO
Stop of ACF2   (P ACF2):                 NO
Modify of ACF2 (F ACF2):                 NO
Signons:                                 NO
Signoffs:                                NO
Signon violations:                       NO
Security label violations:               NO
Data set violations:                     NO
Data set loggings:                       NO
Resource violations:                     NO
Resource loggings:                       NO
Logonid administration:                  NO
```

```
Logonid administration violations:        NO
Rule administration:                      NO
Rule administration violations:           NO
Resource rule administration:             NO
Resource rule administration violations:  NO
Infostg record administration:            NO
Infostg record administration violations: NO
USS ck_access:                            NO
USS initUSP:                              NO
USS deleteUSP:                            NO
USS initACEE:                             NO
USS R_audit:                              NO
USS R_chaudit:                            NO
USS R_chmod:                              NO
USS R_chown:                              NO
USS R_setfacl:                            NO
USS R_setgid:                             NO
USS R_setegid:                            NO
USS R_setuid:                             NO
```

FIELDS

Displays all logonid field names that you have the authority to view or modify. If your logonid has any special privileges (SECURITY, ACCOUNT, AUDIT, LEADER, or CONSULT), this display includes the fields you can modify in the logonid records of other users. An asterisk precedes the fields that you can modify in your own logonid record or in other logonid records.

```
show fields

 -- IDENTIFICATION --
  COMPANY    IDNUM      LEVEL      LID       *NAME      *PASSWORD  *PHONE
  PROJECT    SITE       UID
 -- CANCEL/SUSPEND --
 *CANCEL     CSDATE     CSWHO     *MON-LOG   *MONITOR   *SUSPEND   *TRACE
 *TSO-TRC
 -- PRIVILEGES --
 *ACCOUNT   *ACTIVE    *AUDIT     *AUTOALL   *AUTODUMP  *AUTONOPW  *AUTOONLY
 *BDT       *CICS      *CONSULT   *DG84DIR   *DIALBYP   *DUMPAUTH  *EXPIRE
 *IMS       *JOB       *JOBFROM   *GRPLOGON  *IDMS      *LDEV      *LEADER
 *LOGSHIFT  *MAINT     *MUSASS    *NO-INH    *NO-SMC    *NO-STORE  *NON-CNCL
 *NOSPOOL   *PGM       *PPGM      *PROGRAM   *READALL   *REFRESH   *RESTRICT
 *RULEVLD   *SCPLIST   *SECURITY  *SRF       *STC       *SUBAUTH   *SYNCNODE
 *SYNERR    *TAPE-BLP  *TAPE-LBL  *TSO       *USER      *VLDVMACT  *VM
 *VMBATCH   *VMBATMON  *VMSAF     *VMXA      *VSESRF
 -- ACCESS --
  ACC-CNT    ACC-DATE   ACC-SRCE   ACC-TIME
 -- PASSWORD --
 *KERB-VIO   KERBCURV   LIDTEMP    *LIDZMAX    *LIDZMIN   *MAXDAYS
 *MINDAYS    *PSWD-DAT  *PSWD-EXP  *PSWD-INV   PSWD-MIX   PSWD-SRC
 PSWD-TIM    PSWD-TOD   *PSWD-VIO  *PSWD-XTR
 -- TSO --
 *ACCTPRIV  *ALLCMDS   *ATTR2     *CHAR      *CMD-LONG  *DFT-DEST  *DFT-PFX
 *DFT-SOUT  *DFT-SUBC  *DFT-SUBH  *DFT-SUBM  *INTERCOM  *JCL       *LGN-ACCT
 *LGN-DEST  *LGN-MSG   *LGN-PERF  *LGN-PROC  *LGN-RCVR  *LGN-SIZE  *LGN-TIME
 *LGN-UNIT  *LINE      *MAIL      *MODE      *MOUNT     *MSGID     *NOTICES
 *OPERATOR  *PAUSE     *PMT-ACCT  *PMT-PROC  *PROMPT    *RECOVER   *TSOACCT
 *TSOCMDS   *TSOFSCRN  *TSOPERF   *TSOPROC   *TSORBA    *TSORGN    *TSOSIZE
 *TSOTIME   *TSOUNIT   *VLD-ACCT  *VLD-PROC  *WTP
 -- STATISTICS --
 *SEC-VIO    UPD-TOD
 -- CICS --
 *CICSCL    *CICSID    *CICSKEY   *CICSKEYX  *CICSPRI   *CICSRSL   *IDLE
 -- IMS --
```

```
-- IDMS --
*IDMSPROF  *IDMSPRVS
-- MUSASS --
*MUSDLID   *MUSID     *MUSOPT    *MUSPGM    *MUSUPDT
-- RESTRICTIONS --
*AUTHSUP3  *AUTHSUP4  *AUTHSUP5  *AUTHSUP6  *AUTHSUP7  *AUTHSUP8  *GROUP
*NOTE9     *OID       *PREFIX    *SHIFT     *SOURCE    *VMACCT    *ZONE
-- DFP --
*SMSINFO
```

From the ACF CONTROL and XREF settings, you can use this command to view the fields of infostorage records. Use the following syntax:

```
SHow Fields(recid)
```

For example, issue the command SHOW FIELDS(APPLDEF) from the CONTROL(GSO) setting to view the fields of the APPLDEF record.

LDS

Displays the LDS status and options, the active LDAP records, and the eTrust CA-ACF2 logonid field information that is propagated to an LDAP server. The SHOW LDS command uses standard eTrust CA-ACF2 masking conventions to specify a range of active LDAP records. The following example shows the results after issuing the SHOW LDS command with a MASKED qualifier value.

```
acf
show lds(cpu2-)


-- LIGHTWEIGHT DIRECTORY ACCESS PROTOCOL --

    CURRENT LDS STATUS:          ACTIVE
    CURRENT LDS JOURNAL STATUS:  ACTIVE
    LDS JOURNAL DATASET NAME:    CALDAP.LDSJRNL
    CURRENT LDS RECOVERY STATUS: ACTIVE
    LDS RECOVERY DATASET NAME:   CALDAP.LDSRCVR

    CURRENT SYSID:               TEST
    LDSRING:
    OPTIONS:   DEBUG  RETRY(3)  SYSCLASS(A)  SYSDEST(LOCAL) TIMEOUT(30)

    ACTIVE LDAP RECORD LIST:

    - LDAP.CPU2
      STATUS:    CONNECTED
      OPTIONS:   ACTIVE    BROADCST    CHANGE    DEBUG       DELCHILD
                 DELETE    INSERT      JOURNAL   PSWDASIS    USEEXTOP
      OBJCLASS:  ACF2LID
      NEXTKEY:   *** NO NEXTKEY DEFINED ***
      URL:       LDAP://xxx.xxx.xxx.xx:389
      USERDNS:   acf2lid=%l,host=xxx,o=xx,c=xx
      LDSLABEL:
      XREF:      LID:                ATTRIBUTE:
                 NAME                name
                 PASSWORD            USERPASSWORD
```

LINUX

Displays the Linux machine definitions currently in use.

```
show linux
-- LINUX/390 DEFINITIONS –

-- LINUX MACHINE –


MACHINE NAME     IP ADDRESS          ACTIVE
-------------    ----------------   ---------
patro07-1        176.345.123.57        YES
ussaaplb         176.345.123.68        YES
linux001         176.345.123.106       NO


-- LINUX USERS FOR NODE1 MACHINE --

UID#     NAME        GROUP       APPLICATION NAME
------   ---------   ---------   ------------------
1100     LNXUSER1    LNXGRP1     LINUXLONGUSERNAME
*500     GUEST11     GUSTGRP     - Not Defined -
501      TESTUSR3    TESTGRP     TESTGUY
505      TESTUSR4    n/a         - Not Defined -


-- LINUX USERS --

UID#     NAME        GROUP       APPLICATION NAME
------   ---------   ---------   ------------------
100      LNXUSER1    LNXGRP1     LINUXLONGUSERNAME
*500     GUEST1      GUSTGRP     - Not Defined -
501      TESTGUY3    TESTGRP     TESTGUY
505      TESTGUY4    n/a         - Not Defined -


-- LINUX GROUPS --

GID#   NAME
-----  ---------
200    LNXGRP1
*500   GUSTGRP
600    TESTGRP
601    DFTGRP
```

LINUX [ALL | DUPLICATE | GROUPS[(mmmm-nnnn)] | MACHNAME | USERS[(mmmm-nnnn)]]

Displays the Linux machine, Linux user, and Linux groups definitions.

- ALL—Displays Linux machine, Linux UID, and Linux GID definitions. This value is mutually exclusive with all other keywords, except DUPLICATE. ALL is the default.
- DUPLICATE—Displays Group and User definitions with duplicate UIDs and GIDs. This keyword is mutually exclusive with keyword and MACHNAME.

- GROUPS[(mmmm-nnnn])—Displays Linux Group definitions, which consists of Linux GID number and associated group name. Range value mmmm indicates lower boundary number and nnnn indicates upper boundary number. If range values are not specified, it displays all GIDs. An asterisk(*) before GID number indicates this is default Group. This keyword is mutually exclusive with keyword, ALL.
- MACHNAME—Displays defined Linux machine names with associated IP address and its current status. This keyword is mutually exclusive with keywords ALL and DUPLICATE.
- USERS[(mmmm-nnnn)]—Displays Linux User definitions, which consists of Linux UID number, eTrust CA-ACF2 User id, Group name, and LINUX/390 User Name. Range value mmmm indicates lower boundary number and nnnn indicates upper boundary number. If range values are not specified, it displays all UIDs. An asterisk(*) before UID number indicates this is the default User. This keyword is mutually exclusive with keyword ALL.

```
SHOW LINUX

--- LINUX/390 DEFINITIONS ---

-- LINUX MACHINE --

MACHINE NAME      IP ADDRESS          ACTIVE
---------------   --------------      -------

patro07-1         176.345.123.57      YES
ussaaplb          176.345.123.68      YES
linux001          176.345.123.106     NO


-- LINUX USERS --

UID#     NAME          GROUP        APPLICATION NAME
-------  ------       ---------    --------------------------------

100      LNXUSER1      LNXGRP1      LINUXLONGUSERNAME
*500     GUEST1        GUSTGRP      - Not Defined -
501      TESTGUY3      TESTGRP      TESTGUY
505      TESTGUY4       n/a          - Not Defined -


-- LINUX GROUPS --

GID#     NAME
-----    ----------

200      LNXGRP1
*500     GUSTGRP
600      TESTGRP
601      DFTGRP
```

MLID

The SHOW MLID subcommand of the ACF command displays the GSO MLID records and the @MLID macros in the ACFFDR.

```
show mlid
-- MLID DEFINITIONS --

   GSO MLID DEFINITIONS
   ====================

   MLID NAME: CICS01

   FULL LID      HEXADECIMAL OFFSET
   FIELD NAME    INTO THE MINI LID
   ____________  ____________________

   CICSCL        ......   0
   CICSID        ......   3
   CICSKEY       ......   6
   CICSRSL       ......   9
   CICSPRI       ......   C
   IDLE          ......   D
   CICS          ......   E
   CICSKEYX      ......   F


   ACFFDR MLID DEFINITIONS
   =======================

   MLID NAME: ACF2

   FULL LID      HEXADECIMAL OFFSET
   FIELD NAME    INTO THE MINI LID
   ____________  ____________________

   LID           ......   4
   NAME          ......   C
   PREFIX        ......  2C
   DSNSCOPE      ......  34
   LIDSCOPE      ......  3C
   UIDSCOPE      ......  44
   SCPLIST       ......  5C
   FLAG AT +9    ......  64
   FLAG AT +A    ......  65
   FLAG AT +8    ......  66
   R221PSWD      ......  20
   PRV-TOD1      ......  24
   ZONE          ......  67
   PASSWORD      ......  6A
   HOMENODE      ......  72
   FLAG AT +DC   ......  7A
```

```
MLID NAME: CICS

FULL LID      HEXADECIMAL OFFSET
FIELD NAME    INTO THE MINI LID
____________  __________________

CICSCL        ......   0
CICSID        ......   3
CICSKEY       ......   6
CICSRSL       ......   9
CICSPRI       ......   C
IDLE          ......   D
FLAG AT +DC   ......   E
CICSKEYX      ......   F


MLID NAME: IDMS

FULL LID      HEXADECIMAL OFFSET
FIELD NAME    INTO THE MINI LID
____________  __________________

FLAG AT +20B  ......   0
IDMSPROF      ......   1
IDMSPRVS      ......  21


MLID NAME: IMS

FULL LID      HEXADECIMAL OFFSET
FIELD NAME    INTO THE MINI LID
____________  __________________

IDLE          ......   0
FLAG AT +DC   ......   1
```

MLS

The SHOW MLS subcommand displays the status of MLS and security classifications on the active system.

```
show mls all

-- Multilevel Security (MLS) Facility --

    MLS Status:    ACTIVE
    MLS Mode:      QUIET
    Write-Down:    ALLOWED
    Current SYSID: XE41
    Seclabel by system ID: INACTIVE
    UNIX Files/Directories: SECLABELS NOT REQUIRED
    UNIX IPC Objects: SECLABELS NOT REQUIRED
```

```
-----------------
Active Categories
-----------------
Category Name
--------------------------------
  AA
  BB
  CC
  DD
  EE
  FF
  GG

----------------------
Active Security Levels
----------------------
Number Seclevel Name
------ ------------------------------------------------------------
 5    UNCLASSIFIED
 10   CONFIDENTIAL
 15   SECRET
 20   TOP SECRET

----------------------
Active Security Labels
----------------------

Label Name  Level  Categories
----------  -----  --------------------------------
PRJC2       10     AA
                   BB
                   CC
                   DD
                   EE
                   FF
                   GG
PRJS        15     *NONE*
PRJS3       15     AA
                   BB
                   CC
                   DD
                   EE
                   FF
PRJTS4      20     AA
                   BB
                   CC
                   DD
                   EE
                   GG
PRJTS5      20     AA
                   BB
                   CC
                   DD
                   EE
                   GG
  PRJU       5     *NONE*
  PRJU2      5     AA
                   BB
                   CC
                   DD
                   EE
                   FF
                   GG
```

MODE

Displays the current setting or mode of the ACF command. It also displays the current targets for ACF subcommands as follows:

```
show mode
 MODE: ACF  SYSID:  CPU1  TARGET:  CHI NY1 NY2
```

MUSASS

The SHOW MUSASS subcommand of the ACF command displays the GSO MLID records and the @MUSASS macros in the ACFFDR.

```
show musass
-- MUSASS DEFINITIONS --

GSO MUSASS DEFINITIONS
----------------------

 RECORD     MUSASS    MLID         CACHE  CVT           FAST  WORK
QUALIFIER     ID      NAME   CACHE   #    COM  CVTNAME  PATH   SP   WORKLEN
========== ======== ======== ===== ===== ===== ======== ===== ===== =======
APPL1      APPL1    MLID01    NO    0     YES  <NONE>    YES   0     0
APPL2      APPL2    MLID02    NO    0     YES  <NONE>    YES   0     0


ACFFDR MUSASS DEFINITIONS
-------------------------

 RECORD     MUSASS    MLID         CACHE  CVT           FAST  WORK
QUALIFIER     ID      NAME   CACHE   #    COM  CVTNAME  PATH   SP   WORKLEN
========== ======== ======== ===== ===== ===== ======== ===== ===== =======
N/A        CICSCVT  CICS      NO    0     YES  <NONE>    YES   0     0
N/A        IMS      IMS       NO    0     YES  <NONE>    YES   0     0
```

NJE(*nodename*)

Displays all NJE records defined to the system and the options specified for each. Providing a node name value limits the display to those NJE records associated with the given node name. Use standard eTrust CA-ACF2 masking conventions to specify a range of node names that you want to display.

```
show nje(chi)

 -- NJE OPTIONS IN EFFECT --

   NODE    VALIDATE  VALIDATE  INHERIT-    SEND      DEFAULT  SYSOUT
 NAME OR   INCOMING  OUTGOING    ANCE    ENCRYPTED  LOGONID  DEFAULT
   MASK      JOBS      JOBS    ALLOWED   PASSWORD             LOGONID
  (BOTH)     (IN)      (OUT)     (IN)      (OUT)      (IN)     (IN)
-=================================================================
 ********   YES        NO       YES        YES      SKKDFT   SKKSDFT
```

OMVS

[ALL | GROUPS(mmmm[-nnnn]) | SUPERUSERS | USERS(mmmm[-nnnn])] [Duplicates]]

Displays z/OS Unix System Services users and/or groups.

- **ALL**—displays all defined UIDs and GIDs along with their associated userids.
- **GROUPS(mmmm[-nnnn])**—displays a range of GID values along with their associated userids.
- **SUPERUSERS**—displays all superusers (UID of zero(0)) along with their associated userids.
- **USERS(mmmm[-nnnn])**—displays a range of UID values along with their associated userids.
- **Duplicates**—shows only the UID and GID values that belong to more than one user or group. The DUPLICATES keyword can be used together with another keyword. Example: SHOW OMVS USERS(1-2000) DUPLICATES will show only duplicate UID values that are in the range 1 to 2000.

ALL is the default.

```
Show omvs all

---------- OPENEDITION MVS DISPLAY -----------
              -- OMVS USERS --

          UID                NAME
          ================   ========
          0                  BPXAS
          0                  BPXOINIT
          0                  BPXROOT
          7                  GUEST3
          101                TEST
          8,888,888          OMVSC

            -- OMVS GROUPS --

          GID                NAME
          ================   ========
          0                  NULLGRP
          0                  ZEROGRP
          11                 LDSGRP
          44,444             OMVSG
          99,999,999         OMVSDGRP
```

PROGRAMS

Displays information on the following system options you have established for program name control:

- **Restricted program names**—SHOW PROGRAMS lists the names of those programs that bypass the operating system integrity. Execution of these programs is permitted only to users with the SECURITY privilege level and unlimited scope, or to users with the NON-CNCL privilege. See the descriptions of the SECURITY and NON-CNCL fields in the "Maintaining Scope Records" chapter.
- **Maintenance logonids/programs/libraries**—SHOW PROGRAMS lists the logonids for each user permitted to bypass access rule validation when executing the specified program from a specified library. The MAINT or NON-CNCL privilege is required in the logonid.
- **Tape bypass label programs/libraries**—SHOW PROGRAMS lists the names of programs that, when executed from a specified library, are valid for tape bypass label processing (BLP).
- **Logged programs**—SHOW PROGRAMS lists the names of programs for which each data set access is logged.

```
show programs

  -- RESTRICTED PROGRAM NAMES --
  DRWD**** FDR***   ICKDSF** IEHD****  IEHINIT*

  -- MAINTENANCE LOGONIDS/PROGRAMS/LIBRARIES --
  MAINTLID MAINTPGM SYS1.LINKLIB
  MAINTLID MAINTPG1 SYS1.LINKLIB
  MAINTLID MAINTPG2 SYS1.LINKLIB
  MAINTLID MAINTPG3 SYS1.LINKLIB
  MAINTLID MAINTPG4 SYS1.LINKLIB

  -- NO TAPE BYPASS LABEL PROGRAMS/LIBRARIES --

  -- LOGGED PROGRAMS --
  AMASPZAP
  IMASPZAP
  INCORZAP
```

If no programs exist under a certain category, SHOW PROGRAMS indicates that no such programs exist. For more information about these program controls, see the descriptions of the PPGM, MAINT, BLPPGM, and LOGPGM records in the chapter entitled, "Maintaining Global System Options Records."

PROXY

Displays current PROXY defaults used by Policy Directory Authorization Services and EIM defaults used by Enterprise Identity Mapping.

In the following example, EIM and PROXY defaults exist, but some fields have not been assigned values yet.

```
ACF
SHOW PROXY
 --DEFAULT PROXY INFORMATION SUMMARY

 BIND DISTINGUISHED NAME:    cn=eim administrator, o-CA, st=Illinois, c=US

 DISTINGUISHED DOMAIN NAME:  ibm=eimDomainName=EIM Test Domain,o=CA, st=Illinois, c=US
 LDAP SERVER URL AND PORT:   ldap://usi243me.ca.com:1389
 LOCAL REGISTRY:             RACF XE43
 OPTIONS:                    ENABLE

 --DEFAULT EIM INFORMATION SUMMARY

 BIND DISTINGUISHED NAME:    NONE
 DISTINGUISHED DOMAIN NAME:  NONE
 LDAP SERVER URL AND PORT:   NONE
 LOCAL REGISTRY:             NONE
 OPTIONS:                    ENABLE
```

REALM

Displays the GSO REALM records that are defined on the system.

```
Acf
SHOW REAlm
-- REALM GSO RECORD DEFINITIONS --

REALM.KERBDFLT          LOCAL REALM
                        CURRENT KEY VERSION = 25        DEFAULT TICKET = 7,200
                        MINIMUM TICKET = 15             MAXIMUM TICKET = 14,400
                        REALM NAME = USI243ME.CA.COM

 REALM.MVXE43A          FOREIGN REALM
                        CURRENT KEY VERSION = 2
                        REALM NAME = /.../USI243ME.CA.COM/KRBTGT/CA.COM

 REALM.MVXE75           FOREIGN REALM
                        CURRENT KEY VERSION = 0
                        REALM NAME = USI275ME.CAI.COM
```

RESIDENT

Displays the names of system-resident directories and access rules:

- **Resident directories**—Lists the resource rule directories that are built and made globally resident. This section also indicates the rule sets associated with the directories. Resident rule sets are global (in common storage) or local (in an address space). The GSO INFODIR and RESDIR records determine which infostorage directories and resource rule sets are resident. The INFODIR record replaces the RESDIR record. Although still accepted, we recommend that you convert RESDIR records to INFODIR records.

- **Resident infostorage directories**—Indicates the rule directories in the Infostorage database (such as eTrust CA-ACF2 for DB2 directories) that are built and made globally resident. This section also indicates the rule sets associated with the directories. Resident rule sets are global (in common storage) or local (in an address space). The GSO INFODIR record determines which infostorage records and directories are made resident.
- **Resident access rules**—Lists the access rule sets that are resident in global storage. The GSO RESRULE record determines which rule sets are resident.

```
show resident

  -- RESIDENT DIRECTORIES --
  CKC, RULES GLOBALLY RESIDENT

  -- RESIDENT INFOSTORAGE DIRECTORIES --
  DPLN, RECORDS LOCALLY RESIDENT       DTBL, RECORDS LOCALLY RESIDENT
  DDBS, RECORDS GLOBALLY RESIDENT      DBPL, RECORDS TRANSIENT

  -- RESIDENT ACCESS RULES --
  PAY        ABC         SYS1
```

RSRCTYPE(null|D|R)

Displays all of the resource type codes that are defined in your Infostorage database.

- **null**—displays all R type and DB2 type resource types defined.
- **D**—displays all DB2 type resource types defined.
- **R**—displays all R type resource types defined.

The following example displays all R and DB2 type resource types defined:

```
show rsrctype


-- RESOURCE TYPES DEFINED --

RAB*  RALU  RCFC  RCHG  RCKC  RCMR  RCPC  RCTD  RCTS  RCXD  RCXM  RDAH  RDB2
RDFC  RDPN  RDSM  RDSN  RDTB  RFAC  RIAG  RICM  RIPS  RISF  RITR  RJOK  RJWP
RKKK  RLBM  RLLL  RMGM  RMTP  ROMC  ROPR  RPDS  RPGM  RPRO  RRCM  RRSC  RSAF
RSAS  RSDS  RSEG  RSFP  RSTS  RSUR  RTER  RTGR  RTPR  RTPV  RTP1  RTST  RTWC
RVLB  RVMA  RXCD  RXDC  RXXX  RXYZ  R@D@

TOTAL NUMBER OF
RESOURCE TYPES DEFINED:  59


-- DB2 RESOURCE TYPES DEFINED --
DBPL  DCOL  DDBS  DPKG  DPLN  DSTG  DSYS DTBL  DTSP

TOTAL NUMBER OF DB2
RESOURCE TYPES DEFINED:   9
```

RSVWORDS

Displays the Reserved Word Prefix List. This list defines the words or prefixes that are not allowed in the specification of a password. See the description of the RESWORD record in the chapter, "Maintaining Global System Options Records."

```
show rsvwords

 -- RESERVED WORD PREFIX LIST -
APPL        APR         ASDF        AUG         BASIC       CADAM       DEC
 DEMO       FEB         FOCUS       GAME        IBM         JAN         JUL
 JUN        LOG         MAR         MAY         NET         NEW         NOV
 OCT        PASS        ROS         SEP         SIGN        SYS         TEST
 TSO        VALID       VTAM        XXX         1234
```

SAFDEF(null | *id* | REQ=*xxxx*)

Displays the SAFDEF records that are defined on your system. Valid values are:

- **null**—displays all SAFDEF records in the order they are searched by eTrust CA-ACF2.
- ***id***—displays a specific SAFDEF record. If you specify a mask, eTrust CA-ACF2 displays a group of SAFDEF records.
- **REQ=*xxxx***—displays all SAFDEF records for a specific RACROUTE request. The *xxxx* must specifically match a valid RACROUTE request parameter.

SHOW SAFDEF describes how SAFDEF is used by eTrust CA-ACF2 during processing. However, LIST SAFDEF under SET CONTROL(GSO) and SHOW SAFDEF can differ such as when a SAFDEF for FASTAUTH specifies an ENTITY value for RACROUTE, even though this is not allowed. See the chapters, "Understanding SAF" and "Maintaining Global System Options Records" for detailed information.

```
show safdef

 -- SYSTEM AUTHORIZATION FACILITY DEFINITIONS --

 JESPOOLR JOBNAME=********   USERID=********   PROGRAM=HA$PSUBS   RB=HA$PSUBS
          RETCODE=0          SAFDEF=INTERNAL   MODE=IGNORE        SUBSYS=ACF2

          RACROUTE REQUEST=AUTH,REQSTOR='RDRSYSDS',SUBSYS='JES2-   ',
          RACROCLASS='JESSPOOL'

 AUTJ2RDR JOBNAME=********   USERID=********   PROGRAM=HA$PSUBS   RB=HA$PSABS
          RETCODE=0          SAFDEF=INTERNAL   MODE=IGNORE        SUBSYS=MACS

          RACROUTE REQUEST=AUTH,REQSTOR='RDRSYSDS',SUBSYS='JES2-   ',
          RACROCLASS='JESSPOOL'

 J2RDRVYX JOBNAME=********   USERID=********   PROGRAM=HA$PSUBS   RB=HA$PSUBS
          RETCODE=4          SAFDEF=INTERNAL   MODE=GLOBAL        SUBSYS=ACF2

          RACROUTE REQUEST=VERIFYX,REQSTOR='RDRVERYX',SUBSYS='JES2-    '
```

STATE

Displays the eTrust CA-ACF2 system options in effect.

```
ACF
show state
```

```
RUNNING eTrust CA-ACF2 8.0 /MVS SP7.0.5; WITH MODE = ABORT
USING FDR ASSEMBLY:  11.05  02/13/04

OPTIONS IN EFFECT:
%CHANGE=ALLOWED              ACCESS SUBCMD=DISABLED          CACHE SYNCHRONIZER=DISABLED
CONTROL=DECENTRALIZED        CPF=DISABLED                    CPUTIME=LOCAL
DATABASE CACHE=DISABLED      DATE FORMAT=MM/DD/YY            DDB=DISABLED
DFT LID=BATCHDFT             DFT PRIM LANG=ENU               DFT SECND LANG=ENU
DFT STC LID=ACFSTCID         DYNAMIC COMPILE=DISABLED        ETRUST AUDIT=ENABLED
JOB CHECK=NO                 KERBLVL(0)                      LDS=DISABLED
LID WARN DAYS=0              MAX VIO PER JOB=10              NON-VSAM ERASE=NO
NOSORT=YES                   OMVS DFT LID=OMVSU              OMVS DFT GRP=OMVSG
RPTSCOPE=OFF                 RULELONG=DISABLED               STC OPTION=ON
SYSPLEX=DISABLED
SYSPLEX ALTERNATE STRUCTURE NAME=N/A
SYSPLEX PRIMARY STRUCTURE NAME=N/A
TAPE BLP=NOLOG               TAPE DSN=NO                     TEMPDSN=BYPASS
TNG MONITOR=DISABLED         UADS=BYPASS                     VSAM ERASE=NO
VTAM OPEN=NO                 XAPPLVLD=NO                     XCF GROUP NAME=TESTXCF

PASSWORD OPTIONS IN EFFECT:
LOGON RETRY COUNT=4                  MAX PSWD ATTEMPTS=10          MIN PSWD LENGTH=1
REPEAT PAIR CHAR=0                   REQ ALPHBET CHAR=NO           REQ NUMERIC CHAR=NO
NONALPHANUMERIC CHARACTER(S)         ALLOWED=NONE
PSWD ALTER=YES                       PSWD CMD CHANGE=ALLOW         PSWD EXTRACT=NO
PSWD FORCE=YES                       PSWD HISTORY=NO               PSWD-JES=OFF
PSWD-LID=NO                          PSWD-MAX=0                    PSWD-MIN=0
MIXED CASE PASSWORDS=NO              PSWD NUMERIC=NO               PSWD REQUIRED=NO
PSWD RESERVE WORD=NO                 PASSWORD SIMILARITY=0         PSWD SPLIT=NO
PSWD VERIFY=NO                       PSWD VOWEL CHAR=ALLOW         PSWD WARN DAYS=1
EXTENDED PASSWORD HISTORY=INACTIVE   EXTENDED PASSWORD HISTORY #=0
AGE TEMPORARY PASSWORDS=YES

UID STRING = COMPANY,SITE,LEVEL,PROJECT,LID

DECOMP AUTHORITY = SECURITY, AUDIT

INFO LIST AUTHORITY = SECURITY, AUDIT

VOLUME PSEUDO DSN= @VOLSER.VOLUME

-- DSNAME PROTECTED VOLUMES –
******

-- VOLSER PROTECTED VOLUMES --
NONE SPECIFIED

-- AUTOMATIC ERASE VOLUMES --
NONE SPECIFIED

-- PDS MEMBER-LEVEL PROTECTION: LIBRARY / VOLUME / RESOURCE TYPE --
NONE SPECIFIED
```

For more information about these options, see the descriptions of the PSWD, OPTS, RESVOLS, RULEOPTS, and SECVOLS records in the chapter, "Maintaining Global System Options Records." Also, see the description of the @UID macro in Appendix A, "eTrust CA-ACF2 Field Definition Record Generation," of the *Getting Started* guide.

STC

Displays the logonid and groupid of specific started task IDs.

```
acf
show stcid
-- STARTED TASK TABLE --

STCID     LOGONID   GROUP
======    ========  =======
WEBSRV    IMWEBSRV  IMWEBSRV
CICSA     CICS
```

SYSPLEX

Displays information about the current settings for the SYSPLEX feature. It also displays the number of times that eTrust CA-ACF2 has used the XES feature and the number of times messages have been sent or retrieved through XCF. See SYSPLEX Environment and Options (SYSPLEX) in the "Maintaining Global System Options Records" chapter for more information.

```
show sysplex


-- SYSPLEX COUPLING FACILITY --
OPTION: SYSPLEX
CURRENT XES STATUS: ACTIVE
CURRENT XCF STATUS: ACTIVE

COUPLING FACILITY DATA:
 INFOSTORAGE: INACTIVE
 LOGONIDS:    ACTIVE
 RULES:       ACTIVE

 XCF GROUP NAME: XCFACF

 PRIMARY STRUCTURE NAME: STRUCT1
 ALTERNATE STRUCTURE NAME: N/A
 CURRENT STRUCTURE SIZE= 512K
 MAX STRUCTURE SIZE= 3,840K
 NUMBER OF STRUCTURE ENTRIES= 98

 NUMBER OF XES WRITES=  1,844
 NUMBER OF XES READS=   3,449
 NUMBER OF XES DELETES= 1,203
 NUMBER OF XCF MESSAGES SENT= 5
 NUMBER OF XCF MESSAGE GETS=  3
```

SYSTEMS

Displays various system parameters, such as the eTrust CA-ACF2 SVC numbers and SMF record numbers.

```
show systems

  -- SYSTEM PARAMETERS IN EFFECT --

  SVCS:
  ALTER SVC=222           VALIDATE SVC=221

  SMF RECORD NUMBERS:
  PASSWORD=220            DATA SET VIO=221        LID JOURNAL=222
  RULE JOURNAL=223        LID TRACE=224           TSO COMMAND=225
  INFO JOURNAL=226        RESOURCE VIO=227        ACF2 COMMON=230

  BACKUP:
  AUTO BACKUP TIME=03.30  CPUID=UCC1
  WORK FILE UNIT=VIO      PRIMARY SPACE=5         SECONDARY SPACE=005
  COMMAND STRING=S REPROALT

  OTHER:
  CONSOLE MSGS=ROLL       SHR-DASD=SUPPORTED      SMF LOGONID STAMP=NO
  NOTIFY=YES              CURRENT SYSID=ABC1      STARTUP SYSID=ABC1
  BUILT ACCVT=ABC1
```

TNG

Displays the TNG nodes on the system.

```
show tng

  -- TNG NODE DEFINITIONS --

  NODE NAME    DEBUG      IP ADDRESS
  ---------    -----      ----------

  New York      NO        123.456.789.123
  Dallas        NO        123.456.789.345
  Chicago       YES       123.456.789.789
```

TSO

Displays TSO default options on the system.

```
show tso

-- TSO RELATED DEFAULTS ACTIVE --
LOGON ACCOUNT STRING=1  CMD LIST BYPASS CHAR=#    CHAR DELETE CHAR=NONE
TSO CMD LIST=NONE       COMMAND SMF RECORDS=NO    LINE DELETE CHAR=NONE
LOGON CHECK=NO          PERFORMANCE GROUP=NONE    TSO LOGON PROC=IKJACCNT
QUICK LOGON=YES         TSO REGION SIZE=1024      SUBMIT CLASS=NONE
SUBMIT HOLD CLASS=NONE  SUBMIT MESSAGE CLASS=NONE SESSION TIME=NONE
SYSOUT CLASS=A          TSO UNITNAME=SYSDA        LOGON WAIT TIME=60
FSRETAIN=YES
```

UNIXOPTS

Displays UNIX options on the system.

```
show unixopts
```

```
-- UNIXOPTS OPENEDITION/MVS/UNIX SYSTEM SERVICES (USS) SUMMARY --
OMVS DEFAULT USER: OMVSU
OMVS DEFAULT GROUP: OMVSG
MAX NUMBER OF OMVS GROUPS: 300
HFS SECURITY ACTIVE: NO
HFSACL ACTIVE: NO
FILE.GROUPOWNER.SETGID ACTIVE: NO

-- AUDIT FLAG STATUS --
CHOWN_RESTRICTED: YES
DIRACC_ACTIVE:    NO
DIRSRCH_ACTIVE:   NO
FSOBJ_ACTIVE:     NO
FSSEC_ACTIVE:     NO
IPCOBJ_ACTIVE:    NO
PROCACT_ACTIVE:   NO
PROCESS_ACTIVE:   NO
```

ZEROFLDS

Displays those fields of the logonid record that cannot be copied by the ACF subcommand INSERT USING (under the LID setting).

```
show zeroflds
```

```
-- FIELD VALUES WHICH WILL NOT BE COPIED DURING 'INSERT USING' PROCESSING --
ACC-CNT    ACC-DATE   ACC-SRCE   ACC-TIME   ACCOUNT    ACCTPRIV   ACF2CICS
AUDIT      AUTHSUP1   AUTHSUP2   AUTHSUP3   AUTHSUP4   AUTHSUP5   AUTHSUP6
AUTHSUP7   AUTHSUP8   AUTOALL    AUTODUMP   AUTONOPW   AUTOONLY   BDT
CMD-PROP   CONSOLE    CONSULT    DG84DIR    DIALBYP    GROUP      GRP-OPT
GRP-USER   GRPLOGON   HOMENODE   JOBFROM    KERB-VIO   KERBCUR    KERBCURV
KERBPRE    KERBPREV   LDS        LEADER     LIDTEMP    LIDZMAX    LIDZMIN
LOGSHIFT   MAINT      MOUNT      MUSASS     MUSUPDT    NAME       NO-SMC
NO-STATS   NOMAXVIO   NON-CNCL   NOSPOOL    OPERATOR   PASSWORD   PHONE
PPGM       PRIV-CTL   PRV-TOD1   PRV-TOD2   PRV-TOD3   PRV-TOD4   PRVPSWD1
PRVPSWD2   PRVPSWD3   PRVPSWD4   PSWD-DAT   PSWD-INV   PSWD-MIX   PSWD-SRC   PSWD-
TIM   PSWD-TOD   PSWD-VIO   PSWD-XTR   PSWD-XTV   READALL    REFRESH    RSRCVLD
RULEVLD    SCPLIST    SEC-VIO    SECURITY   SHIFT      SRF        SYNCNODE
SYSPEXCL   SYNERR     TAPE-BLP   TAPE-LBL   TDISKVLD   TSORBA
UNICNTR    UPD-TOD    VAX        VLDVMACT   VMACCT     VMD4AUTH   VMD4RSET
VMD4TARG   VMSAF      VMSFS      VSESRF     ZONE
```

To alter the fields included on this list, see the description of the @CFDE macro in Appendix A, “eTrust CA-ACF2 Field Definition Record Generation,” of the *Getting Started* guide.

SN Subcommand

This subcommand interfaces with the TSO SEND command. The syntax of the SN subcommand is:

```
SN 'message'   [Operator(2|routecode)]
               [User(*|logonid,...,logonid)]
               [CN(console-id)]
               [NOW|Logon|Save]
               [NOWait|Wait]
```

You can issue this subcommand under any setting of the ACF command. For a description of SN subcommand parameters, see the description of the TSO SEND command in the *IBM TSO Command Language Reference* guide.

SYNCH Subcommand

The SYNCH subcommand is described in the "Maintaining Logonid Records" chapter.

The ACF Command Using ISPF Panels

To use the eTrust CA-ACF2 ISPF panels, specify ISPF from the TSO READY prompt. The following panel appears:

```
-----------------------  ISPF/PDF PRIMARY OPTION MENU  ------------------------
OPTION  ===>

   0  ISPF PARMS     - Specify terminal and user parameters    TIME    - 15:53
   1  BROWSE         - Display source data or output listings  TERMINAL- 3278
   2  EDIT           - Create or change source data            PF KEYS - 12
   3  UTILITIES      - Perform utility functions               JULIAN  - 98.282
   4  FOREGROUND     - Invoke language processors-Foreground   DATE    - 98/10/09
   5  BATCH          - Submit job for language processing      PREFIX - ACT001
   6  COMMAND        - Enter TSO command or CLIST              PROC    - $PRDISPF
   7  DIALOG TEST    - Perform dialog testing                USERID    - ACT001
   8  LM UTILITIES   - Perform library management utility functions
   9  IBM PRODUCTS   - Additional IBM program development products
   A  eTrust CA-ACF2 - Perform ACF2 processing
   C  CHANGES        - Display summary of changes for this release
   T  TUTORIAL       - Display information about ISPF/PDF
   X  EXIT           - Terminate SPF using list/log defaults




Enter END command to terminate ISPF. Enter option A to perform CA-ACF2
processing.
```

If your site does not have ISPF or you want to administer eTrust CA-ACF2 from native TSO, see The ACF Command earlier in this chapter for an overview of the ACF command.

eTrust CA-ACF2 ISPF Option Selection Menu

The following panel displays all the options for processing eTrust CA-ACF2 records:

```
ACFOPTS------ eTrust CA-ACF2 Security ISPF Option Selection Menu-------
OPTION  ===>

   1  RULES     - PROCESS eTrust CA-ACF2 ACCESS AND GENERALIZED RESOURCE RULES
   2  LOGONIDS  - eTrust CA-ACF2 LOGONID CREATION/MAINTENANCE FACILITY
   3  SYSTEM    - eTrust CA-ACF2 SHOW COMMANDS
   4  REPORTS   - eTrust CA-ACF2 REPORT PROGRAM PROCESSOR
   5  UTILITIES - PROCESS eTrust CA-ACF2 UTILITIES
   6  GSO       - GLOBAL SYSTEM OPTIONS SERVICES
   7  NET       - NETWORKING SYSTEM OPTIONS SERVICES
   8  CAC       - MVS DATABASE CACHE RECORD SERVICES
   9  XREF      - SOURCE/RESOURCE AND GROUP RECORD SERVICES
  10  MAC       - MANDATORY ACCESS CONTROL ADMINISTRATION
  11  CPF       - COMMAND PROPAGATION FACILITY SERVICES
  12  FIELD     - RECORD LEVEL PROTECTION CONTROLS
  13  TARGETS   - SET CPF TARGET NODES, DEFAULTS IN USE
  14  PROFILE   - PROCESS PROFILE INFORMATION RECORDS
  15  SMS       - PROCESS DFSMS SUPPORT RECORDS
  16  ENTRY     - PROCESS ENTRY SOURCE RECORDS
  17  SHIFT     - PROCESS SHIFT/ZONE RECORDS
  18  RACDCERT  - PROCESS KEYRING/CERTIFICATE COMMANDS
  19  C-CIC     - PROCESS C-CIC CICS INITIALIZATION RECORDS
  20  LDS       - PROCESS LDAP DIRECTORY SERVICES
```

Options

1 RULES
The panels for processing rules are described in the chapters, “Maintaining Access Rules” and “Maintaining Resource Rules.”

2 LOGONIDS
The panels for processing logonids are described in the chapter entitled, “Maintaining Logonid Records.”

3 SYSTEM
The output for the SHOW subcommand is described later in this chapter.

4 REPORTS
The panels for the eTrust CA-ACF2 report generators are described in the *Reports and Utilities Guide*.

5 UTILITIES
The panels for the eTrust CA-ACF2 utilities are described in the *Reports and Utilities Guide*.

6 GSO
The panels for processing global system options records are described in the “Maintaining Global System Options Records” chapter.

7 NET
The panels for processing distributed database records are described in the *Distributed Database Support Guide*.

8 CAC
The panels for processing cache records are described in the "Maintaining Cache Records" chapter.

9 XREF
The panels for processing cross-reference records are described in the chapter "Maintaining Cross-Reference Records."

10 MAC
The panels for processing mandatory access control records are described in the *MAC Administrator Guide*.

11 CPF
The panels for processing command propagation facility records are described in the "Using the Command Propagation Facility" chapter.

12 FIELD
The panels for processing field records are described in the "Maintaining Field Records" chapter.

13 TARGETS
The TARGETS option applies to all options except RULES, SYSTEM, REPORTS, and UTILITIES. The TARGETS option lets you specify the target nodes where you want a particular ACF command to take effect. This feature is known as command propagation. For details on how to use the command propagation facility (CPF), see the "Using the Command Propagation Facility" chapter.

14 PROFILE
The panels for processing profile information records are described in the "Maintaining Profile Records" chapter.

15 SMS
The panels for processing DFSMS records are described in the "Implementing DFSMS Support" chapter.

16 ENTRY
The panels for processing entry source records are described in the "Maintaining Entry Source and Source Group Records" chapter.

17 SHIFT
The panels for processing shift and zone records are described in the "Maintaining Shift and Zone Records" chapter.

18 RACDCERT
The panels for processing KEYRING records are described in "Maintaining Profile Records" and "z/OS Unix System Services Support."

19 **C-CIC**
The panels for processing C-CIC CICS initialization records. These records are described in the *CICS Support Guide*.

20 **LDS**
The panels for processing LDS LDAP records, LDS XREFLDAP records, and LDS OPTIONS records.

The ACF Command in Batch

A z/OS site can execute the ACF command in batch by using the ACFBATCH utility. In addition to executing the ACF command, the user can execute ACF subcommands, such as INSERT, CHANGE, LIST, and DELETE.

For example, an authorized user can establish a new logonid record with a job stream, such as:

```
//ACFJOB  EXEC PGM=ACFBATCH
//SYSPRINT DD SYSOUT=A
//SYSLBC   DD DSN=SYS1.BRODCAST,DISP=SHR
//SYSHELP  DD DSN=SYS1.HELP,DISP=SHR
//SYSIN    DD *
SET LID
INSERT USING(PAYSEC) PAYJSD NAME(JANE S. DOE) LEADER PHONE(EXT. 458) TSO
/*
```

The same facility is also available through the execution of the Terminal Monitoring Program (TMP) in background.

```
//ACFJOB  EXEC PGM=IKJEFT01,DYNAMNBR=25
//SYSTSPRT  DD SYSOUT=A
//SYSHELP   DD DSN=SYS1.HELP,DISP=SHR
//SYSLBC    DD DSN=SYS1.BRODCAST,DISP=SHR
//SYSTSIN   DD *
ACF
SET LID
INSERT USING(PAYSEC) PAYJSD NAME(JANE S. DOE) LEADER PHONE(EXT. 458) TSO
/*
```

For more information about the ACFBATCH program, see the "Utilities for eTrust CA-ACF2 Administration" chapter in the *Reports and Utilities Guide*.

When using any ACF2 commands in batch jobs, CLISTs or EXECs, we recommend that the command be spelled out in its entirety. Any new commands that might change the minimum number of characters to be specified for a command will not affect site applications.

The ACFM CICS Transaction

The eTrust CA-ACF2 CICS interface provides a transaction called ACFM that permits maintenance of most eTrust CA-ACF2 records in CICS conversational mode. This transaction provides a function called command processor (CP). The CP function lets an authorized user enter subcommands under CICS similar to the ACF subcommands used for logonid record processing under TSO.

For further information about logonid maintenance using the ACFM transaction, see the *CICS Support Guide*.

The ACF IMS Transaction

The eTrust CA-ACF2 IMS interface provides an ACF conversational transaction that lets you maintain logonid records directly from an IMS terminal. A user must be signed on with the /SIGN command to use the ACF transaction. See the *IMS Support Guide* for a description of the ACF IMS transaction.

Chapter 2

Controlling System Entry

You must define a unique logonid record for each user of your system. eTrust CA-ACF2 uses the information you specify in a user's logonid record to determine if that user can access the system. This chapter provides an overview of what eTrust CA-ACF2 checks at system entry and it describes in detail how to log on to a system protected by eTrust CA-ACF2.

Specifically, this chapter describes what eTrust CA-ACF2 checks at logon. It also describes:

- TSO logon
- CICS sign-on
- IMS sign-on
- Submitting batch jobs
- Specifying a group or project name at logon
- Specifying multi-value logonid fields and UID strings
- How users can determine unauthorized use of their logonids

In addition, this chapter introduces the following topics and directs you to other guides in the documentation set where you can find more detailed information:

- Providing extended user authentication
- Choosing the UADS or NOUADS option
- Providing dynamic logonid privileges

The logon procedures for Advantage CA-Roscoe, WYLBUR, and other online systems are similar to those for TSO. Check with your site for exact procedures.

What eTrust CA-ACF2 Checks at Logon

eTrust CA-ACF2 uses the information you specify in a user's logonid record to determine if that user can access the system. If MLS is active on the system, eTrust CA-ACF2 also uses information specified in the User and Seclabel Profile records and checks for and validates a security label, if one has been specified or defaulted, to determine if that user can access the system. For information on how to implement MLS on a system, see the *Multilevel Security Planning Guide*.

Logonids

To gain entry to the system, the user must provide a logonid that has not been canceled or suspended and has a valid password. The logonid uniquely identifies each authorized system user. Logonids can be from one to eight characters. If your site requires a user to log on to the system from a specific source or source group, eTrust CA-ACF2 verifies that the input device is in the source group for that user. If it is not in his source group it denies access to the system. No source checking is done for started task logonids during system entry validation. eTrust CA-ACF2 also checks the SHIFT field to ensure that the user is attempting access during an acceptable time of day. If not it denies access to the system. If you specify the LOGSHIFT privilege and the user logs on outside his shift, eTrust CA-ACF2 creates a logging record. If a logonid contains a multi-value field, eTrust CA-ACF2 can validate each value when necessary. See the *Getting Started Guide* for instructions on implementing multi-value logonid fields.

Passwords

In addition to checking these logonid record fields, eTrust CA-ACF2 ensures that the user supplies his own logonid by checking the password he enters. Passwords authenticate the identity of a logonid. A password can be from one to eight characters. Passwords can contain any alphabetic and any numeric character. eTrust CA-ACF2 also allows U.S. National characters (@, $, and #) and user defined non-alphanumeric characters. See the description of the PSWDPLST field of the Password Maintenance and Support (PSWD) record in the "Maintaining Global System Option Records" chapter for a listing of these characters. eTrust CA-ACF2 recognizes the national characters as the following hexadecimal characters:

National Character	Hexadecimal Character
@	X'7C'
$	X'5B
#	X'7B'

In countries other than the United States, these U.S. National characters represented on terminal keyboards might generate a different hexadecimal representation and cause an error when entered. For example, in some countries, the dollar sign ($) character can generate as x'4A'. eTrust CA-ACF2 does not accept other characters. Before processing the password information, it encrypts the password with a one-way method that cannot be deciphered.

Although security administrators can assign new passwords, they cannot display a user's current password. eTrust CA-ACF2 also checks the password-related fields in the user's logonid before it decides to permit the user access to the system. For example, it checks to see if the password has expired, if it should suspend the password due to violations, or whether it should warn the user when his password will expire.

UID String

eTrust CA-ACF2 also builds a user identification string (UID) for the user at logon. The user identification string identifies a user or group of users to reduce the number of access and resource rules you must write.

Security Label

If MLS is active on the system, a user may specify a security label or one may be defaulted for them. This security label is validated and endures for the duration of the user's session. For information on how to implement MLS on a system, see the *Multilevel Security Planning Guide*.

TSO Logon

To log on to the IBM Time Sharing Option (TSO), enter your logonid and password. In the following example, system responses are capitalized, and information entered by the user is in lowercase.

```
LOGON logonid

ACF82004 ACF2, ENTER PASSWORD - your-password (entered into a nondisplayable
field)
```

eTrust CA-ACF2 completes its logon processing. To abort a TSO logon, enter a plus sign (+) in response to any eTrust CA-ACF2 prompt.

Your site can permit the LOGON command, logonid, and password to be entered on one line, such as:

```
LOGON logonid/password
```

If MLS is active on the system, a user may also specify a security label on the LOGON command, such as:

```
LOGON logonid/password SECLABEL(seclabel)
```

Important! Before security labels can be processed at logon when MLS is active on a system, you MUST link authorized logon pre-prompt exit, IKJEFLD1, into LPALIB. eTrust CA-ACF2 will dynamically link authorized logon pre-prompt exit, IKJEFLD1 into LPALIB for you when you create a CONTROL(GSO) TSO record and specify IKJEFLD1 in it. This lets you use the authorized logon pre-prompt exit. You must perform the eTrust CA-ACF2 REFRESH command to activate the IKJEFLD1 facility. The default is NOIKJEFLD1, which indicates eTrust CA-ACF2 will not dynamically link the authorized logon pre-prompt exit.
Note: Once activated, an IPL is required to deactivate the IKJEFLD1 facility. Do not do this unless you also deactivate MLS. For more information, see the *System Programmer's Guide*.

```
acf
 ACF
set control(gso)
 CONTROL
change tso ikjefld1
```

To activate the changes to the TSO record, issue the following command:

```
f acf2,refresh(tso)
```

*Warning! If you want to use a different security label to logon, you **must** logoff first and then logon again with that security label. You cannot change your session security label by reconnecting to a TSO session with a different security label.*

For more information on how to gain entry to a system in an MLS environment, see the *Multilevel Security Planning Guide.*

We recommend that you do not permit one-line logons because the password is **visible** on the screen. Anyone standing nearby can read it. Your site can force the use of a separate entry by specifying the NOQLOGON option in the GSO TSO record. For more information, see "Maintaining System Global Options Records" chapter.

The SHOW TSO and SHOW ACF2 subcommands of the ACF command display the TSO options as specified in the GSO TSO record.

Changing Your Password

To change your password, perform the following steps:

1. Follow your usual logon procedure until the system prompts you with the following message:

   ```
   LOGON logonid

   ACF82004 ACF2, ENTER PASSWORD -
   ```

2. Enter your old password, a slash, and your new password:

   ```
   oldpassword/newpassword
   ```

   The password must be from one to eight characters long. Passwords can contain alphanumeric and national characters. The system prompts:

   ```
   ACF82020 ACF2, REENTER NEW PASSWORD FOR VERIFICATION -
   ```

3. Enter your new password again.

   If you reenter your new password successfully, the system displays:

   ```
   ACF82000 ACF2, LOGON IN PROGRESS
   ACF01129 PASSWORD SUCCESSFULLY ALTERED
   ```

   The logon procedure continues as usual.

   If you reenter your new password incorrectly, the system displays:

   ```
   ACF82916 ACF2, VERIFICATION OF NEW PASSWORD FAILED
   ACF82008 ACF2, ENTER NEW PASSWORD -
   ```

   You must enter your **new** password, and then enter it again when the system asks you to reenter it for verification. If you successfully reenter your new password, the logon procedure continues as usual.

TSO Logon Parameters

You can also logon to TSO by entering the LOGON command, your logonid, and any chosen operands directly on the command line as follows:

```
logon your-logonid logon-operand1 logon-operand2......
```

For example, to reconnect to a TSO session that was disconnected, USER01 might enter:

```
logon user01 reconnect
```

The TSO LOGON command parameters are described in the following section. These parameters can be specified with the logonid in response to the ACF82033 prompt, in the non-display field with the password in response to the ACF82044 prompt, or on the command line. It is also possible to pass these parameters from VTAM USS if your site is set up to connect users to TSO through a site-defined command.

Parameter Descriptions

ACCT(*acct-number*)
Specifies the account number associated with this TSO session. This value overrides any default from the TSOACCT field of the user's logonid record or the GSO TSO record. To specify ACCT during logon, the logonid must have the LGN-ACCT privilege. To validate the account number you must write resource rules with the type code TAC and specify the VLD-ACCT field in the logonid record.

FORCE
Indicates the user wants to gain TSO access to the system even though eTrust CA-ACF2 has not yet been started. Because it is unavailable, the user must be defined in the UADS data set. This parameter is intended for use in recovery situations when the system is IPLed without eTrust CA-ACF2.

FSCREEN|NFSCREEN
Prevents the user from being presented with a full-screen logon panel. NFSCREEN can be used during a specific logon to override the TSOFSCRN privilege specified in the logonid record; however, specifying FSCREEN at logon time does not override the NOTSOFSCRN privilege for a logonid and does not display the full-screen panel.

GROUP(*groupname*)
Specifies the one to eight-character group or project name associated with this logonid for this system access session. To specify GROUP during logon, enter GROUP(*grpname*) on the command line, or fill in the GROUP field on the TSO full-screen logon panel. eTrust CA-ACF2 validates the GROUP value you enter by checking resource rules with the type code TGR. For more details, see Specifying a Specifying a Group or Project at Logon later in this chapter.

MAIL|NOMAIL
Indicates to TSO whether the user wants to see any messages from TSO at logon time.

MSGCLASS(*messageclass*)
Specifies the one-character message class for this TSO session. The user must have the LGN-MSG attribute to specify MSGCLASS.

NOTICES|NONOTICES
Indicates to TSO whether the user wants to receive TSO notices at logon time.

PERFORM(*perfgroup*)
Specifies the TSO performance group to be used during the session. Enter a value from one to 250.

PROC(*procedure*)
Specifies the procedure to be used for this TSO session. This value overrides any default from the TSOPROC field of the logonid record or the GSO TSO record. To specify PROC during logon, the user must have the LGN-PROC privilege. To validate the PROC value, you must write resource rules with the type code TPR and specify the VLD-PROC field in the logonid record.

RECONNECT|NORECONNECT
Specifies that the user wants to reestablish a TSO session that was disconnected (RECONNECT) or that the user does not want a session reestablished if it has been disconnected (NORECONNECT). If neither parameter is specified, eTrust CA-ACF2 reconnects to any pre-existing session or starts a new session and displays the full-screen logon panel.

RECOVER|NORECOVER
Indicates whether the user wants the TSO RECOVER option on for this TSO session.

SIZE(*regionsize*)
Specifies the desired region size in K for this TSO session up to 2,097,152K. The SIZE value is limited and cannot exceed the size specified in the TSOSIZE field of this logonid record unless it also has the LGN-SIZE privilege. **Note:** If a SIZE(0) is specified, LGN-SIZE must be in the LIDREC.

TIME(*sessiontime*)
Specifies the desired CPU time for this TSO session. User must have LGN-TIME to specify TIME.

UNIT(unitname)
Specifies the desired default generic unit name for this TSO session and is one to eight characters long. The logonid must have the LGN-UNIT attribute to specify UNIT.

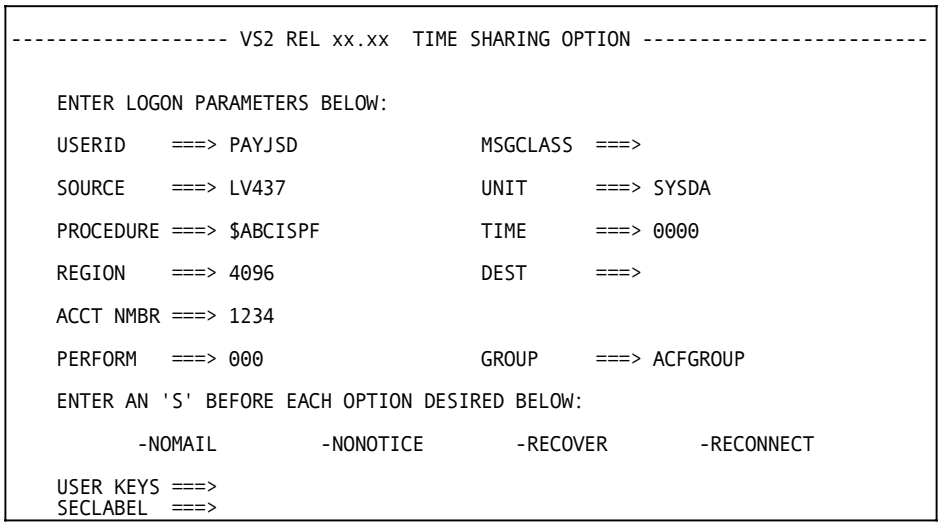
TSO Full-screen Logon Procedure

eTrust CA-ACF2 provides full-screen logon support for authorized TSO users. This section describes the following full-screen features:

- The logon panel
- Default logon fields
- Bypassing full-screen logon
- Support for hard-copy devices

The Logon Panel

If you have full-screen privileges in your logonid record, the logon panel displays after eTrust CA-ACF2 validates your logonid and password.



```
------------------- VS2 REL xx.xx  TIME SHARING OPTION -------------------------

   ENTER LOGON PARAMETERS BELOW:

   USERID    ===> PAYJSD                MSGCLASS  ===>

   SOURCE    ===> LV437                 UNIT      ===> SYSDA

   PROCEDURE ===> $ABCISPF              TIME      ===> 0000

   REGION    ===> 4096                  DEST      ===>

   ACCT NMBR ===> 1234

   PERFORM   ===> 000                   GROUP     ===> ACFGROUP

   ENTER AN 'S' BEFORE EACH OPTION DESIRED BELOW:

          -NOMAIL         -NONOTICE        -RECOVER        -RECONNECT

   USER KEYS ===>
   SECLABEL  ===>
```

Logon Panel Fields

Your site can display the following fields. You can save these fields from session to session except where noted.

USER ID
Specifies the user's logonid. You **cannot** change this value.

MSGCLASS
Specifies the one-character TSO message class for this user.

SOURCE
Specifies the physical or logical name of the input device you are at. You **cannot** change this value.

UNIT
Specifies the one to eight-character TSO unit name for this user.

PROCEDURE
Specifies the one to eight-character name of the procedure that contains the JCL for initiating the TSO session.

TIME
Specifies the maximum CPU time permitted for the user's session. Specify this value as *mmmm* (minutes); 1440 means unlimited time.

REGION
Specifies the 0 to 2,097,152K TSO region size for the user's session.

DEST
Specifies the default TSO remote destination for this user.

ACCT NMBR
Specifies the required one to 40-character TSO account number.

PERFORM
Specifies the TSO performance group to be used during the session. Enter a value from one to 250.

GROUP
Initially displays the value entered in the GROUP(*grpname*) logon parm if specified. Users can specify another valid group in this field that is used only for this session. If no GROUP is specified by the user at system entry, or if this field is blank on the full-screen display, eTrust CA-ACF2 uses the default value from the GROUP logonid field without redisplaying the screen.

USER KEYS
Accommodates special keywords required by the site for logon and displays at the bottom of the screen.

SECLABEL
Specifies the user's one- to eight-character security label. MLS must be active on the system before a value may be specified in this field. If you specify a security label, it must be valid, and you must be authorized to use it, otherwise, you will be prompted until you either specify a valid security label or specify no value in the field (blanks). For information on how to implement MLS on a system, see the *Multilevel Security Planning Guide*.

Also on the logon screen, you can enter S before each of the following operands that you want to have in effect.

NOMAIL
Suppresses display of system mail at logon time.

NONOTICE
Suppresses display of TSO notices at logon time.

RECOVER
Creates a work file during your editing session that you can use for recovering edits made to a data set in the event of a disconnect or system failure.

RECONNECT
Lets you reestablish an existing session after your line has been disconnected. This logon must occur in a reconnect time limit after your line has been disconnected. When logging on again, you must specify the same logonid and password as you used previously for beginning the interrupted session; operand values from the interrupted session remain in effect and cannot be changed. You **cannot** save this value from session to session.

User or site-wide options for TSO full-screen logon and these LOGON fields are described in Implementing TSO Full-Screen Logon Support in the "Special eTrust CA-ACF2 Procedures" chapter. For information about the C-TSO full screen retention records, see TSO Full-Screen Logon Retention Records in the chapter entitled, "Maintaining Global System Options Records." Information about the various global system options that control TSO is also in that chapter.

Bypassing Full-screen Logon

A TSO user with the TSOFSCRN privilege can bypass display of the logon screen. During logon, enter the TSO logon command and specify your logonid and the NFSCREEN keyword, as shown in the following:

```
logon user01 nfscreen
```

After eTrust CA-ACF2 validates the logonid and password, you can proceed as if no full-screen authorization exists.

Support for Hard-Copy Devices

The full-screen display is limited to IBM 3270-type display terminals. If you are at a hard-copy terminal and are authorized for full-screen logon, you might see a message similar to the following one printed at the terminal at logon time:

```
ACF82022 ACF2, THE FOLLOWING KEYWORDS ARE IN EFFECT:
logon USER01/USER01  ACCT(4) PROC($PRDISPF) SIZE(01024) UNIT(SYSDA)
ACF82021 ACF2, ENTER OVERRIDES OR HIT ENTER TO CONTINUE
```

You can do one of the following:

- Enter any of the listed operands to change the values in effect. For example, you can change the value of the SIZE operand to 8192K by entering,

  ```
  size(8192)
  ```

  eTrust CA-ACF2 repeats the logon message but lists any new values.

- Enter operands not already listed to place additional values into effect. For example, you can put the NOMAIL operand into effect by entering,

  ```
  nomail
  ```

  eTrust CA-ACF2 repeats the logon message, listing the newly specified operand and any value it might have.

- Retain the values in effect by pressing the ENTER or the RETURN key. You also press the ENTER or the RETURN key after changing or adding operands and values as desired. eTrust CA-ACF2 continues validating your logon request. If the validation is successful, normal TSO logon occurs.

To perform the previously described actions in the first two alternatives, you must have permission to specify the operand that you want to change at logon time.

CICS Sign-on

We recommend that you require individual CICS users to sign on to the CICS region. However, CICS sign-on is optional with the eTrust CA-ACF2 CICS interface. If a terminal runs under CICS but a sign-on has not been performed, eTrust CA-ACF2 uses the site-defined default logonid for validating terminal access requests.

You can sign on to CICS using any of the formats listed in the following. You can substitute spaces for commas.

```
CESN

CESN logonid/password/new-password

CESN LID=logonid,PW=password,NPW=newpassword,G=groupname

CESN NAME=logonid,PS=password/newpassword,GROUP=groupname
```

The variables for these formats are described in the following:

logonid
The user's eTrust CA-ACF2 logonid.

password
The user's current eTrust CA-ACF2 password.

newpassword
The new password to replace the current password, provided the site permits password changes.

groupname
The optional group name that authorizes access for this user based on a particular group.

Preferably, you should enter only your logonid with the sign-on transaction ID so that eTrust CA-ACF2 prompts for your password. This procedure lets you enter the password separately in a nondisplay area. eTrust CA-ACF2 also provides a sign-on screen with a nondisplay area. To access the sign-on screen, enter any of the one-word sign-on transactions (for example, CESN) without entering any other data.

```
CICS/ESA - eTrust CA-ACF2 (SYSTEM SIGN-ON/SIGN-OFF FACILITY)

SYSTEM:   CICS     -- CICS/ESA SYSTEM --
TERMINAL:  L43E
NODE:    LV43E

DAY:    Tuesday

SYSTEM DATE:  July 28, 2001
SYSTEM TIME:  02:03 PM

LOGONID: ===>       GROUP ===>
PASSWORD: ===>

NEW PASSWORD ===>  (protected nondisplay area)
       ===>
```

eTrust CA-ACF2 validates the logonid and password first. Before you proceed, it also validates whether you can access CICS from that source at that time and whether you have the authority to access the system as a member of a group.

To sign off from CICS, enter the standard CESF or a site-defined alternative. The site can also modify the CICS sign-on facility to accommodate unique site requirements. For further information about CICS sign-on, see the *CICS Support Guide*.

IMS Sign-on

IMS sign-on is optional if your site installs the eTrust CA-ACF2 IMS interface. eTrust CA-ACF2 uses a default logonid for transaction authorization if the terminal user has not signed on. The IMS system itself can specify which terminals are required to sign-on for a user to enter transactions

You can use one of the following formats to sign on to IMS:

`/SIGN ON` *`logonid password`*`{/`*`newpassword`*`}` `{GROUP` *`groupname`*`}`

`/SIGN ON logonid password {NEWPW newpassword} {GROUP groupname}`

The variables for these formats are described in the following:

logonid
The user's eTrust CA-ACF2 logonid.

password
The user's current eTrust CA-ACF2 password.

newpassword
The new password to replace the current one, provided that the site permits password changes.

groupname
The optional group name that authorizes access for this user based on a particular group.

Your site can use preformatted screens that require the data to be entered in the designated fields.

For further information about IMS sign-on, see the *IMS Support Guide*.

Submitting Batch Jobs

Your site can direct eTrust CA-ACF2 to check that the logonid that submits the job has the authority to submit batch jobs. To do this, you must specify the JOB field in the logonid record and specify the JOBCK option of the GSO OPTS record. If you want eTrust CA-ACF2 to check the authority of logonids that submit jobs from other nodes, you must also specify the JOBCK option of the GSO NJE record. For details on the GSO OPTS record, see eTrust CA-ACF2 Option Specifications (OPTS) in the "Maintaining Global System Options Records" chapter.

Jobs Submitted by TSO Users

If you submit a job with the SUBMIT command, you do not need to make any changes to the JCL you use to submit the job. eTrust CA-ACF2 automatically transfers your logonid and input source information to the submitted job. However, if you submit a job for another logonid, you must specify the //*LOGONID and //*PASSWORD statement for that logonid.

Place the following control statements in your JCL after the JOB statement:

```
//*LOGONID logonid
//*PASSWORD your-password
```

eTrust CA-ACF2 always validates the first //*LOGONID and //*PASSWORD statements in a batch job stream. If eTrust CA-ACF2 finds multiple entries, it validates the first occurrence of each statement. It treats subsequent //*LOGONID and //*PASSWORD statements as comments. If USER= and //*LOGONID statements appear in the same job, USER= overrides the //*LOGONID statement.

To change your password, add the new password to the JCL, as follows:

```
//*PASSWORD password/newpassword
```

You can also specify your logonid, password, GROUP, or a SECLABEL logon parameter with JCL JOB statement parameters, as follows:

```
USER=logonid,PASSWORD=password,GROUP=groupname,SECLABEL=seclabel
```

For information on how to implement MLS on a system, including assigning security labels to jobs, see the *Multilevel Security Planning Guide*.

Similarly, you can change your password through the JOB statement parameters, as follows

```
USER=logonid,PASSWORD=(password,newpassword)
```

For compatibility with non-eTrust CA-ACF2 systems, specify logonids up to seven characters in the USER= JOB statement parameter.

Jobs that Submit Other Jobs

When a job submitted by a logonid submits other jobs, these jobs inherit the logonid of the submitter of the first job. Two utilities help you manage how jobs get submitted: ACFSUB and JOBCOPY.

- ACFSUB is a TSO command issued from TSO READY mode that lets you run production and other jobs under a logonid other than that of the TSO operator. eTrust CA-ACF2 verifies that the TSO operator has the ability to submit jobs through ACFSUB out of a referenced JCL library. ACFSUB creates the logonid that the jobs run under.
- JOBCOPY is a batch utility or started task that lets you run production and other jobs under a logonid other than that of the TSO operator. eTrust CA-ACF2 verifies that the user submitting the job has the authority to submit jobs through JOBCOPY out of a referenced JCL library. JOBCOPY creates the logonid that the jobs run under.

Note: For more information about these utilities, see the "Other Utilities" chapter in the *Reports and Utilities Guide*.

Restricting Production Jobs

You should specify the RESTRICT privilege in the logonid record of production jobs. Then, if a production job submits other jobs, these jobs run under a production logonid with no logonid or password required in the submitted JCL.

eTrust CA-ACF2 logs all jobs submitted by restricted logonids in the ACFRPTJL report. To monitor jobs submitted by jobs that run under a restricted logonid, specify the //*LOGONID or USER= job statements in the JCL. Specify the restricted logonid. Including one of these job statements lets eTrust CA-ACF2 perform a second validation for the restricted logonid and create a logging for the spun job in the ACFRPTJL report.

Started Tasks

You should assign started tasks a logonid and give them the STC privilege. By default, the logonid used is the same as the procedure name of the started task. If you want a different logonid assigned to the started task, you can add the JOBNAME parameter to the START command specifying the desired logonid. If there are no logonid matches for the started task procedure name/jobname, a search of the optional GSO STC started tasks table is performed. Each entry in the GSO STC started task table contains an explicit or masked STC procedure name, the logonid name to be associated with the started task, and an optional group name. If a match is found in the GSO STC started task table, the logonid associated with the match record is assigned. Otherwise, the default started task logonid, from the GSO OPTS record is used. eTrust CA-ACF2 denies access to STCs that do not have this privilege if the GSO OPTS record specifies STC for started task validation. Logonids with the STC privilege cannot log on to TSO, but can submit batch jobs, as long as those batch jobs have a logonid specified that doesn't have the STC attribute. A batch job cannot inherit an STC logonid. Do not specify the RESTRICT privilege for STCs. No source checking is done for started task logonids during system entry validation. You can monitor the use of logonids with the STC privilege by running the ACFRTPLL report with the UPDATE parameter.

Certain started tasks represent important system components that are required for z/OS to run properly. The OMVS started task is an example of such an address space. To prevent the disruption of service, you can give the logonid for the OMVS started task NON-CNCL. If you want the started task subject to the validation process, but not to cancellation for maximum violations (MAXVIO), you can give the logonid the NOMAXVIO attribute instead of NON-CNCL. NOMAXVIO allows normal validation to take place, but prevents eTrust CA-ACF2 from canceling the started task when the MAXVIO count is reached.

Specifying a Group or Project Name at Logon

eTrust CA-ACF2 provides the GROUP(*grpname*) logon parameter to let you access the system based on a group or project name. If you want to associate yourself with a group to access its data, enter the group name at logon. If a resource rule lets you use the group name, eTrust CA-ACF2 changes your UID and access privileges based on the privileges for the group. You can also specify the GROUP= statement in JCL for a batch job or you can enter a group name in the GROUP field on the TSO logon full-screen panel.

If you do not specify GROUP at system entry, eTrust CA-ACF2 uses the value specified in the GROUP field of the user's logonid record for validation. Only a logonid with the authority to create or change a logonid record (such as SECURITY or ACCOUNT) can define the GROUP field. This value is the default value for the logonid. A user can associate with another group by specifying that group name at logon. However, any change the user makes is active for the session only. When he logs on again, he must specify that group name again if it is not his default group name.

Implementing the GROUP Logon Parameter

One way to implement access based on groups or projects is to write access and resource rules so that validation is largely based on the GROUP field in the UID. Then users can access data available to the group and you do not need to change access or resource rules.

For example, auditors that belong to the AUDT group require access to accounting data that can be accessed only by members of the ACCT group. If the auditors are authorized to use the ACCT group name at system entry, eTrust CA-ACF2 grants them system access as members of the ACCT group. If you define the GROUP field in the @UID macro, the auditors can access any data that can be accessed by ACCT group members without requiring additional or changed access or resource rules.

Since eTrust CA-ACF2 creates the UID for users at each system entry, you can see that this feature affects data set and resource access. If your site's @UID macro contains the GROUP logonid field, a user could have a different UID each time he or she logs on. Although the values for company, division, and site cannot change, the value for group does. This is why you must secure the GROUP logonid field with resource rules.

Note: If your site has defined an @CFDE entry called GROUP, you must change its external name so that it does not conflict with the eTrust CA-ACF2 GROUP @CFDE entry. Otherwise, assembly of the eTrust CA-ACF2 Field Definition Record (ACFFDR) fails during installation. Furthermore, if your UID concatenation contains a reference to your GROUP @CFDE entry, you must also change the name in the @UID macro of the ACFFDR.

If your site's UID concatenation does not contain the GROUP logonid field and you do not use this value in user exit processing, this logonid parameter has no significance or effect on data access; however, it is still used with z/OS Unix System Services (OMVS). Perform the following steps to use the GROUP logon parameter to alter the UID at logon or to use exit processing:

- Decide how to process the group name:
  - Modify the @UID macro in the ACFFDR to include the GROUP logonid field. See Appendix A, "eTrust CA-ACF2 Field Definition Record Generation," in the *Getting Started* guide.
  - Process the group name in user exits as necessary. To reference the GROUP logon parameter in logon exits, eTrust CA-ACF2 stores the group name value in an extension of the ACVALD parameter list. To reference the group name value in data set or resource exits, the GROUP logonid field stores the group name. See the *Systems Programmer Guide* for detailed information about exit parameter lists.
- Create resource rules for group names.
- Enter the GROUP logon parameter at system entry, with the logonid and password.

Allowing Use of the Group Parameter

eTrust CA-ACF2 only validates the use of the GROUP parameter if the user specifies a group that is not the default specified in his logonid record. When a user specifies a different group at logon, eTrust CA-ACF2 validates group names by checking resource rules at logon. The $KEY of the rule identifies the one to eight-character group name. This field is maskable, but you must create an entry for the TGR type code in the GSO RESDIR or INFODIR record and rebuild the resident directory. For details, see the "Global System Options Records" chapter. The following is an example of a resource rule that grants system entry to users in the payroll, finance, and audit departments as members of the ACCT group.

```
$KEY(acct) TYPE(TGR)
 UID(******pay-) ALLOW
 UID(******fin-) ALLOW
 UID(******aud-) ALLOW
```

Specifying Multi-Value Logonid Fields and UID Strings

CA-ACF2 supports multi-value logonid fields and multi-value UID strings. This feature affords sites the potential for even greater control over resources. When the UID string contains a multi-value field, validation processing detects and processes each value when necessary. Only the first multi-value field defined as part of the UID string is used during validation processing. Any additional multi-value fields within the UID string use the first value. A multi-value field must be defined in the UID string as a full field; you cannot use a partial multi-value field as part of the UID string.

To implement this feature, perform these tasks:

- Define a multi-value logonid field
- Convert the UID string
- Review USERMOD validation routine
- Activate the multi-value logonid field and UID string
- Add a resource rule for the validation routine
- Run the ACFRPTSL report

The following sections describe these tasks.

Define a Multi-Value Logonid Field

Multi-value fields are defined in the USERLID or USERXLID portion of the LIDREC. The number of entries is limited by the number of available bytes within the USERLID or USERXLID LIDREC area. The maximum number of available bytes within each of these areas is 192 bytes. A multi-value field requires some overhead in addition to the actual number of bytes needed for all occurrences of the multi-value field defined. The length of each multi-value entry, the overhead of a multi-value, and the available bytes in the user portion of the LIDREC limit the number of values you can define.

When you add a new user-defined field to the LIDREC, you must add a @CFDE entry to the USERCFDE portion of the ACFFDR. The @CFDE entry defines the external field name and its related internal characteristics and attributes for the new field. These characteristics include the type of field, who can modify or list the field, flag settings (multi-value), the maximum number of entries, the display group to which the field belongs, and any validation routine to be used when adding, changing, or deleting values from this field.

You should not modify default CA-ACF2 @CFDE entries to point to new multi-value fields. For example, GROUP should not point to a multi-value field in the UID string because most LIDREC fields are pointed to directly. Modification of the GROUP at logon would not update the multi-value field, but the LIDGROUP LIDREC field. Multi-value fields should only point to user-defined USERLID fields.

A supplied USERMOD validation routine verifies the user modifying the multi-value field. This validation is based on resource rules. Set the ALTER parameter to ALL to allow the validation routine to be used. If the ALTER parameter is set to some level of security, a security check is performed prior to the validation routine and the call may not be performed.

See the *Systems Programmer Guide* for details on all available parameters for an @CFDE entry.

The following example shows how to add the new field to the LID record in the USERLID. Since this is a multi-value field, a 12-byte header section is added at the beginning of the values defined. The MAX parameter determines the number of values in the defined field. In this example, there are ten values kept in the LIDMLTFL field. The total number of bytes comprising the entire field is 92 bytes:

(10 entries x 8 length) + 12 (header) = 92

```
*****************************************************************
*                                 *
* USERLID - THIS SOURCE MATERIAL IS COPIED INTO THE USER    *
*      DEFINITION SECTION OF THE LOGONID RECORD 'DSECT'. *
*      THE INSTALLATION MAY REPLACE THIS MODULE OR     *
*      EDIT ITEMS WHICH ARE TO BE DEFINED BY @CFDE     *
*      MACRO ENTRIES IN THE ACFFDR. THE LENGTH ATTRIBUTE *
*      OF EACH SYMBOL DEFINED HERE IS USED IN THE RELATED *
*      @CFDE MACRO EXPANSION.              *
*                                 *
* NOTE -- THE TOTAL LENGTH OF ALL INSTALLATION ADDED SYMBOLS  *
*     CAN NOT EXCEED 192(DECIMAL), C0(HEX) BYTES.     *
*                                 *
*****************************************************************

 LIDMULTF DS  0XL(LIDMLTLN)       FOR MINILID DEFINITION
 LIDMLTFL AMULTFLD LENGTH=8,MAX=10,TYPE=GEN
 LIDMLTLN EQU  *-LIDMLTFL

 ************* END OF USERLID *******************************
```

The following example illustrates an @CFDE that defines the new multi-value field to the USERCFDE. The MVMAX value must be the same as the MAX value specified in the AMULTFLD macro.

```
@CFDE MLTFLD,LIDMLTFL,CHAR,
       ALTER=ALL,LIST=ALL,
       FLAGS=NULL+MULTIVAL,MVMAX=10,
       GROUP=0,ZERO=YES,VRTN1=ACF00SFP
```

The following examples illustrate the use of a multi-value logonid field in a logonid:

1. The first example indicates how to place value in the multi-value field using an insert command:

```
Insert testlid name(multivalue field) mltfld(entry03 entry04 entry02)
    TESTLID        TESTLID MULTIVALUE FIELD
                   MLTFLD(ENTRY03 ENTERY04 ENTRY02)
    ACCESS         ACC-CNT(1) ACC-DATE(11/29/00) ACC-TIME(14:41)
    PASSWORD       PSWD-DAT(00/00/00) PSWD-INV(0) PSWD-TOD(00/00/00-00:00)
                   PSWD-VIO(0)
    TSO            DFT-PFX(TESTLID)
    STATISTICS SEC-VIO(0) UPD-TOD(11/29/00-14:43)
    RESTRICTIONS   PREFIX(TESTLID)
```

2. The second example illustrates how to delete a specific entry from a multi-value field of a logonid record.

```
ch testlid mltfld(entry02) DEL
    TESTLID        TESTLID MULTIVALUE FIELD
                   MLTFLD(ENTRY03 ENTRY04)
    ACCESS         ACC-CNT(1) ACC-DAT(11/29/00) ACC-TIME(14:41)
    PASSWORD       PSWD-DAT(00/00/00) PSWD-INV(0) PSWD-TOD(00/00/00-00:00)
                   PSWD-VIO(0)
    TSO            DFT-PFX(TESTLID)
    STATISTICS     SEC-VIO(0) UPD-TOD(11/29/00-14:44)
    RESTRICTIONS   PREFIX(TESTLID)
```

3. The third example illustrates what happens when you add a value to an existing multi-value field. Notice that the value is appended to the existing list of values in the field.

```
ch testlid mltfld(entry01)
    TESTLID          TESTLID MULTIVALUE FIELD
                     MLTFLD(ENTRY03 ENTRY04 ENTRY01)
    ACCESS           ACC-CNT(1) ACC-DATE(11/29/00) ACC-TIME(14:41)
    PASSWORD         PSWD-DAT(00/00/00) PSWD-INV(0) PSWD-TOD(00/00/00-00:00)
                     PSWD-VIO(0)
    TSO              DFT-PFX(TESTLID)
    STATISTICS       SEC-VIO(0) UPD-TD(11/29/00-14:44)
    RESTRICTIONS     PREFIX(TESTLID)
```

4. The final example illustrates how to specify the values in the multi-value field to appear in a specific order. You obtain these results by changing the field and specifying the values in the desired order, and by specifying the REP (replace) operand.

```
ch testlid mltfld(entry01 entry03 entry04) rep
    TESTLID          TESTLID MULTIVALUE FIELD
                     MLTFLD(ENTRY01 ENTRY02 ENTRY03 ENTRY04)
    ACCESS           ACC-CNT(1) ACC-DATE(11/29/00) ACC-TIME(14:41)
    PASSWORD         PSWD=DAT(00/00/00) PSWD-INV(0) PSWD-TOD(00/00/00-00:00)
                     PSWD-VIO(0)
    TSO              DEFT-PFX(TESTLID)
    STATISTICS       SEC-VIO(0) UPD-TOD(11/29/00-14:45)
    RESTRICTIONS     PREFIX(TESTLID)
```

Note: Adding a multi-valued field to an existing database, which has had other field definition changes, may result in an ACF00114 ERROR IN LOCATING/PROCESSING FIELD ff – ACF00FLC RETURN/REASON error message. This would be caused by residual data in the LIDREC. That residual data can be cleared out in several ways. One way would be to create a dummy record equal in length to the proposed multi-valued field prior to defining the multi-valued field. Once ACF2 has been started, issue a mass change to set the dummy field to null values. Then eliminate the dummy field and define the new multi-valued field.

Convert the UID String

If you plan to use the multi-value field within the UID string, a conversion may be necessary. Since UID strings are limited to 24 characters, you may need to convert the UID string to allow for the multi-value field. If enough room exists for the multi-value field, add the field to the UID string. The first occurrence of the multi-value field is used in the default UID string built at logon time. If there is not enough room within the current UID string for one value of the multi-value field, convert the UID structure to accommodate it. A multi-value field must be defined in the UID string as a full field; you cannot use a partial multi-value field as part of the UID string.

Here's how to add a multi-value field to the UID definition in the ACFFDR.

```
************************************************************
*
*  THE UID ENTRY DEFINES THE USER IDENTIFICATION STRING.
*
************************************************************
    @UID  COMPANY,SITE,LEVEL,PROJECT,LID,MLTFLD
```

When the UID is built at logon time, the UID string contains the default multi-value field (the first value). Processing does exist, however, to detect the multi-value field and to process all values of that field when necessary. The order of the values defined for any one user is very important. Since the validation process uses multi-value field values in the order defined, place the most used value as the first entry followed by the next most used, and so on. The UID entry within the ACFFDR must reflect the change of the UID string (defined in the @UID macro). Whenever a field is added to the UID string, it must be added to all minilid structures and @MLID definitions in the ACFFDR.

See the *Implementation Planning Guide* for details on the UID string.

With minilids, more than just the AMULTFLD must be defined. Labels defined in the USERLID are used to satisfy the minilid requirements. This example shows how to add the multi-value field for a minilid IMS definition by modifying the MLAIMS macro and the @MLID definition in the ACFFDR.

```
*****************************************************************
*
*  MLAIMS- IMS MINI LID SECTION DEFINITION
*
*  THE IMS MINI LID DEFINITION CONTAINS FIELDS THAT ARE
*  REQUIRED BY ALL ACF2/IMS SUPPORT INTERFACES FOR VALIDATION
*  OF USER/SYSTEM REQUESTS.
*
*  NOTE: NO PADDING IS ALLOWED (I.E. NO UNUSED SPACE OR
*     RESERVED ARES).
*
*****************************************************************
        MACRO
        MLAIMS  &DSECT=YES
MLAIMS  ACDEF &DSECT
MLAIIDLE DS  XL1          MAX IDLE TIME IN MINUTES
MLAISAUT DS  XL1          IMS AUTHORIZATION BYTE
MLAIMSSF DS  XL(LIDMLTLN)      MULTI-VALUE FIELD
MLAIMSL EQU  *-MLAIMS       LENGTH OF IMS MINI LID
       MEND


*****************************************************************
*
*    SPECIFY THE IMS @MLID DEFINITION
*
*****************************************************************

    @MLID IMS,MLAIMS,MLAIMSL,   NAME, START, LENGTH
        (LIDIDLE,MLAIIDLE),   MAX IDLE TIME IN MINUTES
        (LIDMULTF,MLAIMSSF),  MULTI-VALUE FIELD
        (LIDM2FLG,MLAISAUT)   IMS SIGNON AUTH BYTE
```

Review USERMOD Validation Routine

The supplied validation routine ACF00SFP, defined in the @CFDE entry for the multi-value field, is part of the multi-value logonid field support. This routine is called when a replace, add or change is performed on this field. Its purpose is to validate that the user making the change is authorized to change that value. A user must be an unscoped security officer to use the REPLACE feature. A validation call is performed for an add or delete of any value. The resource type is SFP and the resource name is the value of the LIDREC field. Resource records with type R(SFP) must be written to allow users with less than the SECURITY privilege to update the field.

This routine issues two separate error messages when rejecting a field update:

```
ACF00103 - NOT AUTHORIZED TO CHANGE FIELD fff
ACF00109 - INVALID OPERATION REQUESTED FOR FIELD fff
```

Any part of this validation routine can be changed, whether it is the resource type, the error messages, or the processing in particular.

You are not required to use the validation routine. You can control the multi-value field by using the ALTER parameter in the defined @CFDE or by writing a different validation routine.

Review this validation routine and make any changes through SMP/E ++SRCUPD, which assembles ACF00SFP and relinks the ACFFDR.

Activate the Multi-Value Logonid Field and UID String

When all the above changes are complete, IPL the system and restart CA-ACF2. This refreshes all module updates including the ACFFDR with the multi-value LID field and the updated UID string.

After the IPL is complete, an authorized administrator can issue the following command to initialize all logonid records to have no values within the multi-value field:

```
CH LIKE(-) multi-value field()
```

To insert values within the multi-value field, an authorized administrator can issue this command:

```
CH userid multi-value field(value1,value2,...)
```

Review Processing of the Multi-Value UID String

Access and resource validations use the default UID string containing the first occurrence of the multi-value field. If a violation occurs, a temporary UID string with the next occurrence of the multi-value field is built and revalidated. This process continues until access is allowed or all occurrences of the multi-value field are exhausted.

If access is allowed, loggings are reported on the active UID string. If a violation occurs after all values are validated, the violation is reported against the default (first value).

Add a Resource Rule for the Validation Routine

Resource rules control who can update the multi-value field. The supplied validation routine uses a resource type of R(SFP) and is called each time the multi-value field is changed on a logonid record. The $KEY value on the resource rule is the value of the field.

Be sure to add the resource type (R-RSFP) to the GSO INFODIR record. This allows these resource records to be placed into a directory, which accelerates the validation process.

Examples of type R(SFP) resource rules follow. The $KEY value in these rules is a valid value that may be used in the LIDREC multi-value field.

```
$KEY(VALUE01) TYPE(SFP)
UID(*****USER001) ALLOW
UID(*****USER002) ALLOW

$KEY(VALUE02) TYPE(SFP)
UID(*****USER003) ALLOW

$KEY(VALUE03) TYPE(SFP)
UID(*) ALLOW
```

The following example shows the resource type in the GSO INFODIR record.

```
SYSID / INFODIR TYPES(R-RSFP)
```

Run the ACFRPTSL Report

The ACFRPTSL report records multi-value field information. Use this report utility to determine which users are assigned certain values. Use the IF parameter to extract a particular value. The REPORT parameter determines the type of output. REPORT(FULL) displays the full logonids of all users and REPORT(SHORT) displays only the fields indicated by the SFLDS parameter. Sample JCL to run the ACFRPTSL report follows:

```
//RPTSL   JOB
//ACFRPTSL EXEC PGM=ACFRPTSL
//SYSPRINT DD  SYSOUT=*
//SYSIN   DD  *
TITLE(MULTI-VALUE FIELD: VALUE01/VALUE02)
REPORT(SHORT)
INPUT(ACF2)
IF(MLTFLD = 'VALUE01' OR MLTFLD = 'VALUE02')
SFLDS(MLTFLD)
//
```

When the criteria are met, the report shows all values of the multi-value field for each user.

```
ACF2 UTILITY LIBRARY - ACFRPTSL - LOGONID SUPERLIST REPORT - PAGE 1
DATE mm/dd/yy (yy.ddd) TIME hh.mm MULTI-VALUE FIELD: VALUE01/VALUE02
 LOGONID NAME         DATE  TIME  CHANGER MULTFLD
 ------- ----         --------------     -------
 USER001 USER NAME 1     yy/dd/yy-hh:mm     VALUE01
 USER002 USER NAME 2     yy/dd/yy-hh:mm     VALUE02
 USER003 USER NAME 3     yy/dd/yy-hh:mm     VALUE02
                            VALUE05
                            VALUE07
 USER004 USER NAME 4     yy/dd/yy-hh:mm     VALUE01
                            VALUE03
```

Providing Extended User Authentication

You can incorporate any user authentication device or routine with the standard eTrust CA-ACF2 CICS and TSO logon support. The eTrust CA-ACF2 user authentication feature expands system entry validation so that it is made up of the usual eTrust CA-ACF2 logon controls (for example, password, source, shift) plus some additional user authentication. You can use up to eight different extended validation routines.

You can perform the additional user authentication using a physical device or software routine. Using a device with eTrust CA-ACF2 might require users to enter data into both the device and the terminal keyboard. Some devices that perform this processing input the user's unique information directly to the terminal, such as operator identification (OID) card readers. (OID card support is provided as a standard feature of eTrust CA-ACF2. See Appendix A, "Operator Identification Card Support," for details.) Others require the user to enter data into the device. Then the device interprets the information and displays data for the user to then enter into the terminal for validation by a software routine provided by the makers of the device. In some situations, software routines are used in place of a device. We provide an example of how a routine might be used for extended authentication in Note 9 in the " Notes" appendix of the *Systems Programmer Guide*.

The normal logon process signals the extended authentication requirement by way of a field in the user's logonid record. Processing options for another vendor's device or software routines are identified in a GSO record in the eTrust CA-ACF2 Infostorage database. It coordinates communication between the user and the authentication routine in a dialog. For example, the routine prompts the user for information, the user responds, the routine processes the user's response and then might prompt again, and so on.

eTrust CA-ACF2 lets the dialog continue until the vendor device (or software routine) makes a recommendation as to whether the user should be given access to the system. Whether allows or prevents access depends on the user authentication routine. It reacts only to this recommendation. Denied system accesses are reported in the Invalid Password/Authority Log (the ACFRPTPW report).

During extended authentication processing, the user is required to provide some unique information. The information can change every time a user attempts sign-on. This information is generally made up of a user-challenge algorithm, user-unique keys, and possibly some vendor control data.

You can store this user-provided data in the eTrust CA-ACF2 Infostorage database. Using this option lets you further centralize the security and control of a data center. You can then use eTrust CA-ACF2 to regulate who and how the information is maintained. Changes to the data can automatically be included in the standard eTrust CA-ACF2 reports and file backups. More information is contained in the Providing Extended User Authentication Support section of the "Special Usage Considerations" chapter in the *Systems Programmer Guide*.

Choosing the UADS or NOUADS Option

Deciding whether to continue to use the user attribute data set (UADS) after you install eTrust CA-ACF2 support is an important decision you must make. When you opt to continue UADS validation after eTrust CA-ACF2 is installed, it validates only the logonid, password, and group supplied during logon. Other logon parameters, such as ACCT and PROC and the resulting session attributes, are controlled by normal TSO processing. To specify your choice, you must set the UADS|NOUADS option of the GSO OPTS record. For details on the GSO OPTS record, see eTrust CA-ACF2 Option Specifications (OPTS) in the "Maintaining Global System Options Records" chapter.

Selecting UADS

If you want to continue to use the SYS1.UADS data set, you must specify the UADS option in the GSO OPTS record. You must also define each TSO user in the eTrust CA-ACF2 Logonid database and in the SYS1.UADS data set. This creates more administration for you. Also, it cannot support the multiple password tree structure permitted using UADS alone. eTrust CA-ACF2 only supports one password per user. This means it supports only the first UADS password for a user while it is active.

Selecting NOUADS

We recommend that you use the eTrust CA-ACF2 default setting, NOUADS. Since this is the default setting, you do not need to alter the GSO OPTS record. You also save on administration because you no longer have to maintain the SYS1.UADS data set. With NOUADS, eTrust CA-ACF2 controls the parameters entered during logon and the resulting TSO session attributes. The TSO logon parameters and full-screen feature described earlier in this chapter are also available when you select NOUADS.

Providing Dynamic Logonid Privileges

The logonid record contains privilege fields that affect whether the logonid can access the system and what it can do after it gains access. These privileges are usually assigned by specifying the corresponding privilege field directly in the logonid record. See the "Maintaining Logonid Records" chapter.

Logonid privileges can also be dynamically assigned using a special set of eTrust CA-ACF2 resource rules. Using resource rules to control privileges lets you grant privileges based on any of the environment criteria in resource rules, such as SHIFT, SOURCE, and date controls.

You can control the privileges and authorities corresponding to any bit field in the logonid record (as defined in the ACFFDR @CFDE definitions) using these dynamic logonid privilege resource rules.

Implementing Dynamic Logonid Privileges

You activate resource-based logonid privilege controls by specifying a GSO CLASMAP record with a resource class of PRIVCTL. For details on CLASMAP records, see SAF Resource Classes (CLASMAP) in the "Maintaining Global System Options Records" chapter.

To create the CLASMAP record, enter:

```
SET CONTROL(GSO) SYSID(sysid)
INSERT CLASMAP.qualifier RESOURCE(PRIVCTL) RSRCTYPE(ttt)
```

sysid
The appropriate system ID for the GSO records.

qualifier
Any unique character string to uniquely qualify the CLASMAP record.

ttt
The three-character type code you chose for logonid privilege resource rules.

If eTrust CA-ACF2 finds and processes a CLASMAP record with a resource class of PRIVCTL during initialization, it creates a globally resident directory for the resource type and makes the privilege resource rules globally resident. This is done automatically. You do not have to specify the resource type in the GSO INFODIR record.

If you create the PRIVCTL CLASMAP record after eTrust CA-ACF2 initialization, you must activate the record using the F ACF2,REFRESH(CLASMAP) console command, and create the directory with the F ACF2,REBUILD(*ttt*) console command, where *ttt* is the privilege control resource type code.

Writing Logonid Privilege Resource Rules

The $KEY value of a privilege resource rule is the external field name of the bit field in the logonid record, for example SECURITY, NON-CNCL, or TSO. You cannot mask this value.

The keywords and parameters for these resource rules are the same as those for any resource rule. You can use SHIFT, SOURCE, and date controls.

The access decision in the resource rule lines is limited to ALLOW or PREVENT. Any decision other than ALLOW (such as ALLOW with LOG) is treated as a PREVENT. In addition, the access decision must be made in the initial resource rule. No NEXTKEY or resource group processing takes place for privilege resource rules.

The following sample privilege resource rule gives a user SMITH the NON-CNCL privilege through the end of the year.

```
$KEY(NON-CNCL) TYPE(TPV)
 UID(SMITH) UNTIL(12/31/01) ALLOW
```

For details on writing resource rules, see the "Maintaining Resource Rules" chapter.

Caution! *We recommend that you do not use dynamic privilege resource rules to give someone an unscoped SECURITY privilege. If you do, the privilege can be used during the period of availability to make permanent changes in the security environment and the user's own logonid record.*

Activating Dynamic Privileges for a Logonid

If dynamic logonid privileges are active (for example, a PRIVCTL CLASMAP record was processed), you should enable dynamic privilege controls for the logonid by specifying the PRIV-CTL field in the logonid record. The default value of the PRIV-CTL field is NOPRIV-CTL.

Note: The STC bit in the logonid is not eligible to be given to a user by use of PRIV-CTL.

For more information about the PRIV-CTL field and maintaining logonid records, see the "Maintaining Logonid Records" chapter.

System Entry Processing

When a logonid is used to access the system (system entry), eTrust CA-ACF2 checks to see if dynamic privilege controls are active in the system and if PRIV-CTL is set in the logonid record.

If privilege controls are not active or if NOPRIV-CTL is set in the logonid record, no dynamic privilege checking is performed for the user. The only privileges and authorities held by the logonid are the ones specified directly in the logonid record.

If privilege controls are active and PRIV-CTL is set in the logonid record, eTrust CA-ACF2 checks the privilege control resource rules to see what additional privileges and authorities the user has. These additional privileges are merged with the privileges granted directly in the logonid record, and the resulting set of privileges are held by the user for the duration of the access (the TSO session, the life of the batch job, and so on).

How Users Can Determine Unauthorized Use of Their Logonids

You can make users more responsible for checking unauthorized use of their logonids. To do this, specify the NOTIFY option of the GSO OPTS record. For details, see eTrust CA-ACF2 Option Specifications (OPTS) in the "Maintaining Global System Options Records" chapter. This option helps system users determine if someone has tried to use their logonids. It provides the following message after eTrust CA-ACF2 completes its logon processing:

```
ACF01137 your-logonid LAST SYSTEM ACCESS hh:mm-mm/dd/yy FROM source
```

If invalid password attempts have been made since the last successful logon, the user receives this message:

```
ACF01138 yourlogonid HAD nnnnn INVALID PASSWORDS SINCE LAST LOGON
```

This message warns a user that someone tried to sign on with his logonid by guessing the password. Users can determine the date, time, and source of the last invalid password attempt by displaying their logonid using the ACF LIST subcommand or the ISPF panels. They can then look for the PSWD-DAT, PSWD-TIM, and PSWD-SRC fields.

Chapter

3

Maintaining Logonid Records

The logonid record is the most important eTrust CA-ACF2 record. It identifies each user on a system protected by eTrust CA-ACF2. You define users and their privileges by specifying values for logonid record fields. These fields contain information about the logonid and password that a user must specify to enter the system. They also contain other information that is used by eTrust CA-ACF2 to validate the user's authority to the system.

This chapter describes the following topics:

- A quick look at a logonid record
- Logonid privileges and authorities
- Masking logonid records
- Who can maintain logonid records
- How to specify logonid record fields
- Managing passwords
- Logonid record fields
- Logonid record sections
- Logonid record fields and sections list
- Using the ISPF Panels
- Using the ACF Command
- Using User Profile Records

You can maintain logonid records using ISPF panels, ACF commands, and administrative utilities. The ISPF panels and the ACF commands are described at the end of this chapter. For information on the administrative utilities, see the *Reports and Utilities Guide*.

A Quick Look at a Logonid Record

Here is a sample logonid record for Jane Doe, an auditor in an accounting department. This sample logonid record is shown with extra blank lines to help define the various sections of the record:

```
USER01              ACCTGAUDUSER01        JANE DOE EXT.413
                    DEPT(ACCTG) FUNCTION(AUD)

CANCEL/SUSPEND      EXPIRE(02/02/06)

PRIVILEGES          AUDIT JOB TSO

ACCESS              ACC-CNT(133) ACC-DATE(08/15/04) ACC-SRCE(LV248)
                    ACC-TIME(09:21)

PASSWORD            PSWD-DAT(08/15/03) PSWD-TOD(07/28/04-13:23)
                    PSWD-VIO(1)

TSO                 DFT-PFX(USER01) DFT-SOUT(A) DFT-SUBM(A)
                    INTERCOM JCL LGN-SIZE LINE(ATTN) MAIL MSGID
                    NOTICES TSOPROC(IKJACCNT) TSORGN(1,024)
                    TSOSIZE(8,172) WTP

STATISTICS          SEC-VIO(1) UPD-TOD(08/11/04-09:21)

RESTRICTIONS        PREFIX(USER01)

DFP                 SMSINFO(DEFPROD)
```

Jane's logonid record illustrates the kind of information that you can specify in each section of the logonid record. See Logonid Record Sections later in this chapter for detailed information on these sections. Here is a description of the information in Jane's logonid record:

Identification

The user's name is Jane Doe, her logonid is USER01, and her phone number is extension 413. The entry ACCTGAUDUSER01 is her expanded UID. This site has defined the UID as the DEPT field, followed by the FUNCTION field, and followed by the logonid. The values ACCTG, AUD, and USER01 are taken from these fields to form the UID ACCTGAUDUSER01. The DEPT and FUNCTION fields have been defined by the site and do not appear in the logonid record supplied with eTrust CA-ACF2.

Cancel/Suspend

Jane's logonid record is temporary, because it expires on 02/02/06.

Privileges

Jane has the authority to list but not change eTrust CA-ACF2 rule sets, records, and system options (AUDIT); to run batch jobs (JOB); and to use TSO (TSO).

Access

Jane has made 133 system accesses. The last access was made at 09:21 on 08/15/04 from a terminal identified as LV248.

Password

Jane's last invalid password attempt was made on 08/15/03. The last time she changed her password was 07/28/04. On 08/15/03, she made only one invalid password attempt (PSWD-VIO). The PSWD-VIO field is incremented by one for every password violation incurred within the same date. Any password violations incurred after the current value in PSWD-DAT will cause the PSWD-VIO count to be reset to 1 and the PSWD-DAT field will be updated to reflect the current date. The only time the PSWD-VIO field is physically set to zero (0) is when the password is changed or the security administrator resets the field.

TSO

Jane's default TSO prefix is the same as her logonid (USER01). Her default SYSOUT and message classes (DFT-SOUT and DFT-SUBM) are A, and she can receive messages from other TSO users (INTERCOM). She can submit jobs from TSO (JCL), and can specify any region size at logon time (LGN-SIZE). For brevity, the other fields are not described here but are described later in this chapter.

Statistics

To date, she has had a total of only one security violation (SEC-VIO). Her logonid record was last updated at 09:21 on 08/11/04 (UPD-TOD).

Restrictions

The PREFIX field is the only field shown here. Jane's prefix is USER01 (the same as her logonid). This field gives Jane ownership of all data sets with a high-level index of USER01. For example, she has ownership over the data sets USER01.WORK.TEXT and USER01.STATS.MASTER. No security checking is done for "owned" data sets unless RULEVLD is specified on the logonid record.

DFP

The SMSINFO field points to a CONTROL(SMS) infostorage record (DEFPROD) that holds the default storage management class values for production data sets. When Jane attempts to allocate a production data set, these defaults will be tested by the Automatic Class Selection (ACS) routines to establish the storage management class values for those data sets.

eTrust CA-ACF2 does not supply all sections for every user. For example, if no Cancel/Suspend fields are active for a given user, the display of that user's logonid record does not contain the Cancel/Suspend section lines.

Logonid Privileges and Authorities

The fields you define in a logonid record determine if it can access the system, when it can access the system, where it can access the system from, and what it can do after it gains access to the system. This section provides an overview of some of the privileges and authorities you can define in a logonid record.

Also see Managing Passwords later in this chapter, for information on specifying password controls.

eTrust CA-ACF2 Privileges

Some logonid record fields grant special eTrust CA-ACF2 privileges or special access to system data and resources. You can assign a logonid more than one privilege. For example, a logonid can have SECURITY and ACCOUNT privileges, which gives it all authorities associated with the SECURITY privilege and all authorities associated with the ACCOUNT privilege. See Identifying Who Can Maintain Logonid Records later in this chapter for more information.

Logonid records with the following privileges can perform the actions described.

ACCOUNT

A user with the ACCOUNT privilege defined in their logonid record can create, change, delete, and display logonid records. A logonid with this privilege is known as an account manager. A user with this privilege is authorized to display the records and the parameters of the ACF2 system.

SECURITY

A user with the SECURITY privilege defined in their logonid record can access all data sets, protected programs, and resources. A logonid with this privilege is known as a security administrator. A security administrator has unlimited access to system resources, unless you create a scope record that restricts the user's privileges or use RULEVLD or RSRCVLD on the logonid record. A security administrator can also maintain access and resource rules, maintain infostorage records, and change certain fields of logonid records. However, a security administrator cannot insert or delete logonid records unless the user's logonid record also has the ACCOUNT privilege. A user with this privilege is authorized to display the records and the parameters of the ACF2 system.

AUDIT

A user with the AUDIT privilege defined in their logonid record can display logonid records, access and resource rules, and infostorage records. A logonid with this privilege is known as an auditor. An auditor can issue the ACF SHOW subcommands that display eTrust CA-ACF2 system control options, but an auditor cannot modify any of these components of the eTrust CA-ACF2 system. An auditor cannot update or delete logonid records or access any resources other than those authorized through rules, although a site can authorize auditors to update certain logonid record fields. The AUDIT privilege also gives users search and read access to directories in HFS.

CONSULT

A user with the CONSULT privilege defined in their logonid record can display all logonid records except those logonids with higher privileges, such as SECURITY. A CONSULT logonid cannot modify any logonid records unless the individual fields in the ACFFDR has been modified to allow this. Your site can determine the fields permissible for display and update. A logonid with this privilege is known as a consultant. A consultant can help answer users' questions about their logonid privileges.

LEADER

A user with the LEADER privilege defined in their logonid record can display most logonid records and also has additional authority for updating selected fields of the logonid records as specified by your site.

How to Limit Administrative Privileges

You can use a *scope record* to limit a user's access to the Logonid, Rule, and Infostorage databases. Scope records limit the powers of a privileged user. The authority you grant depends upon the privilege the user has. For example, adding a scope record to a logonid with the ACCOUNT privilege limits that user's access to those logonid records related to his office, instead of to his entire company.

To establish these limits, you must create a scope record and make entries in the LID and UID fields related to those logonids that the user creates. Next, you must enter the name of the scope record in the SCPLIST field of that logonid before any restrictions apply. For more information about how to create a scope record, see the "Maintaining SCOPE records" chapter. For more information about how to specify the SCPLIST field of the logonid record, see Logonid Record Field Descriptions later in this chapter.

Special Use Privileges

Several other fields of the logonid record can grant special privileges that have major effects on eTrust CA-ACF2 validation. Use these fields with care. These privileges and the actions they permit are described in the following:

CMD-PROP

A user with the CMD-PROP privilege can use the SET TARGET command or the TARGET parameter on the INSERT, CHANGE, LIST, and DELETE commands to override the global CPF target list. For details on CPF, see the "Using the Command Propagation Facility" chapter.

DUMPAUTH

A user with the DUMPAUTH privilege defined in their logonid record can generate a storage dump even when its address space is in an execute-only environment. EXECUTE-ONLY status is set for a job step when any data set in that job step is allowed access by a rule that specifies PROGRAM or LIBRARY, or allows only an EXECUTE level of access. The dump suppression applies to SYSUDUMP, SYSMDUMP, and SYSABEND dumps and does not affect system SVC dumps (those written to SYS1.DUMP*nn*).

MAINT

A user with the MAINT privilege defined in their logonid record as defined in the GSO MAINT record is allowed any type of access to a data set referenced in a maintenance job **without logging or rule validation**. A specific program must be executed from a specific library. Otherwise, the access is subject to normal eTrust CA-ACF2 validation. For more information about how to specify the library and program names, see System Maintenance Options (MAINT) in the chapter, "Maintaining Global System Options Records."

MUSASS

A user with the MUSASS privilege defined in their logonid record indicates that the logonid is for a multiple-user single address space system (MUSASS), such as CICS or IMS. A MUSASS has more authority than a normal user and can access resources on behalf of its users.

NON-CNCL

A user with the NON-CNCL privilege defined in their logonid record has full access to any data set or resource despite any security violations that can occur during the access attempt. These violations are logged by eTrust CA-ACF2. However, the logonid can access a data set or resource **without logging** as long as the access is defined by an existing data access or resource rule, or is permitted by virtue of the logonid's PREFIX field. All accesses outside of those normally permitted by the PREFIX or rules are permitted, but logged.

You cannot create a scope record to limit the access of a logonid with the NON-CNCL privilege.

PRIV-CTL

The PRIV-CTL privilege defined in a logonid record indicates that the logonid is valid for dynamic logonid privileges. When the user accesses the system (system entry), a special set of logonid privilege resource rules are checked to determine if the logonid should be assigned some dynamic privileges or authorities. See Providing Dynamic Logonid Privileges in the "Controlling System Entry" chapter for more information about dynamic logonid privileges.

Note: A user with the PRIV-CTL privilege cannot specify the STC bit in the logonid.

READALL

A user with the READALL privilege defined in their logonid record can read data and execute all programs at a site, regardless of what access rules might specify. This privilege is like the NON-CNCL privilege except that READALL grants read or execute access only. This privilege also bypasses PDS member-level validation.

Any read or execute accesses that violate access rules are allowed but logged. Other types of access, such as write, are validated just as they are when the user does not have this attribute.

STC

The STC privilege defined in a logonid record indicates that the logonid is for use by started tasks only. Validation of started tasks is determined by the STC field of the GSO OPTS record. For details on the STC field of the GSO OPTS record, see eTrust CA-ACF2 Option Specifications (OPTS) in the "Maintaining Global System Options Records" chapter.

Masking Logonid Records

Before you learn how to use eTrust CA-ACF2 to maintain logonid records, you should understand how to mask logonid records. A mask lets you specify multiple logonids at one time. For example, the logonid mask PAY*** represents all logonids that begin with the letters PAY and end with any zero to three characters.

Masking is useful for processing multiple logonid records with a single ACF subcommand. Two different symbols can signify masking: the asterisk (*) and the dash (-).

Using the Asterisk

A logonid mask that contains asterisks represents all valid logonids that begin with the specified characters and end with any number of characters that do not exceed the number of asterisks. These characters can include blanks.

For example, PAY*** matches the following:

```
PAY
PAYDS
PAYDSJ
```

However, PAY*** does not match:

```
PA
PBYKL
PAYKLST
```

Using the Dash

A logonid mask that contains a dash represents all valid logonids that begin with the specified characters. The total length of a valid logonid must be eight or fewer characters.

For example, PAY- matches the following:

```
PAY
PAYDS
PAYDSJ
PAYDS12J
```

However, PAY- does not match:

```
PA
PBYKL
PAVKLSTS
```

Identifying Who Can Maintain Logonid Records

The following types of users can maintain logonid records: security administrators, account managers, group leaders, consultants, and users.

All users can display their logonid record and change certain fields. The following table summarizes the types of access to the Logonid database eTrust CA-ACF2 grants to users with the following privileges as the product is shipped. What each privilege can display or modify can be adjusted in the ACFFDR assembly by using the @CFDE macros. See the *Getting Started* guide for more information on these changes:

Privilege	Access
ACCOUNT	Change, create, delete, and display
SECURITY	Display and change
LEADER	Display and change certain fields
CONSULT	Display certain fields
USER	Display and change certain fields in own record

This section describes these users, what they can do, and then describes other means to limit access to logonid records.

Security Administrators

A security administrator has the SECURITY privilege defined in the logonid record. Security administrators can display and change fields of logonid records based on their scope. If the security administrator does not have the SCPLIST field defined in their logonid record, then the security administrator can display and change any logonid record field. If the security administrator has RSRCVLD set and inserts or changes the GROUP field, a TYPE(TGR) resource validation call is issued for the GROUP name. However, a security administrator cannot create or delete logonid records unless the security administrator's logonid record also has the ACCOUNT privilege level. To display or change a security logonid record, you must have the SECURITY or ACCOUNT privilege. If the security logonid record is unscoped, you must also be unscoped.

Logonids with the authority to alter the GROUP field in the logonid record are also required to have permission to use the value specified for the GROUP field. This permission is normally given by granting access to the group name via the type TGR resource rules. If the logonid is a scoped SECURITY logonid, permission can also be given by adding the desired TGR group values to an INF scope.

Account Managers

An account manager has the ACCOUNT privilege defined in the logonid record. An account manager can display and change any field of the logonid record based on their scope, and can create and delete any records within their scope. If an account manager does not have the SCPLIST field defined in their logonid record, the account manager can display, change, create, and delete any logonid record. To display or change an account manager logonid record, you must have the ACCOUNT privilege. If the record is unscoped, you must also be unscoped. However, an account manager that has only the ACCOUNT privilege cannot change or list the logonid record of a user with the ACCOUNT and SECURITY privileges. eTrust CA-ACF2 considers a user with both these privileges more powerful than a user with only one of these. Additionally, a logonid with the ACCOUNT privilege cannot synchronize the records for SECURITY logonids with the UADS unless the account manager also has the SECURITY privilege.

Auditors

An auditor has the AUDIT privilege defined in the logonid record. An auditor can display any logonid record. If scoped, the auditor can only display logonid records with the same scope. You must have the SECURITY, ACCOUNT, or AUDIT privilege to display or change an auditor's logonid record.

Group Leaders

A group leader has the LEADER privilege defined in his logonid record. Group leaders are usually scoped. This means that they can only display and change certain fields of logonid records for their group only. They cannot display the logonids for users in another group. You must have the SECURITY, ACCOUNT, AUDIT, or LEADER privilege in your logonid record to display or change a group leader logonid record.

Consultants

A consultant is a user who has the CONSULT privilege in his logonid record. Consultants are usually scoped. This means that they can only display certain fields of logonid records based on their scope. They cannot display the logonids for users outside their scope. You must have the SECURITY, ACCOUNT, AUDIT, or CONSULT privilege in your logonid record to display or change a consultant logonid record.

All Users

If a logonid has no special privileges, you must have SECURITY, ACCOUNT, AUDIT, LEADER, or CONSULT to change it.

Updating the @CFDE Entry in the ACFFDR

You specify the authority required to display or change a logonid field in the @CFDE entry of the ACFFDR. This authority is known as *field-level authority*. The LIST= operand specifies the privilege required to display a logonid record field. The ALTER= operand describes the privilege required to change a logonid record field. eTrust CA-ACF2 provides default values in the @CFDE entry for every logonid field. To change the privilege required to display a particular logonid field, specify a different value.

For example, only a user with the ACCOUNT or SECURITY privilege level has the authority to change the NAME field of a particular logonid record, but any user can display the NAME field when he lists the logonid record. Of course, a user must also be able to access the logonid record.

How to Maintain Logonids in a CPF Environment

In a CPF environment, logonid maintenance can be performed on multiple databases with a single command. Users who can maintain logonids on one node might not have the same authority on another node. CPF does not increase a user's authority on other nodes. All commands received on a node through CPF verify that the issuer has the proper authority on the receiving node to issue the command.

The CMD-PROP privilege prevents users from overriding the global defaults set by the system administrator. If the default target list (defined in the DFTCMD parameter of the CPF OPTIONS record) specifies nodes A, B and C, users are not able to send commands to node D using the TARGET parameter unless their logonid has the CMD-PROP privilege set.

How to Specify Logonid and User Profile Record Fields

The external names of the logonid record fields, as described in the Logonid Record Fields section later in this chapter, are one to eight characters in length and are used to display the contents of a logonid record for modification. Each field name has various parameters associated with it that tell eTrust CA-ACF2 how to process the field. The valid field types are described in the following sections.

Bit Fields

Bit fields are associated with bits that are turned on through specification of the field name. For example, TSO grants the TSO attribute. Specification of the field name preceded by the keyword NO turns the bit off. NOTSO takes away the TSO attribute. On output, the field is listed only as the field name or the field name preceded by the word NO. Optionally, the listing of the field can be suppressed altogether if the bit is off.

Character Fields

Character fields are specified with the field name followed by the value of the field in parentheses. To specify a blank field, the null string of () can be used. Optionally, the listing of the field can be suppressed if the field is blank or zero.

Packed Decimal Date Fields

Packed decimal date fields are specified by the field name followed by the Gregorian date in parentheses. The format specified in the DATE field of the GSO OPTS record determines how you should enter dates (mm/dd/yy, dd/mm/yy, yy/mm/dd). For more information about DATE, see the chapter entitled "Maintaining Global System Options Records." Dates entered as 00-69 are assumed to be in the 21st century (2000-2069); dates entered as 70-99 are assumed to be in the 20th century (1970-1999). Optionally, the listing of the field can be suppressed if the value is zero.

Binary Fields

Binary fields can be defined as being one, two, three, or four bytes long. This byte length determines the maximum value that can be assigned to the field. Only binary two- and four-byte fields can contain negative values. The permissible formats are listed in the following:

- **ddddd**—up to five binary digits with no sign assigns the specified value to the field.
- **+ddddd**—the string preceded by a plus sign increments the field value by the specified value.
- **-ddddd**—the minus sign decrements the field value by the specified value.

Optionally, the listing of the field can be suppressed if the value is zero.

Hexadecimal Fields

Hexadecimal fields can be defined as a maximum of eight bytes (16 hexadecimal digits). The valid operands are a string of valid hexadecimal digits (0-9, A-F) of even length, enclosed in parentheses. The assignment to the field is left-justified. Trailing zeros are truncated. Optionally, you can suppress the listing of the field if the value is zero.

Abbreviating Logonid and User Profile Record Fields

Logonid record fields and user profile records field names can be abbreviated when specified on the ACF command. Fields might be abbreviated to the minimum number of letters required to differentiate between other fields in the logonid record and user profile records. If abbreviations match more than one field name or keyword, they are treated as invalid.

Adding Fields to the Logonid Record

When adding fields to the logonid record:

1. Determine additional or different fields to be used and write the field definition @CFDE macros.
2. Modify the USERLID section of the LIDREC DSECT to reflect any new @CFDE entries.
3. IPL with the ACFFDR that contains the new @CFDE entries and modify all logonid records so that the contents of these fields are meaningful for the type of field. You can use the following command for global changes:

   ```
   ACF CHANGE LIKE(-)
   ```

For more information about updating logonid record fields in the ACFFDR, see the *Systems Programmer Guide*.

Managing Passwords

eTrust CA-ACF2 contains many features for managing passwords. This section provides an overview of the logonid record fields and GSO records you can use to control passwords at your site.

Note: eTrust CA-ACF2 rejects access attempts that use the underscore (_) in passwords at TSO logon.

Password Related Logonid Record Fields

eTrust CA-ACF2 stores a user's password in their logonid record in a one-way encrypted format. Users must memorize their passwords because eTrust CA-ACF2 never displays them. Not even a security administrator can display a user's password.

Although you cannot display another user's password, you can use the following logonid record fields to control password use:

- IDLE
- MAXDAYS
- MINDAYS
- MON-LOG|NOMON-LOG
- PASSWORD
- PRVPSWD1 through PRVPSWD4
- PRV-TOD1 through PRV-TOD4
- PSWD-DAT
- PSWD-EXP|NOPSWD-EXP
- PSWD-INV
- PSWD-SRC
- PSWD-TIM
- PSWD-TOD
- PSWD-VIO
- PSWD-XTR|NOPSWD-XTR
- RESTRICT|NORESTRICT
- STC|NOSTC

Note: VM sites can also use the following logonid record fields to specify password controls: AUTOALL, AUTONOPW, DG84DIR, and VMSAF.

For descriptions of these fields, see Logonid Record Field Descriptions later in this chapter.

Password Related GSO Records

You can use the following global system options records to exert tighter controls over password specifications: PSWD and RESWORD.

PSWD Record Options

Use the GSO PSWD record to specify global system option password controls. See Password Maintenance and Support (PSWD) in the "Global System Options Records" chapter for detailed information.

The following table describes some of the options you can specify with fields on the PSWD record:

PSWD Field	Description
NOPSWDMIXD	Prevents the use of mixed-case passwords
NOPSWDVOWL	Prevents the use of vowel characters to be specified in new password
PSWDALPH	Requires at least one alphabetic character in new password.
PSWDHST	Prevents the use of up to four previous passwords as a new password. **This eliminates the need for eTrust CA-ACF2 Note 12.**
PSWDLID	Prevents the use of a user's logonid as a new password.
PSWDNMIC	Requires at least one numeric character in new password.
PSWDNUM	Prevents the use of a numeric password.
PSWDPAIR	Specifies the number of consecutively repeated characters allowed to be in a password.
PSWDPLST ()	Allows the user-defined characters to be specified in new password.
PSWDREQ	Prevents the use of logonids without passwords, except for logonids with STC and RESTRICT values. See Logonids Must Have Passwords next for more information.
PSWDRSV	Prevents the use of a set of reserved word prefixes as the new password.
PSWDSPLT	Requires national character (@, #, $) or user-defined character list in PSWDPLST between the first and last character position in new password.

PSWD Field	Description
PSWDVFY	Prompts for a new password verification whenever a logonid password is updated.
PSWXHIST	Up to 64 previous passwords as a new password. **This eliminates the need for eTrust CA-ACF2 Note 12.** Use the PSWXHST# field to set the number of previous passwords to be retained (up to 64).

Specifying NOPSWDxxx for these fields indicates that the option is not active.

Administrators can also control when users are able to change their passwords using the PSWDNCH|NOPSWDNCH field of the GSO PSWD record. Specifying PSWDNCH indicates that users can change their passwords only at system entry time, ensuring that the password change process happens through a trusted path. Specifying NOPSWDNCH lets users change their passwords at their own discretion.

Logonids Must Have Passwords

By default, eTrust CA-ACF2 requires passwords for all logonids on the database. If you try to insert or change a logonid that does not already have a password defined for it, eTrust CA-ACF2 issues an error message informing you that a password is required. You must include a password when you create a new logonid or change an existing logonid.

This requirement prevents potential security exposures that could arise if a logonid was created without a password or if an existing logonid with the RESTRICT or STC privileges had those privileges subsequently removed. In both cases, a logonid would exist on the database without an associated password, leaving open the possibility that someone could enter the logonid only and gain system access.

By checking for passwords whenever a new logonid is inserted or an existing logonid is changed, eTrust CA-ACF2 secures system entry to authorized users only.

The PSWDREQ field of the GSO PSWD record sets this requirement. You can change the default by specifying NOPSWDREQ in the GSO PSWD record. See Password Maintenance and Support (PSWD) in the "Maintaining Global System Options Records" chapter for details about this option.

RESWORD Record

The GSO RESWORD record lists the words or prefixes that are not allowed in the specification of passwords. Use the SHOW RSVWORDS subcommand to display this list.

Logonid Record Fields

Logonid record fields contain information to identify a user's privileges, such as:

- Whether he can use CICS or TSO
- How many security violations he has accumulated
- What his TSO privileges are
- How often he must change his password

The logonid record describes a system user. The fields of the logonid record let eTrust CA-ACF2 identify and validate each user request on an individual basis. This section describes each of the logonid record fields. The fields are listed alphabetically. We list the defaults for bit fields, for example NOACCOUNT. However, eTrust CA-ACF2 never displays the defaults for bit privileges. If a logonid does not have a privilege, eTrust CA-ACF2 does not display that privilege.

Logonid Record Field Descriptions

An asterisk (*) before the field name indicates that eTrust CA-ACF2 uses this field if you choose not to use UADS by specifying NOUADS in the GSO OPTS record. See Choosing the UADS or NOUADS Option in the "Controlling System Entry" chapter or the section on GSO OPTS records in the "Maintaining Global System Options Records" chapter.

ACC-CNT(*count*)
The count of the number of system accesses made by this logonid since it was created. This field is displayed and maintained by eTrust CA-ACF2. (Four-byte binary field)

ACC-DATE(*date*)
The date of the last system access by this user. This field is displayed and maintained by eTrust CA-ACF2. The date is displayed in the format *mm/dd/yy, dd/mm/yy,* or *yy/mm/dd*, depending on the DATE field in the GSO OPTS record. Year designations of 70-99 assume a date in the 20th century (1970-1999); year designations of 00-69 assume a date in the 21st century (2000-2069). You cannot set this field. This field will be updated only once a day for default logonids kept in storage. (Four-byte packed field)

ACC-SRCE(*source*)
The one to eight-character logical or physical input source name or source group name from which this user last accessed the system. This field is displayed and maintained by eTrust CA-ACF2. (Character field)

ACC-TIME(*hh.mm*)
The time of the last system access by this user. This field is displayed and maintained by eTrust CA-ACF2. The display format is *hh.mm*. (Four-byte binary field)

ACCOUNT | NOACCOUNT
Specifies that this user can insert, list, change, and delete logonid records. You can restrict this privilege with a scope record. A user with ACCOUNT only or SECURITY only privilege cannot list or change a logonid record of a user who has both ACCOUNT and SECURITY, because the user with both is more powerful than a user with only one of these two authorities. (Bit field)

***ACCTPRIV | NOACCTPRIV**
Indicates user has TSO accounting privileges (for UADS updates with the TSO ACCOUNT command). (Bit field)

ACF2CICS | NOACF2CICS
Indicates that eTrust CA-ACF2 CICS security is to be initialized in any CICS region running with this address space logonid. This bit has no effect on batch, TSO, other MUSASS environments such as IMS. If you are running your CICS using CICS external security facilities with eTrust CA-ACF2 SAF controls do not turn this bit on. (Bit field)

ACTIVE(*date*)
Activates the logonid one minute after midnight on the date contained in this field. Specify the date as mm/dd/yy, yy/mm/dd, or dd/mm/yy depending on the DATE field in the GSO OPTS record. Year designations of 70-99 assume a date in the 20th century (1970-1999); year designations of 00-69 assume a date in the 21st century (2000-2069). The logonid cannot be used until the specified date. (Four-byte packed field)

ALLCMDS | NOALLCMDS
Indicates the ability to bypass the eTrust CA-ACF2 restricted command lists by entering a special prefix character. (Bit field)

***ATTR2(*pscbatr2*)**
Specifies the PSCBATR2 field used by the IBM Program Control Facility (PCF) for command limiting and data set protection. The default value is zero (no PCF controls). You can change this default if the user's logonid record is created with UADS in effect, if the user's logonid record is copied from a model logonid record, or if a new value is inserted. eTrust CA-ACF2 also has an option for TSO command limiting. (Two-byte hexadecimal field)

AUDIT | NOAUDIT
Specifies that this user can display the records and parameters of the eTrust CA-ACF2 system. This privilege can be limited by a scope. This also gives the user audit privileges in an OMVS system. (Bit field)

***AUTHSUP1 through AUTHSUP8**
Specifies the extended user authentication (EUA) routine for a user. EUA lets you require some users to be processed for additional authentication beyond the normal eTrust CA-ACF2 logonid and password validation, and enables other users to sign on without further user authentication. A site can use up to eight different types of authentication routines in the site, but only one can be designated for any one user.

Since eTrust CA-ACF2 cannot anticipate what actual authentication routines a site might select, these fields have generic names. You can change the @CFDE macro external entry name to one more closely tied to the device name or application at your site. These routines are available for TSO and CICS logon validation only. (Bit fields)

AUTOALL|NOAUTOALL
Specifies that this user can autolog any virtual machines without specifying a password. This applies no matter what privileges the machines being autologged might have. This field applies to VM sites only. (Bit field)

AUTODUMP|NOAUTODUMP
Specifies that eTrust CA-ACF2 takes an SVC dump whenever a data set or resource violation occurs. Specify this field for debugging purposes only. (Bit field)

AUTONOPW|NOAUTOPW
Specifies that this virtual machine can be autologged without specifying a password. AUTONOPW has no effect on the logon process. This field applies to VM sites only. (Bit field)

AUTOONLY|NOAUTOONLY
Specifies that this virtual machine cannot be logged on. It can be autologged only. This field applies to VM sites only. (Bit field)

BDT|NOBDT
Specifies that the address space for this logonid belongs to the Bulk Data Transfer (BDT) product. Since BDT can use several address spaces at once, certain privileges associated with BDT should also extend to this address space. (Bit field).

CANCEL|NOCANCEL
Specifies that this logonid cannot be used to access the system. eTrust CA-ACF2 does not differentiate between CANCEL and SUSPEND (described in the following), except as to who can alter or display the fields as defined in the ACFFDR. Your site should establish procedures for the use of these two fields. (Bit field)

***CHAR(*char*)**
The TSO character-delete character for this user. Specify a single character or use one of the special strings **BS** (signifying the backspace key, X'16') or **NO** (signifying no character-delete character desired). (One-byte binary field)

CICS|NOCICS
Specifies whether this user can sign on to CICS. This is the field that eTrust CA-ACF2 checks for CICS authority. By changing the eTrust CA-ACF2 CICS system initialization parameters, a site can choose to use a different field for one or more CICS regions. For more information, see the eTrust CA-ACF2 CICS Parameters chapter in the *CICS Support Guide*. (Bit field)

CICSCL(*class*)
Indicates CICS operator class. For CICS support only. (Three-byte hexadecimal field)

CICSID(*id*)
Indicates CICS operator ID. For CICS support only. (Three characters)

CICSKEY(*key*)
Contains the first three bytes of transaction security key values (refer also to CICSKEYX). Use hexadecimal notation, such as, CICSKEY(00000F). This field is obsolete for all currently supported CICS releases and is subject to future reuse by Computer Associates. (Three-byte hexadecimal field)

CICSKEYX(*key*)
Contains the last five bytes of transaction security key values. Use hexadecimal notation. This field is obsolete for all currently supported CICS releases and is subject to future reuse by Computer Associates. (Five-byte hexadecimal field)

CICSOPT(*cicsopt*)
Specifies the SYSID of the C-CIC records to use at initialization time. If CICSOPT is blank or null, eTrust CA-ACF2 CICS will only use the ACF2PARM parameter file (or applicable default values if ACF2PARM is not supplied or cannot be opened). This file cannot be masked. (Eight characters)

CICSPRI(*class*)
Indicates CICS operator priority. For CICS support only. (One-byte binary field)

CICSRSL(*key*)
Indicates CICS resource access key. For CICS support only. (Three-byte hexadecimal field)

CMD-LONG|NOCMD-LONG
Indicates that eTrust CA-ACF2 should bypass the TSO command list feature for this user. That is, the user must enter the full command names or aliases to issue the command. Normally, eTrust CA-ACF2 accepts the shortest character string that uniquely identifies a command in the list. (Bit field)

CMD-PROP|NOCMD-PROP
Indicates that the user can use the SET TARGET command or the TARGET parameter on the INSERT, CHANGE, LIST and DELETE commands to override the global CPF target list. For details on CPF, see the chapter entitled, "Using the Command Propagation Facility." (Bit field)

CONSOLE|NOCONSOLE
Permits you to access the TSO/E CONSOLE facility. (Bit field)

CONSULT|NOCONSULT
Specifies that this user can display other logonid records. Most sites scope this privilege. (Bit field)

CSDATE(*date*)
Specifies the date that the CANCEL, SUSPEND, MON-LOG, or MONITOR field was set for this user. You cannot set this field. eTrust CA-ACF2 displays and maintains it. The date is displayed in the format mm/dd/yy, dd/mm/yy, or yy/mm/dd, depending on the DATE field of the GSO OPTS record. Year designations of 70-99 assume a date in the 20th century (1970-1999); year designations of 00-69 assume a date in the 21st century (2000-2069). (Four-byte packed field)

CSWHO(*logonid*)
Specifies the eight-character logonid of the user who set the CANCEL, SUSPEND, MON-LOG, or MONITOR field for this user. You cannot set this field. eTrust CA-ACF2 displays and maintains it. (Eight characters)

***DFT-DEST(*dest*)**
Specifies the default remote destination for TSO spun SYSOUT data sets. (Eight characters)

***DFT-PFX(*prefix*)**
Specifies the one to eight-character (the last character is reserved) default TSO prefix that is set in the user's profile at logon time. This prefix is assumed as the high-level index whenever the user specifies a data set name that is not fully qualified and in single quotes. (This prefix operates like the UADS Profile Prefix field.) A DFT-PFX value of period (.) indicates that the user must always specify a fully qualified data set name. Any security administrator who adds or changes this field must not have an associated scope record that restricts access to those data sets with the high-level index to be specified in this field. (Eight characters, but the last character is reserved)

DFT-SOUT(*class*)
Specifies the default TSO SYSOUT class. For TSO or TSO/E Command Package Program product only. (One character)

DFT-SUBC(*class*)
Specifies the default TSO submit class. For TSO or TSO/E Command Package Program product only. (One character)

***DFT-SUBH(*class*)**
Specifies the default TSO submit hold class. For TSO or TSO/E Command Package Program product only. (One character)

DFT-SUBM(*class*)
Specifies the default TSO submit message class. For TSO or TSO/E Command Package Program Product only. (One character)

DG84DIR | NODG84DIR
Permits a user to issue a diagnose 84 instruction. This lets a VM user dynamically update certain fields in a VM Directory entry. This privilege lets a site use the VM Directory password or the user's eTrust CA-ACF2 password to update the specific fields. This field applies to VM sites only. (Bit field)

DIALBYP | NODIALBYP
Permits standard DIAL validation to be bypassed. When a user dials to a secured target machine with this privilege, standard DIAL validation does not take place. This field applies to VM sites only. (Bit field)

DUMPAUTH | NODUMPAUTH
Specifies that this user can generate a dump, even in an execute-only environment. If the logonid is in that state without this privilege, he cannot generate a dump. (Bit field)

EXPIRE(*date*)
Indicates when the privileges for this logonid will expire. When the specified date is reached, the user is no longer able to log on or submit jobs. He receives a LOGONID EXPIRED message. Specify the date in the format mm/dd/yy, dd/mm/yy, or yy/mm/dd, depending upon the DATE field of the GSO OPTS record. Year designations of 70-99 assume a date in the 20th century (1970-1999); year designations of 00-69 assume a date in the 21st century (2000-2069). To deactivate this expiration date, change the logonid record and specify the EXPIRE(0) parameter. (Four-byte packed field)

GROUP(*grpname*)
Specifies the one to eight-character group name that is the default for a logonid. Users can specify a group name at system entry in one of three ways: by specifying GROUP= on the command line, by specifying GROUP= on the TSO full-screen logon panel, or by using the JCL GROUP= parameter. At system entry, eTrust CA-ACF2 validates whether a user belongs to a group other than the default from this logonid field by consulting resource rules with the type code TGR. When the GROUP field is specified in the @UID macro in the ACFFDR, eTrust CA-ACF2 dynamically creates the UID for a user every time that user accesses the system, using whatever group name they have specified as the GROUP= to validate. If a user specifies no GROUP at system entry, eTrust CA-ACF2 performs validation against this default group name to determine the user's access to the system. (Eight characters)

This field should not contain any embedded blanks. Also, this field cannot be changed to a multi-value field. For more details, see Specifying a Group or Project Name at Logon in the "Controlling System Entry" chapter.

GRPLOGON | NOGRPLOGON
Specifies that this virtual machine can be used by more than one person, while maintaining individual accountability. This type of virtual machine is sometimes referred to as a maintenance machine and usually has a centralized function. This field applies to VM sites only. (Bit field)

GRP-OPT | NOGRP-OPT

Designates an ID as an optional group ID. A logonid with this attribute can be logged onto as the primary ID, or a group ID. To access a virtual machine with this attribute as a group ID, this privilege must be present and a group resource rule must exist. If the GRP-OPT and the GRPLOGON field are present, the GRPLOGON attribute takes precedence. GRP-OPT requires using LOGON BY when logging on as a group user. This field applies to VM sites only.

GRP-USER(*logonid*)

Indicates the last user (logonid) to use the group virtual machine. If this machine is not a group machine, eTrust CA-ACF2 does not display this field. This field applies to VM sites only. **Note:** You cannot set this field. eTrust CA-ACF2 displays and maintains this field.

HOMENODE(*nodeid*)

Specifies the one to eight-character name of the node where this logonid record is stored in a Logonid database. eTrust CA-ACF2 uses this field to keep track of where the logonid record resides in a distributed database (DDB) network. You cannot specify this field; it is displayed and maintained by eTrust CA-ACF2.

IDLE(*time*)

Specifies the maximum time permitted (in minutes) between terminal transactions for this user. If exceeded, eTrust CA-ACF2 causes the logonid and password to be revalidated before another transaction is accepted. Zero indicates no limit is enforced. This field is available for IMS and CICS online processing. (One-byte binary field)

IDMS | NOIDMS

Specifies whether the user can sign on to CA-IDMS. This is the default field that eTrust CA-ACF2 checks for CA-IDMS authority. By changing the CA-IDMS @MOPT macro, a site can choose to use a different field. See the chapter on eTrust CA-ACF2 IDMS option selection in the eTrust CA-ACF2 6.2 *CA-IDMS Support Guide*. (Bit field) This field is inactive without the eTrust CA-ACF2 IDMS interface. CA-IDMS has created an external security interface in their product eliminating the need for a specific eTrust CA-ACF2 IDMS interface. This field is now obsolete.

IDMSPROF(*profilename*)

Specifies the one to 32-character name of the sign-on profile CLIST executed when the user signs on to CA-IDMS. (32 characters) This field is inactive without the eTrust CA-ACF2 IDMS interface. CA-IDMS has created an external security interface in their product eliminating the need for a specific eTrust CA-ACF2 IDMS interface. This field is now obsolete.

IDMSPRVS(*profilename*)
Specifies the version of the sign-on profile CLIST executed when the user signs on to CA-IDMS. (Two-byte binary field) This field is inactive without the eTrust CA-ACF2 IDMS interface. CA-IDMS has created an external security interface in their product eliminating the need for a specific eTrust CA-ACF2 IDMS interface. This field is now obsolete.

IMS|NOIMS
Specifies whether this user can sign on to IMS. This is the default field that eTrust CA-ACF2 checks for IMS authority. Through the IMS AUTH operand, a site can choose to use another field. (Bit field)

***INTERCOM|NOINTERCOM**
Indicates this user is willing to accept messages from other users through the TSO SEND command. (Bit field)

***JCL|NOJCL**
Indicates the ability to submit batch jobs from TSO and to use SUBMIT, STATUS, CANCEL, and OUTPUT commands (for example, use TSO SUBMIT). (Bit field)

JOB|NOJOB
Specifies that batch or background jobs can use this logonid. If you //*LOGONID in the JCL for a batch or background job, that logonid must have the JOB privilege. Or, if the batch or background job inherits the logonid of the submitter, the submitter must have the JOB privilege.

eTrust CA-ACF2 checks this field if the JOBCK field of the GSO OPTS record is specified. For more information, see Submitting Batch Jobs in the chapter entitled, "Controlling System Entry," or for details on the JOBCK field of the GSO OPTS record, see the "Maintaining Global System Options Records" chapter. (Bit field)

JOBFROM|NOJOBFROM
Specifies that this user can use //*JOBFROM control statements. Specify this field in the record for all multiple-user single address space systems (MUSASSes) such as, CICS, IMS, and CA-ROSCOE or in batch production environments like CA-7. This control statement allows another logonid and source to be transmitted by the MUSASS for any jobs submitted by that MUSASS without knowing the password. This privilege is not limited to MUSASS environments. (Bit field)

KERB-VIO
Specifies the number of Kerberos key violations. This field is similar to the PSWD-VIO count and will be used with the PSWD-VIO count to suspend the user's LID when the combined counts exceed the global PSWDLMT count field.

KERBCUR
Specifies the current Kerberos key. This field is not modifiable via the ACF command and is only updated when the password is modified.

KERBCURV
Specifies the current Kerberos key version. This field is not modifiable via the ACF command and is only updated when the password is modified.

KERBPRE
Specifies the previous Kerberos key. This field is not modifiable via the ACF command and is only updated when the password is modified.

KERBPREV
Specifies the previous Kerberos key version. This field is not modifiable via the ACF command and is only updated when the password is modified.

LDS | NOLDS
Specifies whether Logonid administrative changes for this user will be propagated to the active Lightweight Directory Access Protocol (LDAP) servers in the network. (Bit field)

LDEV | NOLDEV
Specifies that this user can create logical devices using the IBM Pass-Through Virtual Machine (PVM) product. This privilege applies only when the optional eTrust CA-ACF2 intercept is in place. This field applies to VM sites only.

LEADER | NOLEADER
Specifies that this user can display and alter certain fields of logonid records for other users. Most sites scope this privilege. (Bit field)

***LGN-ACCT | NOLGN-ACCT**
Indicates permission to specify an account number at logon time. If a user has the PMT-ACCT field, eTrust CA-ACF2 prompts him for an account number unless he specified an account number before the prompt. If a user does not specify an account number at logon and PMT-ACCT is not specified in his logonid record, eTrust CA-ACF2 uses the user's default account number (TSOACCT in the logonid record) or the system default account number. Specify the default in the ACCOUNT field of the GSO TSO record.

If a user does not specify an account number and no user or system default is available, eTrust CA-ACF2 prompts him for an account number in spite of the settings of LGN-ACCT and PMT-ACCT. eTrust CA-ACF2 validates the account number against the TAC type resource rules if the user has the VLD-ACCT field in his logonid record. (Bit field)

***LGN-DEST | NOLGN-DEST**
Indicates permission to specify a remote output destination at TSO logon that overrides the value specified in the DFT-DEST field. (Bit field)

***LGN-MSG | NOLGN-MSG**
Indicates this user has permission to specify a message class at logon time. This might be helpful in debugging (in non-UADS mode only) by permitting the debugger to specify the message class for TSO session outputs. (Bit field)

***LGN-PERF|NOLGN-PERF**
Indicates permission to specify a performance group at logon time. (Bit field)

***LGN-PROC|LGN-PROC**
Indicates permission to specify the TSO procedure name at logon time. (Bit field)

***LGN-RCVR|NOLGN-RCVR**
Indicates permission to use the recover option of the TSO or TSO/E Command Package. If not specified, the user cannot enter the PROFILE RECOVER command. (Bit field)

***LGN-SIZE|NOLGN-SIZE**
Indicates that this user is authorized to specify any region size at logon time (overriding TSOSIZE). A user can specify size at logon time without this field, but is restricted to a maximum size based on his TSOSIZE unless he has the LGN-SIZE field in his logonid record. (Bit field)

***LGN-TIME|NOLGN-TIME**
Indicates permission to specify the TSO session time limit at logon time. (Bit field)

***LGN-UNIT|NOLGN-UNIT**
Indicates permission to specify the TSO unit name at logon time. (Bit field)

LID
Contains the eight-character logonid of the user. This field is the key to the logonid record. (Eight characters)

LIDZMAX|NOLIDZMAX
Specifies that a zero value for the MAXDAYS field in the LIDREC will override the global PSWDMAX value in the GSO PSWD record.

LIDZMIN|NOLIDZMIN
Specifies that a zero value for the MINDAYS field in the LIDREC will override the global PSWDMIN value in the GSO PSWD record.

LIDTEMP
Specifies that the current password is a temporary password. This bit will be set if the current password was set by a non-owner of the LOGONID, such as a security administrator or account manager, and the password was immediately expired. This bit is not modifiable via the ACF command.

***LINE(*char*)**
Specifies the TSO line-delete character. Specify a single character or one of the following special strings:

- **ATTN**—the ATTENTION key
- **CTLX**—the CONTROL-X control character (X'18')
- **NO**—no line-delete character desired. (One character)

LOGSHIFT | NOLOGSHIFT
Specifies that a user can access the system outside of the time period specified in the SHIFT field of his logonid record. eTrust CA-ACF2 logs all such system accesses. You can display accesses outside a shift using the ACFRPTPW report. (Bit field)

MAIL | NOMAIL
Indicates that a user can receive mail messages from TSO at logon time. (Bit field)

MAINT | NOMAINT
Specifies that a logonid can access data sets without eTrust CA-ACF2 rule validation or loggings by means of a specified program executed from a specified library. The field is used in conjunction with the GSO MAINT record. (Bit field)

MAXDAYS(*days)*
Specifies the maximum number of days (based on the date specified in the PSWD-TOD field) permitted between password changes before the password expires. Zero indicates that the global MAXDAYS value specified in the PSWDMAX field of the GSO PSWD record will be used unless the LIDZMAX flag is set.

MINDAYS(*days)*
Specifies the minimum number of days that must elapse before a user can change his password. Zero indicates that the global MINDAYS value specified in the PSWDMIN field of the GSO PSWD record will be used unless the LIDZMIN flag is set.

***MODE | NOMODE**
Indicates this user wants to receive modal messages from TSO. (Bit field)

MON-LOG | NOMON-LOG
Specifies whether an SMF record is written (for the Invalid Password/Authority Log, ACFRPTPW) each time this user enters the system. No messages are sent to the user.

The STC field overrides the MON-LOG field. This means that no loggings will appear on the ACFRPTPW report for a logonid with the STC and MON-LOG privileges. (Bit field)

MONITOR | NOMONITOR
Specifies whether a message is sent to the security console and to a designated person each time this user enters the system. The CSWHO field in the logonid record of the user entering the system indicates the logonid of the designated person.

The STC field overrides the MONITOR field. This means that no messages are sent to the console for a logonid with the STC and MONITOR privileges. (Bit field)

***MOUNT | NOMOUNT**
Indicates permission to issue mounts for devices. (Bit field)

MSGID|NOMSGID
Indicates this user wants TSO messages to have message IDs prefixed. (Bit field)

MULTSIGN|NOMULTSIGN
Specifies that a user has multiple sign-on privileges (CICS-only).

MUSASS|NOMUSASS
Specifies that this logonid is for a multiple-user single address space system, such as CICS, or IMS. (Bit field)

MUSDLID(*logonid*)
Specifies the default logonid for a multiple-user single address space system (MUSASS) address space. The logonid specified **does not** need the MUSASS attribute. Currently, this field is used only for SAF. (Eight characters)

MUSID(*musid*)
Specifies a one to eight-character ID you assign to a multiple-user single address space (MUSASS). If the MUSID value is the name of a bit field in the logonid record, eTrust CA-ACF2 uses that field to validate the user's access to the MUSASS. For example, if a CICSPROD MUSASS logonid has a MUSID value of CICS, users accessing the CICSPROD region must have the CICS privilege in their logonids. If the MUSID value does not correspond to a logonid bit field, this access authorization check is not performed.

The eTrust CA-ACF2 IMS interface also uses the MUSID value to associate IMS control records with specific IMS regions. This is documented in the *IMS Support Guide*. When you use the MUSID parameter for this purpose, the MUSID value specified should not be the name of a logonid bit field, unless the bit field is also the access authorization field for the IMS region. (Eight characters)

MUSIDINF|NOMUSIDINF
Indicates that the MUSID field should be used to restrict access to a MUSASS region for eTrust CA-ACF2 Info type system entry calls. (Bit field).

MUSOPT(*module*)
Specifies the one to eight-character name of the eTrust CA-ACF2 IDMS options module that is used to control the CA-IDMS address space. You must specify this field in the logonid record assigned to the CA-IDMS address space. See the Logonid Record Considerations chapter in the eTrust CA-ACF2 6.2 *CA-IDMS Support Guide* for more information. (Eight characters) This field is inactive without the eTrust CA-ACF2 IDMS interface. CA-IDMS has created an external security interface in their product eliminating the need for a specific eTrust CA-ACF2 IDMS interface. His file is now obsolete.

MUSPGM(*program*)
Specifies the one to eight-character name of the CA-IDMS startup program. You must specify this field in the logonid record for the CA-IDMS address space. See the Logonid Record Considerations chapter in the eTrust CA-ACF2 6.2 *CA-IDMS Support Guide* for more information. (Eight characters) This field is inactive without the eTrust CA-ACF2 IDMS interface. CA-IDMS has created an external security interface in their product eliminating the need for a specific eTrust CA-ACF2 IDMS interface. His file is now obsolete.

MUSUPDT|NOMUSUPDT
A MUSASS logonid with this privilege has the authority to make calls on behalf of users who are updating the databases. The user must have the appropriate privilege to alter that data. (Bit field)

NAME(*username*)
Specifies the one to 20-character name of the user. eTrust CA-ACF2 displays this name on logging and security violation reports. eTrust CA-ACF2 also uses this name as the NAME field of the job statement created for a TSO logon session if you specify the NOUADS field in the GSO OPTS record. (20 characters)

NO-INH|NONO-INH
Indicates that a network job cannot inherit this logonid from its submitter. If a logonid with this attribute submitted a job from one node in a network to another at where the logonid is defined, eTrust CA-ACF2 does not let the job inherit the logonid. eTrust CA-ACF2 cancels the job and issues an error message. (Bit field)

NOMAXVIO|NONOMAXVIO
Prevents the user violation counter from incrementing and MAXVIO processing from occurring. Logonids with this attribute will never be canceled with S222 and the associated ACF99910 message. NOMAXVIO is intended for logonids associated with started tasks or production batch jobs that you want subject to validation, but not to cancellation. The OMVS started task is an example of this type of started task. (Bit field)

The GSO OPTS MAXVIO option specifies the maximum number of security violations that can occur in a single job or TSO session before eTrust CA-ACF2 terminates the job.

Note: eTrust CA-ACF2 external interfaces are not included.

NO-OMVS|NONO-OMVS
Specifies that this user cannot use any z/OS Unix System Services. NO-OMVS overrides any user OMVS profile record defined for the user. This user cannot use the defaults specified in the DFTUSER or DFTGROUP fields of the GSO UNIXOPTS record. (Bit field)

NO-SMC | NONO-SMC
Specifies that this user can bypass step-must-complete (SMC) controls. A job is considered noncancelable for the duration of the sensitive VSAM update operation. (Bit field)

NO-STATS | NONO-STATS
Specifies that the last access statistics on a successful full validation (ACVAMVAL) MUSASS signon request are bypassed. (Bit field)

Note: You must also specify MUSASS on the logonid for this privilege to be effective.

With a successful signon, eTrust CA-ACF2 will bypass writing an SMF record for the signon request. In addition, no ACF01134 or ACF01137 messages are returned to the caller. If the signon fails, an SMF record is written and the logonid database is updated.

Note: If the last access date is less then the current date, an SMF record is written and the logonid is updated in the database. The ACF01134 and ACF01137 messages are still suppressed.

NO-STORE | NONO-STORE
Specifies that a user cannot store or delete rule sets. This applies even if the value of the PREFIX field of his logonid record matches the $KEY of the rule of the data set, if he has the SECURITY privilege, or if he has change authority through a %CHANGE or %RCHANGE control statement in the rule set. (Bit field)

NON-CNCL | NONON-CNCL
Specifies that eTrust CA-ACF2 cannot cancel a user for security violations. The event log shows that the request was permitted because the user could not be canceled. (Bit field)

NOSPOOL(null | ALLOW | LOG | PREVENT)
Specifies how eTrust CA-ACF2 reacts when a user with this field enters a command that results in a "spool file not found" condition. The values for this field are listed in the following:

- **PREVENT**—rejects and logs the command
- **LOG**—passes the command to CP for normal syntax checking and generates a logging record
- **ALLOW**—passes the command to CP for normal syntax checking
- **null**—removes the NOSPOOL field from the user's logonid record.

Null, the default value causes the @VM NOSPOOL setting in the ACFFDR to take effect for "spool file not found" conditions. If you specify a value other than null, the setting for this field overrides the @VM NOSPOOL setting in the ACFFDR. This field applies to VM sites only.

NOTICES | NONOTICES
Indicates a user can receive TSO notices at logon time. (Bit field)

SYSPEXCL | NOSYSPEXCL
Indicates that when the system is active in a sysplex environment, this logonid record should not be written to the structure.

***OPERATOR | NOOPERATOR**
Indicates that a user has TSO operator privileges. (Bit field)

PASSWORD(*password*)
Contains the one to eight-character password of the logonid. The password is never displayed. Every logonid on the database must have a password defined for it. You can specify a password using the CHANGE subcommand to reset or change the password for a logonid.

eTrust CA-ACF2 stores the password in a one-way encrypted format. For more information about the encryption algorithm, see "System Requirements" in the *Getting Started* guide. (Eight characters)

***PAUSE | NOPAUSE**
Indicates that a user wants a program to pause when a multilevel message is issued by a command executed in a CLIST. This lets the user enter a question mark to receive the second-level messages. (Bit field)

PGM(*program*)
Specifies a one- to eight-character program name or a mask. The specified program must be used to submit jobs for this logonid. If the user has SUBAUTH, this program must be APF-authorized. Proper use of this program-pathing facility also requires that this logonid be defined with the RESTRICT attribute. This field can also be specified by using the field name PROGRAM instead of the name PGM, except for reports that require the full form of the name (PROGRAM). This is a duplicate of the PROGRAM field. (8 characters)

Two symbols can be used in the PGM field of the LOGONID to signify masking, the asterisk (*) and the dash (-). A dash represents all programs that begin with the specified characters that precede the dash or all programs if the dash is used alone. An asterisk represents one or more masking or wild card characters that can be specified anywhere in the in program.

PHONE(*phonenumber*)
Specifies the one to 12-character telephone number of a user. (12 characters)

***PMT-ACCT | NOPMT-ACCT**
Indicates that eTrust CA-ACF2 requires a user to specify an account number at logon time. Specify the LGN-ACCT field also. eTrust CA-ACF2 does not prompt for an account number if you also specify the FSRETAIN field. FSRETAIN obtains the account values from the last session. (Bit field)

***PMT-PROC | NOPMT-PROC**
Indicates that eTrust CA-ACF2 requires a user to specify a TSO procedure name at logon time. Specify the LGN-PROC field also. eTrust CA-ACF2 does not prompt for a procedure name if you also specify the FSRETAIN field. FSRETAIN obtains the procedure name from the last session. (Bit field)

PPGM|NOPPGM
Specifies that a user can execute the protected programs specified in the GSO PPGM record. For more information, see Displaying GSO PPGM Record Information in the "Maintaining Global System Options Records" chapter. (Bit field)

PP-TRC|NOPP-TRC
Specifies whether eTrust CA-ACF2 creates SMF loggings that contain the Active Library List for all data set access attempts made by this logonid in a batch job. PP-TRC is a tool to help in the coding of program-pathed data set access rules. You can view these loggings in the Data Set Report Log (ACFRPTDS) report generator. For details on program pathing trace records collected for the report generator, see the *Reports and Utilities Guide*. For details on program pathing and the Active Library List, see The Active Library List in the "Maintaining Access Rules" chapter of this guide. PP-TRC is only valid for Batch jobs. (Bit field)

PP-TRCV|NOPP-TRCV
Specifies whether eTrust CA-ACF2 creates SMF loggings that contain the Active Library List for all data set access violations made by this logonid in a batch job. PP-TRCV is a tool to help in the coding of program-pathed data set access rules. You can view these violations in the Data Set Report Log (ACFRPTDS) report generator. For details on program pathing violation trace records collected for the report generator, see the *Reports and Utilities Guide*. For details on program pathing and the Active Library List, see The Active Library List in the "Maintaining Access Rules" chapter. PP-TRCV is only valid for Batch jobs. (Bit field)

PREFIX(*prefix*)
Specifies the zero- to eight-character key of the rule used to validate access to a data set. Specifying a key in the PREFIX field indicates that the user owns that key and therefore automatically has access to that data unless the RULEVLD attribute is specified in the user's logonid record. RULEVLD takes precedence over data set ownership denoted by the PREFIX field. (See description for RULEVLD field.)

The PREFIX field also identifies the key of the access rule set that a user can store and decompile. If the PREFIX field is null, then any data set access that a user makes must be specified in an access rule. The PREFIX field can contain a high-level index mask with asterisks (*), but not with a dash (-).

To set a default prefix for a user, specify a value (usually the user's logonid) in the DFT-PFX field. TSO places the value in the DFT-PFX field in front of data set names specified by the user unless the user qualifies the data set name by placing it in quotes. The CENTRAL field of the GSO RULEOPTS record determines whether users can decompile and store the access rules for the data sets that they own. For details on the GSO RULEOPTS record, see the "Maintaining Global System Options Records" chapter in this guide. (Eight characters)

PRIV-CTL|NOPRIV-CTL
Indicates that the logonid is valid for dynamic logonid privileges. This does not include STC. When the user accesses the system (system entry), a special set of logonid privilege resource rules are checked to determine if the logonid should be assigned some dynamic privileges or authorities. See Providing Dynamic Logonid Privileges in the chapter entitled, "Controlling System Entry," for more information about dynamic logonid privileges. (Bit field)

PROGRAM(*program*)
Specifies a one to eight-character program name or name mask. The specified program must be used to submit jobs for this logonid; if the logonid has SUBAUTH, this program must be APF-authorized. Proper use of this program-pathing facility also requires that this logonid be defined with the RESTRICT attribute. This field can also be specified by using the field name PGM instead of the name PROGRAM, except for reports that require the full form of the name (PROGRAM). (Eight characters)

***PROMPT|NOPROMPT**
Indicates that eTrust CA-ACF2 prompts a user for missing or incorrect parameters. (Bit field)

PRVPSWD1 through PRVPSWD4
Contains password history information if password history is implemented with the PSWDHST option of the GSO PSWD record or with Note 12. If you implement password history, eTrust CA-ACF2 checks and prevents the use of the four previously entered passwords. PRVPSWD1 is the most recently used password and PRVPSWD4 is the password that was used four cycles ago. These fields are used with the PRV-TOD1 through PRV-TOD4 fields and cannot be specified or changed. For information about the PSWDHST option, see Password Maintenance and Support (PSWD) in the "Maintaining Global System Options Records" chapter of this guide. For information about eTrust CA-ACF2 Note 12, see the "eTrust CA-ACF2 Notes" appendix of the *Systems Programmer Guide*. (Eight characters)

PRV-TOD1 through PRV-TOD4
Specifies the date and time that passwords were changed. These fields are used with the PRVPSWD1 through PRVPSWD4 fields to provide the date and time the passwords were changed if you have implemented password history. These fields cannot be specified or changed. PRV-TOD1 is the date and time PRVPSWD1 was changed, PRV-TOD2 corresponds to PRVPSWD2, and so on. The date is displayed in the format specified in the DATE field of the GSO OPTS record, mm/dd/yy, dd/mm/yy, or yy/mm/dd. Year designations of 70-99 assume a date in the 20th century (1970-1999); year designations of 00-69 assume a date in the 21st century (2000-2069). (Four-byte binary field)

PSWD-DAT(*date*)
Specifies the date of the last invalid password attempt. The date is displayed in the format mm/dd/yy, dd/mm/yy, or yy/mm/dd, depending on the DATE field of the GSO OPTS record. Year designations of 70-99 assume a date in the 20th century (1970-1999); year designations of 00-69 assume a date in the 21st century (2000-2069). (Four-byte packed field)

PSWD-EXP|NOPSWD-EXP
Indicates that a user's password has been manually expired (forced to expire). This field lets a security administrator force this user to change his password. (Bit field)

PSWD-INV(*nn*)
Specifies the number of password violations that occurred since the last successful logon. This field can be reset to zero by a security administrator. (Two-byte binary field)

PSWD-MIX
Specifies that the current password is case sensitive. You cannot set this field. eTrust CA-ACF2 maintains and displays it. (Bit field)

PSWD-SRC(*sourceid*)
Specifies the one to eight-character logical or physical input source name or source group name that a user last entered an invalid password from. You cannot set this field. eTrust CA-ACF2 maintains and displays it. (Eight characters)

PSWD-TIM(*hh:mm*)
Specifies the last time a user entered an invalid password. You cannot set this field. eTrust CA-ACF2 maintains and displays it. (Four-byte binary field)

PSWD-TOD(*date*)
Specifies the date and time when a user changed his password. eTrust CA-ACF2 lists the date in the format mm/dd/yy, dd/mm/yy, or yy/mm/dd, depending on the DATE field of the GSO OPTS record. You cannot set this field. eTrust CA-ACF2 maintains and displays it. Year designations of 70-99 assume a date in the 20th century (1970-1999); year designations of 00-69 assume a date in the 21st century (2000-2069). (Eight-byte binary field)

PSWD-VIO(*nn*)
Specifies the number of password violations that occurred on PSWD-DAT. (Two-byte binary field)

PSWD-XTR|NOPSWD-XTR
Specifies that the password for the logonid is halfway encrypted and can be extracted by an APF-authorized program. eTrust CA-ACF2 automatically sets this bit if the logonid changes its password while the PSWDXTR field is set in the GSO PSWD record. It cannot be set using the CHANGE subcommand.

The bit can be turned off by specifying NOPSWD-XTR, using the CHANGE subcommand. The change erases the halfway-encrypted password. NOPSWD-XTR is the default. (Bit field)

PTICKET | NOPTICKET
Specifies that a passticket can be used with a userid that has the RESTRICT attribute.

READALL | NOREADALL
Specifies that a user has read and execute access to all data sets at the site. This is similar to the NON-CNCL attribute, but grants read and execute access only. eTrust CA-ACF2 enforces any existing rules for other types of access such as write and allocate. eTrust CA-ACF2 logs any accesses the user makes that are not allowed through ownership or through rules. READALL also bypasses PDS member-level validation. (Bit field)

***RECOVER | NORECOVER**
Indicates that a user can specify the recover option of the TSO or TSO/E Command Package. The PROFILE RECOVER option is set on at logon time. (Bit field)

REFRESH | NOREFRESH
Specifies that a user can issue the F ACF2,REFRESH operator command. The F ACF2,REFRESH command lets you implement changes to global system options in your system. (Bit field)

RESTRICT | NORESTRICT
Specifies that a logonid is for production use only. A restricted logonid does not require a password for user verification. eTrust CA-ACF2 logs all jobs submitted by restricted logonids, except for jobs submitted by those jobs.

To create loggings for jobs submitted by jobs that are submitted by a restricted logonid, specify the //*LOGONID or USER= job statement in the spun jobs. When eTrust CA-ACF2 reads the restricted logonid it will validate the access by the restricted logonid and create a logging record. You can display these loggings using the Restricted Logonid Job Log report, ACFRPTJL. Do not specify this field for online user logonids.

You should also specify this field for the DFTLID and DFSYSOUT that you specify in the GSO NJE record. For details on the GSO NJE record, see the "Maintaining Global System Options Records" chapter in this guide.

Do not specify this field with the STC field. (Bit field)

RSRCVLD | NORSRCVLD
Specifies that a resource rule must authorize any resource accesses that a user makes. This field applies even if a user has the SECURITY privilege. (Bit field)

RULEVLD|NORULEVLD
Specifies that an access rule must authorize any data set accesses that a user makes. This field applies even if a user has ownership of the data or has the SECURITY privilege. RULEVLD does not apply if a user has READALL or NON-CNCL access. READALL and NON-CNCL access takes precedence over RULEVLD. (Bit field)

SCPLIST(*scpname*)
Specifies the eight-character name of the scope record that restricts accesses for this privileged user (for example, a user with SECURITY or ACCOUNT). This field is not maskable. (Eight characters)

SEC-VIO(*nn*)
Indicates the number of cumulative security violations for a user. (Two-byte binary)

SECURITY|NOSECURITY
Specifies that a user is a security administrator who can:

- Access all data sets, protected programs, and resources
- Maintain rules and all infostorage records
- Display logonid records
- Change certain fields in logonid records

Unless the RULEVLD or RSRCVLD attribute is specified in the user's logonid record. RULEVLD and RSRCVLD take precedence over any privileges granted by the SECURITY field. (See description for RULEVLD and RSRCVLD fields.) eTrust CA-ACF2 logs any accesses the security administrator makes that are not allowed through ownership or through rules. You can scope the SECURITY privilege. (Bit field)

SHIFT(*shiftname*)
Specifies the one to eight-character name of a shift record that defines the period during which a user can log on to the system. The shift record specifies the times of day, dates, and days for access. You cannot mask the SHIFT field.

You can specify the LOGSHIFT field to let a user access the system during periods outside those defined in a shift record. eTrust CA-ACF2 logs these accesses and you can display them using the ACFRPTPW report. (Eight characters)

SMSINFO(*recid*)
Specifies the one to eight-character name of a CONTROL(SMS) record that contains the default storage management class values for a user. (Eight characters)

SOURCE(*sourceid*)
Specifies the one to eight-character logical or physical input source name or source group name from which a user must access the system. Specify a physical ID, an input source name that you define in an E(SRC) record, a source group name that you define in an E(SGP) record, or the source group that you define in the X(SGP) record. You can mask the SOURCE field. (Eight characters)

SRF|NOSRF
Specifies that a user can use the system request facility (SRF) in a VM environment. This field is meaningful to VM sites. It is specified on the logonid record for sites using shared database support for z/OS and VM. (Bit field)

STC|NOSTC
Specifies that a logonid is for use by started tasks only. eTrust CA-ACF2 denies access to started tasks without this privilege; likewise, it prevents logonids with this attribute from submitting jobs or logging on to TSO.

You do not need to specify the RESTRICT, MON-LOG, or MONITOR field and the STC field. eTrust CA-ACF2 does not check for these fields for logonids with the STC privilege.

eTrust CA-ACF2 does not create a special logging record when a logonid with the STC attribute enters the system, unlike the situation for logonids with the RESTRICT privilege. Run the ACFRPTLL report and specify the UPDATE parameter to monitor the use of a logonid with the STC privilege. (Bit field)

SUBAUTH|NOSUBAUTH
Indicates that jobs that specify this logonid can be submitted only through APF-authorized programs. You must also specify the RESTRICT field for this logonid. Most sites also specify the PROGRAM or PGM fields for logonids with the SUBAUTH privilege to indicate that this logonid can submit jobs using a specified program. (Bit field)

SUSPEND|NOSUSPEND
Indicates that a user cannot enter this logonid to access the system. See the description of the CANCEL field earlier in this section. (Bit field)

SYNCNODE(*nodeid*)
Specifies the name of the node where the synchronized logonid for a user resides. This node name is the logical node name as defined by a NETNODE record. eTrust CA-ACF2 displays blanks in this field to indicate a synchronized logonid. (Eight characters)

SYNERR(null|ALLOW|LOG|PREVENT)
Specifies the action eTrust CA-ACF2 takes when a user enters a command that results in a command syntax error. The values for this field are described in the following:

- **null**—removes the SYNERR field from the user's logonid record. This is the default.
- **ALLOW**—passes the command to CP for normal syntax checking.
- **LOG**—passes the command to CP for syntax checking and generates a logging record.
- **PREVENT**—rejects and logs the command.

If you specify any value but null, eTrust CA-ACF2 overrides the SYNERR operand of the command model COMMAND clause and the @VM SYNERR setting in the ACFFDR. Null is the default value. This field applies to VM sites only.

TAPE-BLP|NOTAPE-BLP
Specifies that a user can use full bypass label processing (BLP) when accessing tape data sets. BLP lets a user specify any data set name or volume name in the JCL without any comparison of the information with the tape label. If a user specifies BLP, eTrust CA-ACF2 validates that they have the privilege to permit the job to run. In validating tape access, eTrust CA-ACF2 performs this validation based on the data set name and/or volume name coded in the JCL. You should tightly control this privilege. (Bit field)

TAPE-LBL|NOTAPE-LBL
Specifies that a user has limited bypass label processing authority when using tapes. When a user with this privilege requests a BLP request, eTrust CA-ACF2 checks the actual volume serial number written on the tape label. If the actual volser is available from the tape, it will override the one specified in the JCL. The data set name used is the one from the JCL. The actual tape data set validation depends on the TAPEDSN field of the GSO OPTS record and whether the volser is specified in the GSO SECVOLS record. (Bit field)

TDISKVLD
Indicates that access rules must exist for all data on temporary disks that this user accesses. TDISKVLD is a method that lets you control which files a user can write to or read from on his own T-disks to create a "padded cell" environment. For TDISKVLD to be effective, a user cannot change his own access rule. A special access rule syntax is required for files on T-disks. This field applies to VM users only. (Bit field)

TRACE|NOTRACE
Specifies whether eTrust CA-ACF2 creates SMF loggings for all data set and resource access attempts made by this user. You can view these access attempts in the Data Set Report Log (ACFRPTDS) and Resource Logging (ACFRPTRV) report generators. For details on trace records collected for these two report generators, see the *Reports and Utilities Guide*. (Bit field)

TSO|NOTSO
Specifies that a user can log on to TSO. You can decide whether eTrust CA-ACF2 should check the user's TSO logon authorization. For information about this option, see the description of the LOGONCK field of the TSO record. (Bit field)

TSO-TRC|NOTSO-TRC
Specifies whether eTrust CA-ACF2 traces all TSO commands issued by this user for the Command Statistics Report (ACFRPTCR). For more information, see the description of the ACFRPTCR report generator in the *Reports and Utilities Guide*. (Bit field)

***TSOACCT(*account*)**
Specifies the user's default TSO logon account. (40 Characters)

TSOCMDS(*module*)
Specifies the name of a TSO command list module that contains the list of commands that this user is authorized to use. You cannot mask this field. Command limiting is effective for all logonids including privileged ones. It takes place in all modes with the exception of QUIET. (Eight characters)

***TSOFSCRN|NOTSOFSCRN**
Indicates that a user can use the full-screen logon display. (Bit field)

***TSOPERF(*grp*)**
Indicates the user's default TSO performance group (1-255). Zero indicates no performance group was specified. (One-byte binary field)

***TSOPROC(*procname*)**
Indicates a user's default TSO procedure name. (Eight characters)

***TSORBA(*pointer*)**
Indicates the Mail Index Record Pointer (MIRP) for this user. This pointer is for use only with TSO/E. eTrust CA-ACF2 picks up the RBA from the PSCB if it has been updated and automatically sets this value when the user signs off. (Three-byte hexadecimal field)

***TSORGN(*size*)**
Indicates a user's default TSO region size (up to 2,147,483,648K) if no size is specified at logon time. (Four-byte binary field)

***TSOSIZE(*size*)**
Specifies a user's maximum TSO region size (up to 2,147,483,648K), unless LGN-SIZE is specified for that user. (Four-byte binary field)

***TSOTIME(*time*)**
Specifies a user's default TSO time parameter. (Two-byte binary field)

***TSOUNIT(*unitname*)**
Indicates a user's default TSO unit name. (Eight characters)

UID(*character-string*)
Indicates the one to 24-character pseudo field (not actually stored in the logonid record) that contains the user identification string (UID) for a user. This field is a concatenation of selected logonid fields that might include site-defined fields such as department and job function. You define these fields in the @UID macro of the eTrust CA-ACF2 Field Definition Record (ACFFDR). (24 characters)

UNICNTR | NOUNICNTR
Indicates that this user also resides on the CA-Common Services platform. When set, CPF lets commands referring to this logonid be sent to nodes defined to CPF as UNINODEs. (Bit field)

UPD-TOD(*date-time*)
Indicates the date and time that a logonid record was last updated. eTrust CA-ACF2 displays the date in the format mm/dd/yy, dd/mm/yy, or yy/mm/dd, depending on the DATE field of the GSO OPTS record. You cannot change this field. (Eight-byte binary field)

USER | NOUSER
Indicates that a user can access the system. All logonids defined to eTrust CA-ACF2 have this privilege. eTrust CA-ACF2 never displays this field. You should not alter it.

VLD-ACCT | NOVLD-ACCT
Indicates that eTrust CA-ACF2 validates the TSO account number of a user. You must create a resource rule with a type code TAC and a $KEY of the account number so that eTrust CA-ACF2 performs this validation. (Bit field)

VLD-PROC | NOVLD-PROC
Indicates that eTrust CA-ACF2 validates the TSO procedure name of a user. You must create a resource rule with a type code TPR and a $KEY of the procedure name so that eTrust CA-ACF2 performs this validation. (Bit field)

VLDRSTCT | NOVLDRSTCT
Indicates that PROGRAM and SUBAUTH are to be validated even when this RESTRICTed logonid is inherited. (Bit field)

VLDVMACT | NOVLDVMACT
Specifies that a virtual machine must specify an account number if its VMACCT field contains blanks. eTrust CA-ACF2 prevents virtual machines without the VLDVMACT field whose VMACCT field contains blanks from using the system. eTrust CA-ACF2 performs account resource validation for machines with the VLDVMACT field. This field applies to VM users only. (Bit field)

VM | NOVM
Indicates that a user can log on to VM. This field is meaningful for sites that have eTrust CA-ACF2 VM and use synchronized database support. (Bit field)

VMACCT | NOVMACCT
Specifies an eight-byte logonid field that holds the default account number for a virtual machine. The VMACCT values work with account resource rules to functionally replace the VM/XA ACCOUNT directory control statements. This field applies to VM users only and is available for VM/XA operating systems. (Character field)

VMD4AUTH
A user with this attribute can issue diagnose d4 to surrogate virtual machines with the VMD4TARG attribute. Use **extreme caution** when you assign this privilege. The VMD4TARG and VMD4AUTH privileges are very powerful. A typical class B user with both of these attributes could potentially surrogate itself to any user ID on the system and have access to anything on the system. In previous releases, this privilege was the VMBATMON privilege. This field applies to VM users only.

VMD4RSET
Indicates that this user can be the target of the diagnose d4 reset after the logonid was surrogated to another ID. Use **extreme caution** when assigning this privilege. Never give this logonid attribute to a batch worker machine.

The combination of VMD4RSET, VMD4AUTH, and VMD4RSET lets products like TCP/IP and VMBACKUP function properly. To track the use of the diagnose d4, you can write a diagnose limiting rule to log each time the diagnose d4 is issued. In previous releases of eTrust CA-ACF2 VM (r3.2 and below), this attribute was called VMRESET. This field applies to VM users only.

VMD4TARG
A user ID with this attribute can be the target of diagnose d4 (the alternate user diagnose). Use **extreme caution** when you assign this attribute. In previous releases, this privilege was the VMBATCH privilege. This field applies to VM users only.

VMIDLEMN(*nnn*)
Specifies the number of minutes (from 1 to 240) that this user can be idle on the system before the idle terminal processing begins. This value overrides the system-wide IDLEMN value defined in the OPTS VMO record. This field applies to VM users only.

VMIDLEOP(OFF | DISC | LOGOFF | REPROMPT)
Specifies the type of idle terminal processing to perform when this user exceeds the idle time limit. This value overrides the system-wide IDLEOP VMO record. Values for this field are:

- **OFF**—Disables idle terminal processing for this user.
- **DISC**—Forces disconnection from the system when this user exceeds the idle terminal limit.

- **LOGOFF**—Forces this user off the system when he exceeds the idle terminal limit.
- **REPROMPT**—Prompts the user for his password when he exceeds the idle terminal limit. Incorrect passwords are counted as password violations. The user can also logoff or disconnect from the system at this prompt.

This field applies to VM users only.

VMSAF|NOVMSAF
Indicates this logonid can issue the diagnose code, DIAG 'A0' subfunction code '04'. This support lets users validate eTrust CA-ACF2 passwords from their unique applications. This field applies to VM users only. (Bit field)

VMXA|NOVMXA
Permits this user to log on to the VM/ESA system. This attribute is required only if the VMCHK=VMXA operand has been specified in the @VM macro. This field applies only to VM users. (Bit field)

VSESRF|NOVSESRF
Indicates that this logonid can issue System Request Facility (SRF) requests to the eTrust CA-ACF2 VM service machine from eTrust CA-ACF2 VSE. These SRF requests can validate the accesses of users and perform direct maintenance of the eTrust CA-ACF2 databases. This field applies only to VM and VSE users. (Bit field)

***WTP|NOWTP**
Indicates that eTrust CA-ACF2 displays write-to-programmer messages. eTrust CA-ACF2 issues all violation and warning messages as WTPs. Specify this field for all TSO user logonid records so that they can receive eTrust CA-ACF2 messages. (Bit field)

ZONE(*zone*)
Specifies the three-character name of the zone record that defines the time zone from which this logonid normally accesses the system (that is, the user's local time zone). You cannot mask this field. (Three characters)

Logonid Record Sections

eTrust CA-ACF2 groups the fields of the logonid record into sections. Each section contains information that eTrust CA-ACF2 uses to determine what a user can access:

Section Number	Section Name	Description
0	Identification header	Provides the user's logonid, name, phone, and UID.
1	Cancel/Suspend	Indicates whether the logonid is canceled or suspended, and the date this action was taken.
2	Privileges	Indicates what the logonid is permitted to do. Can it access CICS, IMS, or TSO? Can it submit batch jobs? Can it maintain eTrust CA-ACF2 records?
3	Access	Provides statistics on the number of system accesses that a user makes, the date, time, and source of the logonid's last access.
4	Password	Provides statistics on violation count and date, expiration date, and change date for the password.
5	TSO	Provides TSO data such as the logonid's TSO account number, performance group, region size.
6	Statistics	Indicates the cumulative number of security violations and the date and time the logonid was last updated.
7	CICS	Contains information about the eTrust CA-ACF2 CICS Security subsystem.
8	IMS	Contains information about the eTrust CA-ACF2 IMS Security subsystem.
9	IDMS	Contains information about the eTrust CA-ACF2 IDMS Security subsystem.
10	MUSASS	Contains information about the fields used to define processing by the MUSASS.
11	Restrictions	Provides data on access to data and conditions for logon, such as time of day, time zone, and location.
12	DFP	Provides information about the IBM Data Facility Product facilities

These sections are provided with eTrust CA-ACF2. You can rearrange their locations, move the fields to another section, or define your own sections. For more information, see the *Getting Started* guide.

Logonid Record Fields/Sections List

This alphabetical list shows each field of the logonid record and its corresponding section number.

ACC-CNT(3)
ACC-DATE(3)
ACC-SRCE(3)
ACC-TIME(3)
ACCOUNT(2)
ACCTPRIV(5)
ACF2CICS(7)
ACTIVE(2)
ALLCMDS(5)
ATTR2(5)
AUDIT(2)
AUTHSUP1(11)
AUTHSUP2(11)
AUTHSUP3(11)
AUTHSUP4(11)
AUTHSUP5(11)
AUTHSUP6(11)
AUTHSUP7(11)
AUTHSUP8(11)
AUTOALL(2)
AUTODUMP(2)
AUTONOPW(2)
AUTOONLY(2)
BDT(2)
CANCEL(1)
CICSCL(7)
CICSID(7)
CICSKEY(7)
CICSKEYX(7)
CICSOPT(7)
CHAR(5)
CICS(2)
CICSPRI(7)
CICSRSL(7)
CMD-LONG(5)
CMD-PROP(2)
CONSOLE(2)
CONSULT(2)
CSDATE(1)
CSWHO(1)
DFT-DEST(5)
DFT-PFX(5)
DFT-SOUT(5)
DFT-SUBC(5)
DFT-SUBH(5)
DFT-SUBM(5)
DG84DIR(2)
DIALBYP(2)
DUMPAUTH(2)
EXPIRE(2)
GROUP(11)
GRPLOGON(2)
GRP-OPT(2)
GRP-USER(3)
HOMENODE(0)
IDLE(7)
IDMS(2)
IDMSPROF(9)
IDMSPRVS(9)
IMS(2)
INTERCOM(5)
JCL(5)
JOB(2)
JOBFROM(2)
KERB-VIO(4)
LDS(2)
LDEV(2)
LEADER(2)
LGN-ACCT(5)
LGN-DEST(5)
LGN-MSG(5)
LGN-PERF(5)
LGN-PROC(5)
LGN-RCVR(5)
LGN-SIZE(5)
LGN-TIME(5)
LGN-UNIT(5)
LID(0)
LIDZMAX(4)
LIDZMIN(4)
LINE(5)
LOGSHIFT(2)
MAIL(5)
MAINT(2)
MAXDAYS(4)
MINDAYS(4)
MODE(5)
MON-LOG(1)
MONITOR(1)
MOUNT(5)
MSGID(5)
MUSASS(2)
MUSDLID(8)
MUSID(10)
MUSIDINF(10)
MUSOPT(10)
MUSPGM(10)
MUSUPDT(10)
NAME(0)
NO-INH(2)
PSWD-EXP(4)
NOMAXVIO(2)
NO-OMVS(2)
NO-SMC(2)
NO-STATS(10)
NO-STORE(2)
NON-CNCL(2)
NOSPOOL(2)
NOTICES(5)
OPERATOR(5)
PASSWORD(0)
PAUSE(5)
PGM(2)
PHONE(0)
PMT-ACCT(5)
PMT-PROC(5)
PPGM(2)
PP-TRC(1)
PP-TRCV(1)
PREFIX(11)
PRIV-CTL(2)
PROGRAM(2)
PROMPT(5)
PRVPSWD1(0)
PRVPSWD2(0)
PRVPSWD3(0)
PRVPSWD4(0)
PRV-TOD1(0)
PRV-TOD2(0)
PRV-TOD3(0)
PRV-TOD4(0)
PSWD-DAT(4)
PSWD-INV(4)
PSWD-MIX(4)
PSWD-SRC(4)
PSWD-TIM(4)
PSWD-TOD(4)
PSWD-VIO(4)
PSWD-XTR(4)
PTICKET(2)
READALL(2)
RECOVER(5)
REFRESH(2)
SOURCE(11)
SRF(2)
RESTRICT(2)
RSRCVLD(2)
RULEVLD(2)
SCPLIST(2)
SEC-VIO(6)
SECURITY(2)
SHIFT(11)
SMSINFO(12)
STC(2)
SUBAUTH(2)
SUSPEND(1)
SYNCNODE(2)
SYNERR(2)
SYSPEXCL(2)
TAPE-BLP(2)
TAPE-LBL(2)
TDISKVLD(2)
TRACE(1)
TSO(2)
TSO-TRC(1)
TSOACCT(5)
TSOCMDS(5)
TSOFSCRN(5)
TSOPERF(5)
TSOPROC(5)
TSORBA(5)
TSORGN(5)
TSOSIZE(5)
TSOTIME(5)
TSOUNIT(5)
UID(0)
UNICNTR
UPD-TOD(6)
USER(2)
VAX(2)
VLD-ACCT(5)
VLD-PROC(5)
VLDRSTCT(2)
VLDVMACT(2)
VM(2)
VMACCT(11)
VMD4AUTH(2)
VMD4RSET(2)
VMD4TARG(2)
VMIDLEMN(11)
VMIDLEOP(11)
VMSAF(2)
VMXA(2)
VSESRF(2)
WTP(5)
ZONE(11)

USER Profile Records

USER profile records are associated with a user of the system. Profile data information segments that can be extracted for a user include CERTDATA, CICS, DCE, KEYRING, KERB, KERBLINK, LANGUAGE, LINUX, LNOTES, NDS, NETVIEW, OMVS, OPERPARM, SECLABEL, PASSWORD, and WORKATTR.

The key of a USER profile record is a userid, which can be masked. If you use masking, a directory must be built for the USER profile records. The asterisk (*) is the only valid masking character. Masking for the USER profile record follows the same rules as masking for resource rules. A dash is not allowed in the $KEY as a masking character. If a dash is used in the $KEY, it is treated as a literal dash.

User profile records can also be administered in LID setting as if the user profile information is part of the logonid record. User profile records can be created, changed, displayed, and deleted with a logonid request. eTrust CA-ACF2 distinguishes the particular logonid or user profile records to process and performs the appropriate administrative request. User profile records like CERTDATA and KEYRING records, which might contain suffixes, cannot be inserted or changed in LID setting. Because a user profile record with a suffix cannot be distinguished in LID mode, the records must be directly inserted in user profile administration mode.

For all user profile records, you must use the SET command to enter:

```
SET PROFILE(USER) DIVISION(profile_data_name)
```

The LIST command will only display the currently active set of profile records.

Note: If you insert or change a user profile record and it is resident, then you must issue the REBUILD command for those changes to be activated.

```
F ACF2,REBUILD(USR),CLASS(P)
```

CERTDATA Profile Data Records

The CERTDATA segment of the USER profile identifies an X.509 digital certificate associated with a user. Digital certificates provide a means of authentication through the use of public-key cryptography and a trusted third party, known as a Certification Authority. Each certificate is identified uniquely by its serial number and associated certification authority distinguished name ("issuer's distinguished name").

As an alternative to requesting userid and password information, a z/OS Web server can authenticate users based on their digital certificates. The Web server performs all certificate authentication. If z/OS resources are accessed, the certificate is presented to eTrust CA-ACF2. Using the certificate serial number and the issuer's distinguished name, eTrust CA-ACF2 associates the certificate to a z/OS or OS/390 userid. A z/OS security environment is then created for that user. By using authenticated certificates, passwords are not sent through the network.

Certificates are associated to eTrust CA-ACF2 z/OS or OS/390 users through the use of profile records. A user can have more than one certificate, but a single certificate cannot be used by more than one user. Profile records are inserted using the userid as the record key with or without a specified label (if no label is specified, one is set by default). The record key can also contain a userid with a distinguished suffix if a user is defined with more than one certificate.

Record ID	Fields
Recid *logonid* | *certauth* | *sitecert* | *logonid.suffix* | *certauth.suffix* | *sitecert.suffix*	ACTIVE(*date*) CERTID(*Serial #.IDN*) CERTNSERser(*hex*) DSN(*dsn*) EXPIRE(*date*) ICSF ISSUERDN(*dn*) LABEL(*label*) NEWLABELewlabel(*label*) PASSWORDassword(*password*) SERIAL#(*serial-number*) HITRUST | TRUST | NOTRUST

Field Descriptions

*recid***(***Logonid* | *certauth* | *sitecert* | *logonid.suffix* | *certauth.suffix* | *sitecert.suffix***)**
Specifies the userid that is to be associated with the certificate. It might be a one- to eight-character logonid, or a logonid, a period, and a one- to eight-character suffix. If label is also specified, logonid, rather than logonid.suffix, must be specified, and indicates the logonid that the label is associated with.

Using CERTAUTH in place of a logonid indicates that the certificate is a Certification Authority certificate.

Using SITECERT in place of a logonid indicates that the certificate is a site certificate.

Active(*date*)
Specifies an optional activation date in the format "mm/dd/yy". This date is not the same as the not-before validity date in the certificate itself. This date gives the security administrator the ability to specify when the profile record associating the user to the certificate becomes active. This date must fall within the range of the certificate's not-before and not-after validity dates and must be earlier than the CERTDATA record expiration date, if one exists.

CERTID(*Serial #.IDN*)
Specifies the serial number and certification authority's distinguished name as extracted from the certificate. This field is displayed only and cannot be altered.

CERTNser(*hex value*)
Indicates the next serial number to be used by this certificate when signing another certificate. This field generally should not be modified by hand because you can generate duplicate certificates. All detected duplicate certificates will be unusable on z/OS systems.

Dsn(*dsn*)
Specifies the z/OS dataset that contains the digital certificate that is inserted into a CERTDATA profile record. The data set must be defined as physical sequential (DSORG=PS) and variable-blocked (RECFM=VB) and must be catalogued. If the data set name is enclosed in single quotes, it is considered fully qualified and is used as specified. Otherwise, the user's prefix, as specified by the TSO PROFILE PREFIX command (or defaulted from the DFT-PFX field of the logonid record) is added to the front of the data set name.

You may now insert all of the certificates contained in a PKCS7 certificate package. A PKCS 7 certificate package contains a user certificate and a chain of CA certificates. When a certificate is inserted from a data set and the DSN parameter contains the name of a PKCS 7 certificate package, each of the CA certificates will be added to the database from the highest CA certificate in the chain to the lowest certificate in the chain. The trust status of the first CA certificate added will take the value specified on the insert command. The other CA certificates that are added will take the trust value of the signing certificate. If a certificate is expired or it's validity period is not complete within the validity of it's signing certificate or if the signing certificate of the certificate is not in the PKCS 7 package or in eTrust CA-ACF2, then the certificate is added with a trust status of NOTRUST. If the a CA certificate is already known to eTrust CA-ACF2, the certificate will retain it's trust status. A label of AUTOxxx will be generated for each CA certificate added where xxx is an available number between 0 and 1000.

If an error occurs during the addition of certificates from a PKCS 7 certificate package, any CERTAUTH certificates already added will not be removed.

Expire(*date*)
Specifies an optional expiration date in the format "mm/dd/yy" This date is not the same as the not-after validity date in the certificate itself. This date gives the security administrator the ability to specify when the profile record associating the user to the certificate expires. This date must fall within the range of the certificate's not-before and not-after validity dates and must be later than the CERTDATA record activation date, if one exists.

ICSF
Specifies the private key for the certificate is placed in ICSF. This parameter is not valid on change requests. It is a valid parameter in GENCERT and INSERT commands. If ICSF is specified on an insert command and the certificate is to be renewed and the private key exists in the eTrust CA-ACF2 database, the private key will be moved from the eTrust CA-ACF2 database into ICSF.

ICSF must be active and configured for PKA operations. If it is not an error message will be displayed when attempting to insert or use the private key.

ISSUERDN(*dn*)
Specifies that the ISSUERDN is the certification authority's distinguished name as extracted from the certificate. This field cannot be altered or displayed. However, it can be used with the SERIAL# operand to list, change, or delete list or change a CERTDATA record. **Note:** ISSUERDN cannot be abbreviated.

LABEL(*label*)
Specifies a 32-character label to be associated with the certificate. The label can contain blanks and mixed case characters. If a label is not specified, the label field will default to the upper-case version of the logonid that was specified. To change the label in a record, the NEWLABEL operand must be used. **Note:** Label Label cannot be abbreviated.

NEWLABELewlabel(*label*)
Specifies the 32-character label, which is to replace an existing label associated with a certificate. The label can contain blanks and mixed case characters. Newlabel NEWLABEL can only be specified on a CHANGE command. If LABEL rather than NEWLABEL is encountered in the command input, a CHANGE using LABEL request is assumed to be in effect.

Password(*password*)
Specifies the password used to encrypt the PKCS #12 certification package. This password does not conform to normal eTrust CA-ACF2 password syntax and may be mixed case and up to 255 bytes in length. Note that eTrust CA-ACF2 only supports PKCS #12 certificates that adhere to the PKCS #12 v1.0 standard published by RSA. These certificates are defined with a 3 in the version number of the PKCS #12 certificate package.Specifies the 255-character (or less) password associated with a PKCS #12 certificate. This password must be the same as the password that was specified when the certificate was exported. A password can only be specified with a PKCS #12 certificate.

SERIAL#(*serial_number*)

Specifies the serial number as extracted from the certificate. This field cannot be altered or displayed. However, it can be used with the ISSUERDN operand on a change or list command to list, change, or delete list or change a CERTDATA record. **Note:** SERIAL# cannot be abbreviated.

HITRUST | TRUST |NOTRUST

Specifies a trust status for the certificate. If a trust status is not specified, it is set by default based on the validity of the certificate (self-signed) or, if the certificate was signed by another certificate, the validity of the signing certificate and the private key.

- HITRUST indicates that the certificate is both highly trusted and trusted. Any certificate usage applying to trusted certificates applies to highly trusted certificates. However, only certificate Certification aAuthority certificates (CERTAUTH) can be highly trusted.
- TRUST indicates that the certificate is trusted, i.e., that the certificate is valid for the user, site or certificate authority, and the private key has not been compromised.

  If no trust status has been specified and the certificate is self-signed, by default, the status will be set to TRUST.

  If no trust status has been specified and the certificate was signed by another certificate, by default, the status will be set to TRUST only if the following conditions are met:

  - The signing certificate can be located in the database.
  - The signing certificate is a Certification Authority (CERTAUTH).
  - The signing certificate is not expired.
  - The signing certificate's signature is valid.
  - The validity dates of the certificate being added fall within the range of the signing certificate's validity dates.

  Otherwise, the certificate will be inserted as not trusted (NOTRUST) and an informational message will be issued stating the reason why. However, if the signing certificate's signature is invalid, the certificate is not inserted.

  USER certificate—TRUST indicates that the certificate can be used to authenticate a userid.

  SITECERT certificate—TRUST indicates that the certificate can be used without authenticating it.

  CERTAUTH certificate—TRUST indicates that the certificate can be used to authenticate other certificates.

- NOTRUST indicates that the certificate is not trusted. If NOTRUST is specified or set by default, when the CERTDATA record is displayed, no trust value will be displayed; the trust field will remain blank.

```
change frank01.mycert NOTRUST
   CERTDATA / FRANK01.mycert LAST CHANGED BY FRANK01 ON 01/14/02-15:41
          CERTID(0000000000000000.CN=this is a CA.OU=acf2)
          LABEL(CERTAUTH.CERT4) SUBJDN(CN=this is a CA.OU=acf2)
```

Creating CERTDATA Profile Records

As part of the INSERT command processing, the certificate is read from the z/OS data set. eTrust CA-ACF2 decodes the certificate to extract the serial number, issuer's distinguished name, and other information and inserts some of this information along with information from the command input into CERTDATA profile data record. This information can be displayed with the LIST command.

To create a CERTDATA profile data record, enter profile administration mode.

```
     insert frank01.mycert dsn('frank01.mycert') active(02/02/02)
     expire(02/02/03) trust
CERTDATA / FRANK01.MYCERT LAST CHANGED BY FRANK01 ON 02/02/02-17:24
                  ACTIVE(02/02/02) CERTID(01.CN=histrust CA cert20)
                  EXPIRE(02/02/03) LABEL(FRANK01.MYCERT)
                  SUBJDN(CN=frank01.mycert) TRUST
PROFILE
```

To create a record using logonid and label:

```
     insert frank01 lable(mycert) dsn('frank01.mycert') active(02/02/02)
     expire(02/02/03) trust
CERTDATA / FRANK01.MYCERT LAST CHANGED BY FRANK01 ON 02/02/02-17:24
                  ACTIVE(02/02/02) CERTID(01.CN=histrust CA cert20)
                  EXPIRE(02/02/03) LABEL(MYCERT)
                  SUBJDN(CN=frank01.mycert) TRUST
PROFILE
```

Creating Changing CERTDATA Profile Records

Use the following command to enter profile administration mode.

```
set profile(user) div(certdata)
```

Use the CHANGE command to change **only** the following fields in the CERTDATA profile record:

- active date - [active(*date*)]
- expire date - [expire(*date*)]
- label - [newlabel(*label*)]
- trust status - [hitrust | trust | notrust]

Use the following commands to change a single record using record ID:

```
change frank01.mycert active(02/10/02) expire(02/20/03) newlabel(new certificate)
notrust

CERTDATA / FRANK01.MYCERT LAST CHANGED BY FRANK01 ON 02/02/02-17:24
                  ACTIVE(02/10/02) CERTID(01.CN=hitrust CA cert20)
                  EXPIRE(02/20/03) LABEL(new certificate)
                  SUBJDN(CN=frank0l.mycert)
PROFILE
```

Use the following commands to change a single record using record IDlogonid and label:

```
change frank01 label(FRANK01.MYCERT) active(02/10/02) expire(02/20/03)
newlabel(new certificate) trust

CERTDATA / FRANK01 LAST CHANGED BY FRANK01 ON 02/02/02-17:24
                  ACTIVE(02/10/02) CERTID(01.CN=hitrust CA cert20)
                  EXPIRE(02/20/03) LABEL(new certificate)
                  SUBJDN(CN=frank0l.mycert) TRUST
PROFILE
```

Use the following commands to change a single record using SERIAL # and ISSUERDN:

```
change userid SERIAL#(serial_number) ISSUERDN(dn) active(date) expire(date)
newlabel(label) [hitrust|trust|notrust]
```

Use the following commands to change multiple records:

```
change like(frank01.-) active(02/10/02) expire(02/20/03) trust
  ACF6D071 2 RECORDS CHANGED
PROFILE
```

Note: The label cannot be changed on a multiple (masked) record request.

Activating CERTDATA Profile Records

Use the following console commands to activate a newly created or changed CERTDATA profile data record:

```
f acf2,rebuild(usr),class(p)
f acf2,omvs
```

Viewing/Listing CERTDATA Profile Records

Using Use the following commands to enter profile administration mode.

```
set profile(user) div(certdata)
```

You can Uuse the LIST command as follows to view CERTDATA profile records.

To list a single record using record ID:

```
list frank01.mycert

CERTDATA / FRANK01.MYCERT LAST CHANGED BY FRANK01 ON 02/02/02-17:24
                  ACTIVE(02/10/02) CERTID(01.CN=hitrust CA cert20)
                  EXPIRE(02/20/03) LABEL(FRANK01.MYCERT)
                 SUBJDN(CN=frank0l.mycert)
PROFILE
```

To list a single record using record IDlogonid and label:

```
list frank01 label(MYCERT)

CERTDATA / FRANK01 LAST CHANGED BY FRANK01 ON 02/02/02-17:24
                  ACTIVE(02/10/02) CERTID(01.CN=hitrust CA)
                  EXPIRE(02/20/03) LABEL(MYCERT)
                  SUBJDN(CN=frank0l) TRUST
PROFILE
```

Use the following command to list a single record using SERIAL # and ISSUERDN:

```
list userid SERIAL#(serial_number) ISSUERDN(dn)
```

Use the following commands to list multiple records:

```
list like(frank01.-)

CERTDATA / FRANK01.MYCERT1 LAST CHANGED BY FRANK01 ON 02/02/02-17:24
                  ACTIVE(02/10/02) CERTID(01.CN=hitrust CA cert20)
                  EXPIRE(02/20/03) LABEL(FRANK01.MYCERT1)
                  SUBJDN(CN=frank0l.mycert1) TRUST
Certificate is not connected to any key rings


CERTDATA / FRANK01.MYCERT2 LAST CHANGED BY FRANK01 ON 02/02/02-17:24
                  ACTIVE(02/10/02) CERTID(01.CN=CA cert2)
                  EXPIRE(02/20/03) LABEL(FRANK01.MYCERT2)
                  SUBJDN(CN= frank0l.mycert2) TRUST
Certificate is not connected to any key rings


CERTDATA / FRANK01.MYCERT3 LAST CHANGED BY FRANK01 ON 02/02/02-17:24
                  ACTIVE(02/10/02) CERTID(01.CN=CA cert21)
                  EXPIRE(02/20/03) LABEL(FRANK01.MYCERT3)
                  SUBJDN(CN=frank0l.mycert3) TRUST
Certificate is not connected to any key rings
PROFILE
```

Deleting CERTDATA Profile Records

Use the following command to enter profile administration mode .

```
set profile(user) div(certdata)
```

You can uUse the DELETE command to delete a specific CERTDATA profile record.

Use the following command to delete a single record using record IDlogonid:

```
delete frank01.mycert

 ACF6D073 CERTDATA / FRANK01.MYCERT RECORD DELETED
PROFILE
```

Use the following command to delete a single record using record IDlogonid and label:

```
delete frank01 label(MYCERT)
 ACF6D073 CERTDATA / FRANK01 RECORD DELETED
PROFILE
```

Use the following command to delete a single record using SERIAL # and ISSUERDN:

```
delete userid SERIAL#(serial_number) ISSUERDN(dn)
```

Use the following command to delete multiple records:

```
delete like(userid.-)
```

Automatic Registration of Digital Certificates

CERTDATA profile records can also be dynamically inserted or deleted through a process known as Automatic Registration of Digital Certificates. An installation-written or vendor-provided CGI program requests automatic registration by invoking a z/OS andUNIX callable service. A program of this sort would typically validate a user's digital certificate and then prompt the user for an ID and password, which are then validated by eTrust CA-ACF2. If the validation is successful, the certificate is presented to eTrust CA-ACF2 and a CERTDATA profile record is dynamically created and associated with that user.

Dynamically inserted profile records can be distinguished by in two ways: first, the record key that contains the word AUTO*nnn* as the suffix, where *nnn* is a numeric value; second, the DSN field is blank since the certificate was not read from a z/OS data set.

A user must be authorized through the SAF FACILITY class before a profile record is dynamically inserted or deleted. The FAC resource rule $KEY values to allow dynamic INSERT and DELETE are:

```
IRR.DIGTCERT.ADD
IRR.DIGTCERT.DELETE
```

The following is a rule example:

```
SET RESOURCE(FAC)

COMPILE

$KEY(IRR) TYPE(FAC)
 DIGTCERT.ADD UID(userid) ALLOW
 DIGTCERT.DELETE UID(userid) ALLOW

STORE
```

Note: Ensure that you do not inadvertently allow access to these resources because of masking in your resource rules.

CICS Profile Data Records

CICS profile data information is extracted by CICS to determine the identity and characteristics of users who are trying to access the system.

Record ID	Fields
recid	OPIDENT(*opident*) OPPRTY(*opprty*) TIMEOUT(*timeout*) OPCLASS(*opclass*) FORCE|NOFORCE

Field Descriptions

recid
A one to eight-character maskable userid. Standard logonid masking conventions apply, except that the mask can contain only asterisks (*), not a dash (-).

OPIDENT(*opident*)
A one to three-character operator ID.

OPPRTY(*opprty*)
Operator priority value from zero to 255.

TIMEOUT(*timeout*)
Idle time value from zero through 15,555 minutes.

OPCLASS(*opclass*)
Operator class values from one to 24. Use commas to separate individual values; for example, OPCLASS(1,4,10).

FORCE|NOFORCE
Indicates whether the user is signed off (FORCE) or not signed off (NOFORCE) when an XRF takeover occurs.

Example

To create a CICS profile data record, enter profile record administration mode and insert the record as follows. The record created in the following sample defines the operator identity, priority, timeout and class characteristics of a set of users who are to be signed off when an XRF takeover occurs:

```
SET PROFILE(USER) DIVISION(CICS)

INSERT syspgm** OPIDENT(chi) OPPRTY(3) TIMEOUT(30) OPCLASS(4) FORCE
```

Masking the record ID requires that the USER profile class be defined in GSO INFODIR. To add this entry if it does not already exist, issue the following commands:

```
SET C(GSO)

CHANGE INFODIR TYPES(R-PUSR) ADD
```

To activate newly created or changed profile records, issue this command:

```
F ACF2,REBUILD(USR),CLASS(P)
```

DCE Profile Data Records

A z/OS user is defined to z/OS Unix System Services DCE by assigning a UUID and HOMEUUID to the user. This is done by creating a DCE profile record.

Record ID	Fields
recid	UUID(*uuid*) DCEKEY DCENAME(*dcename*) HOMEUUID(*homeuuid*) HOMECELL(*homecell*) AUTOLOG|NOAUTOLOG

Field Descriptions

recid
z/OS userid. This value cannot be masked.

UUID(*uuid*)
36-character string form of the user's UUID.

DCEKEY
A field which is not administered by the ACF command and can never be displayed. The DCEKEY is stored I encoded (masked) or encrypted form in the database depending on how it was originally designated to be stored in the KEYSMSTR profile record. See "KEYSMSTR Profile Record" for additional information.

DCENAME(*dcename*)
1023-character principal name of the user.

HOMEUUID(*homeuuid*)
36-character string form of the user's home cell UUID.

HOMECELL(*homecell*)
1023-character home cell name.

AUTOLOG|NOAUTOLOG
Indication that the user should be automatically signed on to z/OS Unix System Services DCE.

eTrust CA-ACF2 automatically creates a UUID-to-userid cross-reference table at startup. If you insert new profile records or change the UUID or HOMEUUID fields, you must refresh the cross-reference tables. To accomplish this, issue the following console operator command:

EIM Profile Data Records

The EIM segment of the USER profile identifies the LDAPBIND profile record that contains the bind information for the application.

Record ID	Fields
recid	LDAPPROF(*ldapprof*)

Field Descriptions

recid
This is a one- to eight-character userid that is associated with the application. This value cannot be masked.

LDAPPROF
The one- to eight-character name of the LDAPBIND profile record containing the bind information for the application.

KEYRING Profile Data Records

A KEYRING profile data record contains a collection of digital certificates. Certificates in a key ring identify a trust relationship. Client or peer entities that wish to communicate over a network can use certificates in key rings to evaluate the trustworthiness of a prospective candidate.

Record ID	Fields
recid	DEFAULT(*userid.suffix*) RINGNAME(*ringname*)

Field Descriptions

recid
This is a one to eight-character userid that is to be associated with the key ring. A one to eight-character suffix can be appended to the userid to create a unique record key. The suffix must be separated from the userid with a period.

DEFAULT(*userid.suffix*)
Specifies the record key of a CERTDATA certificate record that is to be used as the default certificate for this key ring. A key ring can have one default certificate. If a default certificate already exists, this certificate replaces its DEFAULT status.

RINGNAME(*ringname*)
Specifies the name of the key ring. The key ring name can be up to 237 characters in length.

Examples

Inserting KEYRING Profile Data Records
To insert KEYRING records:

INSERT userid.suffix RINGNAME(ringname)

Changing KEYRING Profile Data Records
You can use the CHANGE command as described here to change KEYRING profile records in a number of ways.

To change a single record using record ID:

```
CHANGE userid.suffix NEWNAME(ringname) DEFAULT(userid.suffix)
```

To change a single record using RINGNAME:

```
CHANGE userid RINGNAME(ringname) NEWNAME(ringname) DEFAULT(userid.suffix)
```

To change multiple records:

```
CHANGE LIKE(userid.-) DEFAULT(userid.suffix)
```

Notes:

- To change the RINGNAME in a record, the NEWNAME operand must be used. If RINGNAME is encountered in the command input, a change using RINGNAME request is assumed to be in effect.
- The RINGNAME field cannot be changed on a multiple (masked) record request.

Viewing KEYRING Profile Data Records

You can use the LIST command as described here to list KEYRING profile records in a number of ways.

To list a single record using record ID:

```
LIST userid.suffix
```

To list a single record using RINGNAME:

```
LIST userid RINGNAME(ringname)
```

To list multiple records:

```
LIST LIKE(userid.-)
```

Deleting KEYRING Profile Data Records

You can use the DELETE command as described here to delete KEYRING profile records in a number of ways.

To delete a single record using record ID:

```
DELETE userid.suffix
```

To delete a single record using RINGNAME:

```
DELETE userid RINGNAME(ringname)
```

To delete multiple records:

```
DELETE LIKE(userid.-)
```

KERB Profile Data Records

The KERB segment of the USER profile maps Kerberos for z/OS application user identity to an eTrust CA-ACF2 logonid.

Record ID	Fields
Recid	DES | NODES DES3 | NODES3 DESD | NODESD KERBNAME(*kerberos-principal-name*) MAXTKTLF(ticketlife)

Field Descriptions

recid
This is a one to eight character userid that is to be associated with the local principal name. A one to eight-character suffix can be appended to the userid to create a unique record key.

DES|NODES
Enables the DES encryption type setting to be defined for this logonid. This field is valid when KERBLVL(1) is active on the GSO OPTS record. The default is DES.

DES3|NODES3
Enables the DES3 encryption type setting to be defined for this logonid. This field is valid when KERBLVL(1) is active on the GSO OPTS record. The default is DES3.

DESD|NODESD
Enables the DESD encryption type setting to be defined for this logonid. This field is valid when KERBLVL(1) is active on the GSO OPTS record. The default is DESD.

KERBNAME
Specifies the local Kerberos principal name. The *kerberos-principal-name* you define can consist of any character except the @(X'7C') character. It is highly recommended that any of the EBCDIC variant characters be avoided to prevent problems between different code pages. This field is case sensitive. eTrust CA-ACF2 will not ensure that a valid Kerberos principal name has been entered. A local Kerberos principal name must not be qualified with a realm name when specified in the KERBNAME parameter. eTrust CA-ACF2 will verify that the local principal name, when qualified with the local realm name, does not exceed 240 characters.

For example, if the local realm name is REALM1, fully qualified local principal names are prefixed with /…/REALM1/ and are limited to 228 characters. If the local realm name is REALM1.LISLE.CA.COM, fully qualified local principal names will be prefixed with /…/REALM1.LISLE.CA.COM/ and are limited to 215 characters. The length verification requires that the GSO REALM record for the local realm KERBDFLT be defined and contain the name of the local realm before inserting the KERB USER profile records. Otherwise, the local Kerberos principals might not be properly defined.

MAXTKTLF
Maximum ticket life for this field is in seconds. The range of values is from 1 to 2,147,483,647 seconds. If the MAXTKTLF parameter is defined for a principal, the system takes the most restrictive of the values defined for the principal and the value specified on the definition of the local realm (GSO REALM record with REALM(KERBDFLT). If the value in the principal definition exceeds the value in the local realm definition, the value in the local realm definition will be used.

KERBLINK User Profile Record

The KERBLINK USER profile record defines the foreign principals to a local node by linking it to a defined user. KERBLINK profiles can define an individual principal from a foreign realm or all principals from a particular foreign realm. The local LID does not need to have a KERB User Profile record associated with it.

Record ID	Fields
Userid.recid	KBLKNAME(*foreignprincipal*)

Field Description

recid
This is a 1- to 8-character userid that is to be associated with the foreign principal. A 1- to 8-character suffix can be appended to the userid to create a unique record key. The suffix must be separated from the userid with a period.

KBLKNAME
Defines the foreign principal name. The foreign principal must be fully qualified with the name of the foreign realm. eTrust CA-ACF2 will verify that the NAME field starts with /…/ to ensure a valid field. The maximum length of this field is 240-characters. If you wish to map the same eTrust CA-ACF2 LID to all foreign principals in a foreign name, only specify the foreign realm name. The NAME field is folded to uppercase upon entry.

Examples

To map the same eTrust CA-ACF2 LID to all foreign principals in a foreign realm:

Specify only the foreign realm name:

```
/…/REALM1.LISLE.CA.COM/
```

The suffix is an identifier used to distinguish the KERBLINK record from others defined to the same user.

To map a unique eTrust CA-ACF2 LID to each foreign principal:

Specify the foreign realm and foreign principal:

```
/…/REALM1.LISLE.CA.COM/LEEBRO1
```

All characters for the REALM field will be folded to uppercase.

LANGUAGE Profile Data Records

LANGUAGE profile data information is extracted for use by applications that use the z/OS message service. You can define a primary and secondary language code for each user or, if using masking, a group of users.

Record ID	Fields
recid	PRIMARY(*primary*) SECONDRY(*secondary*)

Field Descriptions

recid
A one to eight-character maskable userid. Standard logonid masking conventions apply, except that the mask can contain only asterisks (*), not a dash (-).

PRIMARY(*primary*)
This is a one to 24-byte character field. However, eTrust CA-ACF2 uses only the first three bytes of the field. This field specifies the three-character language code for the user's primary language.

Although you can specify your own language codes, there are accepted standard values for most common languages. These accepted values are documented in many places, including the *IBM z/OS MVS Programming: Assembler Services Guide (z/OS)*, the *IBM MVS/ESA Assembler Services Guide (MVS/ESA 5.x)*, or the *IBM MVS/ESA Assembler Programming Guide (MVS/ESA 4.x)*.

When a SAF EXTRACT call is issued to obtain the profile information for a user, this field is returned in the extracted primary language field for the user. If no primary language is specified for the user in a USER PROFILE LANGUAGE record, this global primary language defined in the GSO OPTS record is returned in the extracted primary language field.

If eTrust CA-ACF2 national language support is implemented (documented in the *Getting Started* guide), the user primary language is used as the primary language for all messages directed to the user, eTrust CA-ACF2 issues the message in that language, if possible, when the message cannot be located or issued in the primary language. If no secondary language is specified for the user in a USER PROFILE LANGUAGE record, the global secondary language is used as the secondary language for the message. If no global primary language is specified, eTrust CA-ACF2 uses ENU (U.S. English) as the default.

SECONDRY(*secondary*)

This is a one to 24-byte character field. However, eTrust CA-ACF2 uses only the first three bytes of the field. This field specifies the three-character language code for the user's secondary language.

Although you can specify your own language codes, there are accepted standard values for most common languages. These accepted values are documented in many places, including the *IBM z/OS MVS Programming: Assembler Services Guide (z/OS)*, the *IBM MVS/ESA Assembler Services Guide* (MVS/ESA 5.x), and the *IBM MVS/ESA Assembler Programming Guide* (MVS/ESA 4.x).

When a SAF EXTRACT call is issued to obtain the profile information for a user, this field is returned in the extracted secondary language field for the user. If no secondary language is specified for the user in a USER PROFILE LANGUAGE record, this global secondary language defined in the GSO OPTS record is returned in the extracted secondary language field.

If eTrust CA-ACF2 national language support is implemented (documented in the *Getting Started* guide), the user secondary language is used as the secondary language for all messages directed to the user, eTrust CA-ACF2that is, issues the message in that language, if possible,. This occurs when the message can not be located or issued in the primary language. If no secondary language is specified for the user in a USER PROFILE LANGUAGE record, the global secondary language is used as the secondary language for the message. If no global secondary language is specified, eTrust CA-ACF2 uses ENU (U.S. English) as the default.

Example

An example setting a primary language for all users and a different primary language for a specific user follows:

```
SET PROFILE(USER) DIVISION(LANGUAGE)

INSERT ******** PRIMARY(ENU) SECONDRY(ENU)
INSERT userid PRIMARY(FRA) SECONDRY(ENU)
```

Masking of the record ID requires that the USER profile class be defined in GSO INFODIR.

LDAPBIND EIM Profile Data Record

eTrust CA-ACF2 LDAPBIND profile records provide support for IBM Enterprise Identity Mapping (EIM). LDAP bind information is extracted from eTrust CA-ACF2 for connection to an LDAP server. The EIM segment of the USER profile identifies an LDAPBIND profile record that contains bind information for the application.

Record ID	Fields
Recid	DOMAINDN(*domaindn*) ENABLE | DISABLE KERBREG(KERBEROS *registry name*) LOCALREG(*localregistry*) X509REG(X590 *registry name*)

Field Descriptions

recid
This is a one- to eight-character id of the record. This value cannot be masked.

DOMAINDN(*domaindn*)
The distinguished name of an EIM domain. The field can be up to 1023 characters in length.

ENABLE | DISABLE
Specifies whether or not new connections may be established with the specified EIM domain. The default is ENABLE.

KERBREG(KERBEROS *registry name*)
A one to 255-character name identifying the KERBEROS registry associated with this EIM or PROXY definition. Any value entered will be changed to uppercase. To eliminate a previously specified KERBREG value on a CHANGE statement, specify KERBREG().

LOCALREG(*localregistry*)
The name of the local registry. The field can be up to 255 characters in length.

X509REG(X509 *registry name*)
A one to 255-character name identifying the X509 registry associated with this EIM or PROXY definition. Any value entered will be cahnged to uppercase. To eliminate a previously specified X509REG value on a CHANGE statement, specify X509REG().

LDAPBIND PROXY Profile Data Record

eTrust CA-ACF2 LDAPBIND profile records provide support for IBM Enterprise Identity Mapping (EIM). LDAP bind information is extracted from eTrust CA-ACF2 for connection to an LDAP server. The EIM segment of the USER profile identifies an LDAPBIND profile record that contains bind information for the application.

Record ID	Fields
Recid	BINDDN(*binddn*) BINDPTOD(00/00/00-00:00) BINDPW(*bindpw*) LDAPHOST(*ldaphost*)

Field Descriptions

recid
This is a one- to eight-character id of the record. This value cannot be masked.

BINDDN(*binddn*)
The distinguished name to use when authenticating to the LDAP server. The field can be up to 1023 characters in length.

Note: The list of attribute pairs specified in the BINDDN field are considered a single value, therefore, it must be enclosed in single quotes. The single quotes will qualify the entire list of attribute value pairs as one value.

BINDPTOD(00/00/00-00:00)
Specifies the date and time the BINDPW field was last changed. eTrust CA-ACF2 lists the date in the format mm/dd/yy, dd/mm/yy, or yy/mm/dd, depending on the DATE field of the GSO OPTS record. You cannot set this field. eTrust CA-ACF2 maintains and displays it. (Eight-byte binary field)

BINDPW(*bindpw*)

The password to use when authenticating to the LDAP server. The field can be up to 128 characters in length.

LDAPHOST

LDAPHOST(*ldapurl*)

Specifies the URL of the LDAP server that the z/OS LDAP Server will contact when acting as a proxy on behalf of a requester. An LDAP URL has a format such as ldap://123.45.6:389 or ldaps://123.45.6:636, where ldaps indicates that an SSL connection is desired for a higher level of security. LDAP will also allow you to specify the host name portion of the URL using either the text form (LDAP.HOST.CA.COM) or the dotted decimal address (111.222.33.44). The port number is appended to the host name, separated by a colon ':'. See your LDAP server documentation for information on how to set up the LDAP server for SSL connections.

This value must be at least 10 bytes long and can be up to 1023 bytes long. It must start with either ldap:// or ldaps://. Any characters may be entered in the remaining portion of the URL, but you should ensure that the URL conforms to TCP/IP conventions. Normally, characters such as commas, blanks, parenthesis, semicolons and single quotes are not allowed in a host name. These characters will be accepted if the LDAPHOST is enclosed in single quotes.

eTrust CA-ACF2 does not validate that the contents of the URL are valid.

LINUX Profile Data Records

LINUX profile data records contain information needed by LINUX/390 Service (PAM Server) to verify user access.

LINUX profile data records maps the Linux application user identity to an eTrust CA-ACF2 logonid, defines Linux home directory, Linux program, Linux UID number, and Linux group name.

Record ID	Fields
recid	AUTOUIDL LINUXGRP(*group-name*) LINUXHOM(*home-directory*) LINUXNAM(*application-user-id*) LINUXPGM(*program*) LINUXUID(*uid#*)

Field Descriptions

recid
Specifies a one-to-eight character eTrust CA-ACF2 logonid. A one-to-eight character qualifier may be added to specify a Linux machine to which this user definition is to be associated. This value cannot be masked.

AUTUIDL
Automatically assigns a LINUXUID value when there is an active GSO AUTOIDLX record that specifies ASSIGNU. If there is **no** active GSO AUTOIDLX record or there is an active GSO AUTOIDLX record that specifies NOASSIGNU, you must explicitly specify a LINUXUID value. AUTOUIDL is implied if neither AUTOUIDL nor LINUXUID is specified on an INSERT command and there is an active GSO AUTOIDLX record that specifies ASSIGNU.

AUTOUIDL is never implied on a CHANGE command, it must be explicitly stated. The LINUXUID and AUTOUIDL keywords are mutually exclusive.

Note: If the recid of the LINUX user profile record that is being inserted contains a qualifier, CA-ACF2 Security will try to find and use an active AUTOIDLX record that has a matching qualifier when attempting to automatically assign a LINUXUID value. If an AUTOIDLX record with a matching qualifier is not found, CA-ACF2 Security will attempt to use an active AUTOIDLX record without a qualifier.

LINUXGRP(*group_name*)
A 1 to 8 character field that specifies the name of the LINUX group profile record. This field is optional.

LINUXHOM(*home_directory*) -
A 1 to 1024 upper or lower case alphanumeric character field that specifies the Pathname of the Initial Directory when a user enters a Linux command or the ISPF shell. This is a required field. If left undefined on an INSERT command, it defaults to '/home/%L'.

The following strings substitutions are allowed:

Symbolic	String Substitution
%L	Substitutes an eTrust CA-ACF2 logonid (8 bytes).
%N	Substitutes to user name defined in eTrust CA-ACF2 logonid record.
%X	Substitutes to LINUX/390 application user id defined in LINUXNAM field. **Note:** If LINUXNAM field is not defined, it is substituted to eTrust CA-ACF2 logonid.

LINUXNAM(*application-user-id*)
A 1 to 1024 upper or lower cased alphanumeric character field that specifies the LINUX Application User Identity. This field is optional.

LINUXPGM(*program*)
A 1 to 1024 upper and lower cased alphanumeric character field that specifies LINUX/390 Service Shell Program when Linux command is first entered. This field is required. If this field is left undefined on an INSERT command, it defaults to '/bin/bash'.

LINUXUID(*uid#*)
A numeric field that accepts values from 0 to 2,147483,647. LINUXUID can be automatically assigned if there is an active GSO AUTOIDLX record that specifies ASSIGNU. The LINUXUID and AUTOUIDL keywords are mutually exclusive. This field is required.

Note: Automatically assigned LINUXUID numbers range from 500 to 2,147,483,647. To specify a LINUXUID number from 0 to 499, it must be explicitly specified.

Note: If a qualified Linux profile does not specify a value for the LINUXNAM field, eTrust CA-ACF2 will look for an unqualified Linux profile with the same record ID and will attempt to use the LINUXNAM from that record. If the unqualified record does not exist or does not specify a LINUXNAM, eTrust CA-ACF2 will use the value in the DFTLNXU field of the GSO OPTS record.

If a qualified user profile does not specify a value for LINUXGRP, eTrust CA-ACF2 will attempt to use the LINUXGRP value from the unqualified Linux profile with the same record ID. If the unqualified profile does not exist or does not contain a value for the LINUXGRP field, the value, if any, in the DFTLNXG field of the GSO OPTS record will be used.

Commands

Build cross-reference table to map LINUX/390 application user id to eTrust CA-ACF2 logonid.

```
F ACF2,OMVS(LINUX)
```

If you insert or change a user profile record and it is resident, then you must issue a REBUILD command to activate the changes.

```
F ACF2,REBUILD(USR),CLASS(P)
```

Example

Following example inserts LINUX user profile record, LNXUSER with Linux a UID value of 99.

```
SET PROFILE(USER) DIVISION(LINUX)

INSERT LNXUSER LINUXHOM(/HOME/%X) LINUXNAM(LINUXUSERGUY) LINUXUID(99)

LINUX / LNXUSER LAST CHANGED BY USER01 ON 06/26/03-16:10
        LINUXNAM(LINUXUSERGUY) LINUXHOM(/HOME/%X) LINUXPGM(/bin/bash)
        LINUXUID(99)
```

This example automatically assigns LINUXUID using both the INSERT and CHANGE subcommands. There is an active AUTOIDLX record that specifies ASSIGNU, UIDSTART(500), and UIDNEXT(500). By end of this example, AUTOIDLX contains UIDSTART(500) and UIDNEXT(502).

```
SET PROFILE(USER) DIVISION(LINUX)

INSERT LNXUSER2

LINUX / LNXUSER2 LAST CHANGED BY USER01 ON 06/26/03-16:18
        LINUXHOM(/home/%L) LINUXPGM(/bin/bash) LINUXUID(500)

CHANGE LNXUSER2 AUTOIDL

LINUX / LNXUSER2 LAST CHANGED BY USER01 ON 06/26/03-16:19
        LINUXHOM(/home/%L) LINUXPGM(/bin/bash) LINUXUID(501)
```

LNOTES Profile Data Records

The LNOTES segment of the USER profile maps a Lotus Notes for Z/OS application user identity to an eTrust CA-ACF2 logonid.

Record ID	Fields
Recid	SNAME(*application-user-id*)

Field Descriptions

recid
eTrust CA-ACF2 logonid.z/OS and OS/390 userid.

SNAME(*application-user-id*)
Specifies the Lotus Notes for Z/OS short name. The name should match the one stored in the Lotus Notes address book for this user, but will not be verified by the command. The short name can contain 1-64 characters. The following characters are valid: uppercase and lowercase alphabetic characters, 0 through 9, & (x'50'), - (x'60'), . (x'4b'), and _ (X'6D').

NDS Profile Data Records

The NDS segment of the USER profile maps a Novell Directory Services application user identity to an eTrust CA-ACF2 logonid.

Record ID	Fields
Recid	UNAME(*application-user-id*)

Field Descriptions

recid
eTrust CA-ACF2 logonid.

UNAME(*application-user-id*)
Specifies the Novell Directory Services user name. The name should match the one stored in the Novell Directory Services directory for this user, but will not be verified by the command. The short name can contain 1-246 characters. Any character is valid, with the following exceptions: * (x'5C'), + (x'4E'), | (x'4F'), = (X'7E'), , (x'6B'), " (x'7B'), (X'79'), / (x'61'), : (x'7A'), ; (x'5E'), and brackets ([and]).

NETVIEW Profile Data Records

The NETVIEW segment of the USER profile defines NetView operator attributes.

Record ID	Fields
Recid	IC(*initial-command*) CONSNAME(*console-id*) SECCTL(GLOBAL|GENERAL) MSGRECVR|NOMSGRECVR NGMFADMN|NONGMFADMN NTVCLASS(ntv*class1,…,ntvclassn*) DOMAINS(*pgmid1,…,pgmidn*)

Field Descriptions

recid
eTrust CA-ACF2 logonid.z/OS and OS/390 userid.

IC(*initial-command*)
The initial command to be executed when the user signs on. This value can be up to 255 characters.

CONSNAME(*console-id*)
The default z/OS andconsole identifier.

SECCTL(GLOBAL|GENERAL)
Security check indicator. Use a null value to represent the value SPECIFIC. Other than null, acceptable values are GLOBAL and GENERAL. To change the SECCTL field of the NETVIEW USER profile record to a null value, you have to specify REP at the end of the CHANGE command. If you do not do this, the change does not occur. Setting SECCTL to a null value is how we represent the value SPECIFIC. See the following example for more information.

MSGRECVR|NOMSGRECVR
Indicates whether the user can receive unsolicited messages.

NGMFADMN|NONGMFADMN
Indicates whether the user has administrative authority to the Graphic Monitor Facility.

NTVCLASS(*ntvclass1,...,ntvclassn*)
A list of scope classes represented by the values 1 through 2040.

DOMAINS(*pgmid1,...,pgmidn*)
A list of program identifiers in another domain to which the user has authority.

Examples

To create a NETVIEW profile data record, enter profile administration mode by issuing the SET PROFILE(USER) DI(NETVIEW) command. Use the INSERT command to define the record name and field specifications.

To activate newly created or changed profile records, issue this command:

```
F ACF2,REBUILD(USR),CLASS(P)
```

To set the SECCTL field to a value of SPECIFIC, use the following commands:

```
SET PROFILE(USER) DIV(NETVIEW)
CHANGE NETREC1 SECCTL( ) REP
```

OMVS Profile Data Records

The OMVS segment of the User profile contains information needed by z/OS Unix System Services to verify user access.

Record ID	Fields
Recid	ASSIZE(*max address space size*) AUTOUID CPUTIME(*max cputime for a dubbed process*) FILEPROC(*max files per process*) HOME(*home-directory*) MEMLIMIT(*max non-shared memory space*) PROCUSER(*max number of processes*) SHMEMMAX(*max shared memory space*) THREADS(*max number of pthread_created threads*) MMAPAREA(*max data space pages for HFS mappings*) OMVSPGM(*program*) UID(*uid*)

Note: The CPUTIME, ASSIZE, FILEPROC, PROCUSER, THREADS, and MMAPAREA values are only valid at z/OS and OS/390 2.8 and above. The MEMLIMIT and SHMEMMAX values are only valid at z/OS 1.6 and above. You might set them at any level, but they will not be recognized until that level.

Field Descriptions

recid
The eTrust CA-ACF2z/OS and OS/390 logonid. This value cannot be masked.

ASSIZE(*max-address-space-size*)
This field overrides the MAXASSIZE parameter in the BPXPRMxx member of parmlibPARMLIB for this user. The value can be from 10,485,760 to 2,147,483,647. This field is not set when the record is inserted. If the field is not set, a RACROUTE EXTRACT will return a length of 4 and a value of X'FFFFFFFF'. Unix System Services takes the system defaults set in BPXPRMxx when this field is not set. You can remove this field from the record by changing it to a null value (change user ASSIZE()). ASSIZE is equivalent to ASSIZEMAX in RACF.

AUTOUID
Automatically assigns a UID value when there is an active GSO AUTOIDOM record that specifies ASSIGNU. AUTOUID is implied if neither AUTOUID nor UID is specified on an insert command and there is an active GSO AUTOIDOM record that specifies ASSIGNU. AUTOUID is never implied on a change command, it must be stated explicitly. The UID and AUTOUID keywords are mutually exclusive.

CPUTIME(*max-cputime-for-a-dubbed-process*)
This field overrides the MAXCPUTIME parameter in the BPXPRMxx member of parmlibPARMLIB for this user. The value can be from 7 to 2,147,483,647. This field is not set when the record is inserted. If the field is not set, a RACROUTE EXTRACT will return a length of 4 and a value of X'FFFFFFFF'. Unix System Services takes the system defaults set in BPXPRMxx when this field is not set. You can remove this field from the record by changing it to a null value (change user CPUTIME()). CPUTIME is equivalent to CPUTIMEMAX in RACF.

FILEPROC(*max-files-per-process*)
This field overrides the MAXFILEPROC parameter in the BPXPRMxx member of parmlibPARMLIB for this user. The value can be from 3 to 65,535. This field is not set when the record is inserted. If the field is not set, a RACROUTE EXTRACT will return a length of 4 and a value of X'FFFFFFFF'. Unix System Services takes the system defaults set in BPXPRMxx when this field is not set. You can remove this field from the record by changing it to a null value (change user FILEPROC()). FILEPROC is equivalent to FILEPROCMAX in RACF.

HOME(*home-directory*)
A field that defines the pathname of the initial directory used when a user enters the OMVS command or enters the ISPF shell. Specify from one to 1023 upper-case or lower-case characters. If HOME is not defined, z/OS Unix System Services sets root as the initial directory.

MEMLIMIT(*max non-shared memory space*)
Specifies the maximum number of bytes of non-shared memory space that can be allocated by the user. The value can be from 0 to 16,777,215 followed by a letter M (megabyte), G (gigabyte), T (terabyte), or P (petabyte). See the multiplier table under the SHMEMMAX field description for a detailed description. The maximum value is 16383P.

This field is not set when the record is inserted. If the field is not set, a RACROUTE EXTRACT will return a length of 4 and a value of x'FFFFFFFF'. You can remove this field from the record by changing it to a null value (MEMLIMIT()). MEMLIMIT is only valid at z/OS 1.6 and above.

For example, if MEMLIMIT (16383P) is specified, then the user can allocate up to 18,445,618,173,802,708,992 bytes.

MMAPAREA(*max-data-space pages-for-HFS-mappings*)
This field overrides the MAXMMAPAREA parameter in the BPXPRMxx member of parmlibPARMLIB for this user. The value can be from 1 to 16,777,216. This field is not set when the record is inserted. If the field is not set, a RACROUTE EXTRACT will return a length of 4 and a value of X'FFFFFFFF'. Unix System Services takes the system defaults set in BPXPRMxx when this field is not set. You can remove this field from the record by changing it to a null value (change user MMAPAREA()). MMAPAREA is equivalent to MMAPAREAMAX in RACF.

OMVSPGM(*program*)
An optional field that defines the user's UNIX System Services shell program started when the OMVS command is entered or when a UNIX System Services batch job is started using the BPXBATCH program. Specify from one to 1023 upper-case or lower-case characters. If OMVSPGM is not defined, UNIX System Services gives control to the default shell program.

PROCUSER(*max-number-of-processes*)
This field overrides the MAXPROCUSER parameter in the BPXPRMxx member of parmlibPARMLIB for this user. The value can be from 3 to 32,767. This field is not set when the record is inserted. If the field is not set, a RACROUTE EXTRACT will return a length of 4 and a value of X'FFFFFFFF'. Unix System Services takes the system defaults set in BPXPRMxx when this field is not set. You can remove this field from the record by changing it to a null value (change user PROCUSER()). PROCUSER is equivalent to PROCUSERMAX in RACF.

SHMEMMAX(*max shared memory space*)
Specifies the maximum number of bytes of shared memory space that can be allocated by the user. This value can be from 1 to 16,777,215 followed by a letter M (megabyte), G (gigabyte), T (terabyte), or P (petabyte). See the multiplier table below for a detailed description. The maximum value is 16383P.

This field is not set when the record is inserted. If the field is not set, a RACROUTE EXTRACT will return a length of 4 and a value of X'FFFFFFFF'. Unix System Services takes the value set in the IPCSHMNSEGS parameter in the BPXPRMxx when this field is not set. You can remove this field from the record by changing it to a null value (SHMEMMAX()). SHMEMMAX is only valid at z/OS 1.6 and above.

For example, if SHMEMMAX (150M) is specified, then the user can allocate up to 157,286,400 bytes.

Multiplier	Decimal	Binary	Hex
M=Megabyte	1,048,576	2**20	00000000 00100000
G=Gigabyte	1,073,741,824	2**30	00000000 40000000
T=Terabyte	1,099,511,627,776	2**40	00000010 00000000
P=Petabyte	1,125,899,906,842,624	2**50	00040000 00000000

THREADS(*max-number-of-pthread_created-threads*)

This field overrides the MAXTHREADS parameter in the BPXPRMxx member of parmlibPARMLIB for this user. The value can be from 0 to 100,000. This field is not set when the record is inserted. If the field is not set, a RACROUTE EXTRACT will return a length of 4 and a value of X'FFFFFFFF'. Unix System Services takes the system defaults set in BPXPRMxx when this field is not set. You can remove this field from the record by changing it to a null value (change user THREADS()). THREADS is equivalent to THREADSMAX in RACF.

UID(*uid*)

A required numeric field that accepts values from zero to 2,147,483,647. A value of zero indicates that this user is a superuser. The NO-OMVS attribute on the logonid nullifies the user profile record and BPX.DEFAULT.USER for this user. NO-OMVS is equivalent to NOUID in RACF. The UID keyword and the AUTOUID keyword are mutually exclusive.

Rebuild Command

If you insert or change a user profile record and it is resident, then you must issue a REBUILD command to activate the change.

```
F ACF2,REBUILD(USR),CLASS(P)
```

Examples

This section explains how to set up eTrust CA-ACF2 to automatically assign UID numbers for PROFILE(USER),DIV(OMVS) records. Readers should already be familiar with the AUTOIDOM record. See the "Maintaining Global System Options Records" chapter for information on the AUTOIDOM record. Considerations are discussed for shared database environments and CPF environments.

AUTO Assignment of UID Numbers

To use this feature there must be an active GSO AUTOIDOM record. For this example, there is an AUTOID.OMVS record with following fields:

```
CPU1 / AUTOID.OMVS LAST CHANGED BY USER01 ON 06/26/03-14:04
                ASSIGNG ASSIGNU GIDEND(50,000) GIDNEXT(25) GIDSTART(9)
                UIDEND(2,147,483,647) UIDNEXT(195) UIDSTART(1)
```

The following command inserts an OMVS Group profile record, OMVSUSR; its UID field is automatically assigned. Note that the AUTOUID field is assumed on the INSERT command and does not need to be specified.

```
SET PROFILE(USER) DIV(OMVS)
INSERT OMVSUSR
OMVS / OMVSUSR LAST CHANGED BY USER01 ON 06/26/03-16:26
                UID(195)
```

The following command automatically assigns the new UID value to a recently inserted, OMVSUSR, record. Note that AUTOUID field needs to be specified to automatically assign the new UID value, otherwise, you must explicitly specify the new UID value.

```
CHANGE OMVSUSR AUTOUID

OMVS / OMVSUSR LAST CHANGED BY USER01 ON 06/26/03-16:30
                UID(196)
```

After updating OMVS User profile record, OMVSUSR, the AUTOIDOM record now reflects the recent updates with a new value in UIDNEXT(197).

```
T C(GSO)
LIST AUTOID.OMVS

CPU1 / AUTOID.OMVS LAST CHANGED BY USER01 ON 06/26/03-14:04
                ASSIGNG ASSIGNU GIDEND(50,000) GIDNEXT(25) GIDSTART(9)
                UIDEND(2,147,483,647) UIDNEXT(197) UIDSTART(1)
```

SHOW OMVS

The default value range is 1 to 2,147483,647 for UIDs. Some sites may want to specify a range of numbers that are not in use by any existing UID records. To see what numbers are already in use, issue the SHOW OMVS command. If, for example, you want to see what UID values are already in use in the range 900 through 399999, you can issue:

```
SHOW OMVS USERS(900-399999)
```

If you wish to see only the UID values that belong to more than one user, issue the ACF command:

```
SHOW OMVS DUPLICATES
```

The DUPLICATES keyword can be used together with another keyword. For example the following will show only duplicate UID values that are in the range of 1 to 2000:

```
SHOW OMVS USERS(1-2000) DUPLICATES
```

OPERPARM Profile Data Records

OPERPARM profile data information is extracted by the TSO/E Extended MCS Console Facility. The profile data information defines a user's console attributes. In addition to defining the OPERPARM profile data records, certain resource rules for the OPERCMDS class must be written. **Note:** Changes made using the CONSPROF command under TSO will not be retained across sessions.

Record ID	Fields
recid	STORAGE(*nnnn*) AUTH(*authority*) AUTO(YES|NO) MFORM(*message-format*) MSGLEV(*msglevel*) MONITOR(*monitor*) ROUTCODE(*routcode*) LOGCMD(YES|NO) MIGID(YES|NO) DOM(*delete-operator-message*) UD(YES|NO) KEY(*console-key*) CMDSYS(*system-name*) ALTGROUP(*altgroup*) OPERMSCP(*opermscp*)

Field Descriptions

recid
A one to eight-character maskable userid. Standard logonid masking conventions apply, except that the mask can contain only asterisks (*), not a dash (-).

STORAGE(*nnnn*)
A value from 0 to 2000 defining the amount of storage in megabytes to be used for message queuing.

AUTH(*authority*)
The authority level to issue console commands.

- **MASTER**—master console authority; this value cannot be specified with any other value.
- **ALL**—authority to issue system control commands, input/output commands, console control commands, and informational commands; this value cannot be specified with any other value.
- **SYSTEM**—authority to issue system control and informational commands.
- **IO**—authority to issue input/output and informational commands.
- **CONSOLE**—authority to issue console control and informational commands.
- **INFO**—authority to issue informational commands; this value cannot be specified with any other value.

AUTO(YES|NO)
Specify YES or NO to indicate whether to receive unsolicited messages.

MFORM(*message-format*)
The format in which messages are displayed:

- **TIME**—messages include a timestamp.
- **SYSID**—messages include the system ID.
- **JOBNAME**—messages include the jobname.
- **MESSAGE**—message text is displayed.
- **EXEMPT**—messages that are exempt from jobname and system name formatting are ignored.

MSGLEVEL(*msglevel*)
The messages to be received by this console:

- **OPER_REPLY**—operator reply messages.
- **IMMEDIATE**—immediate action messages.
- **CRITICAL_EVENTUAL**—critical eventual action messages.
- **EVENTUAL**—eventual action messages.

- **INFO**—informational messages.
- **NO_BROADCAST**—do not receive broadcast messages.
- **ALL**—operator reply, immediate, critical eventual, eventual and informational messages; this value cannot be specified with any value other than NO_BROADCAST.

MONITOR(*monitor*)
Options when monitoring jobs, TSO users or data set status:

- **JOBNAMES**—display information about each job without start and end times.
- **JOBTIME**—display information about each job with start and end times.
- **TSOSESS**—display information about each TSO session without start and end times.
- **TSOTIME**—display information about each TSO session of each TSO session including start and end with start and end times.
- **STATUS**—display data set status information.

ROUTCODE(*routcode*)
The routing codes associated with this console session; multiple routing codes (n1,n2,n3) or ranges (n1-n3) can be specified where the values are 1 through 128.

LOGCMD(YES|NO)
Specify YES or NO as to whether command responses are to be logged in the hard copy log.

MIGID(YES|NO)
Specify YES or NO as to whether a one-byte migration ID is to be assigned to this console.

DOM(*delete-operator-message*)
Specify which delete operator messages this console is to receive:

- **NORMAL**—receive all appropriate DOM requests.
- **ALL**—receive all DOM requests from the SYSPLEX.
- **NONE**—do not receive any DOM requests.

UD(YES|NO)
Specify YES or NO as to whether undelivered messages are to be received.

KEY(*console-key*)
A one to eight-character console key.

CMDSYS(*system-name*)
A one to eight-character system name to which commands issued from this console are to be sent. Specify * for the local system.

ALTGROUP(*altgroup*)
A one to eight-character console group to be used in recovery.

OPERMSCP(*opermscp*)
A list of system names from which this console can receive messages that are not directed to a specific console: specify * for the local system; specify *ALL for all systems.

Example

The following example shows the creation of a USER profile record:

```
SET PROFILE(USER) DI(OPERPARM)

INSERT USER1 AUTH(MASTER) ROUTCODE(1-8,11) MFORM(TIME,SYSID)
```

This command changes console authority to INFO:

```
CHANGE USER1 AUTH(INFO) REP
```

To change MFORM to delete SYSID (leaving TIME), use one of the following commands:

```
CHANGE USER1 MFORM(SYSID) DEL
```

Or:

```
CHANGE USER1 MFORM(TIME) REP
```

If the PROFILE USER records are resident, to activate new or changed profile records, issue the following command:

```
F ACF2,REBUILD(USR),CLASS(P)
```

OPERCMDS Resource Rules

The user's OPERPARM profile is used only when access to the OPERCMDS entity MVS.MCSOPER.*userid* is allowed. When access is not allowed, the default console attributes are used. To allow OPERPARM profiles to be used for some users, compile the following resource rule (assuming that CLASMAPS specifies OPR as the resource type for OPERCMDS):

```
SET R(opr)

COMPILE *

$KEY(MVS) TYPE(opr)
 MCSOPER.user1 UID(-) ALLOW
 MCSOPER.user2 UID(-) ALLOW
 MCSOPER.- UID(-) PREVENT

STORE
```

Masking the key of a resource rule requires an entry in the GSO INFODIR record to make these rules resident. To create this entry, issue the following commands:

```
SET C(GSO)

CHANGE INFODIR TYPES(R-Ropr) ADD
```

To activate the updated INFODIR record, issue this command:

```
F ACF2,REFRESH(INFODIR)
```

To activate new or changed resource records, issue this command:

```
F ACF2,REBUILD(OPR),CLASS(R)
```

Password Profile Records

The PASSWORD segment of the USER profile is used to retain previous passwords and Kerberos passwords. Previous passwords are stored in this record when the GSO PSWD record specifies PSWXHIST and PSWXHST# = 5 or more. Kerberos passwords are stored in this record when the GSO OPTS record specifies KERBLVL(1). Most fields in this record are not displayed. Users can list and delete, but cannot insert or change this record. It is maintained internally by eTrust CA-ACF2.

Record ID	Fields
logonid	#PSWDCNT #PWD-TOD

Field Descriptions

recid
The eTrust CA-ACF2 logonid

#PSWDCNT
Specifies the number of previous passwords stored in the record for extended password history. Users cannot change this field. The #PSWDCNT field is always one count less than the maximum number named in the PSWXHST# field of the GSO PSWD record. This is due to the fact that the current password is included in the count for PSWXHST# and the User Profile password record keeps all but the current password.

#PWD-TOD
Specifies the date and time when the user's most recent old password was saved. eTrust CA-ACF2 lists the date in the format mm/dd/yy, dd/mm/yy, or yy/mm/dd, depending on the DATE field of the GSO OPTS record. You cannot set this field. eTrust CA-ACF2 maintains and displays it. This field should match the PSWD-TOD in the Logonid record. Users cannot change this field.

SECLABEL Profile Records

The SECLABEL segment of the USER Profile is used to assign security labels to users in an MLS environment. User SECLABEL Profile Records are inserted using the logonid or a logonid mask as the record key. The SECLABEL segment of the USER Profile Record specifies one or more SECLABEL Profile Data Records to be used to assign security labels to a user. You must define SECLEVEL, CATEGORY, and SECLABEL segments before you can create valid USER SECLABEL Profile Records.

WARNING! *If you change or delete an existing security label (for example Seclabel Profile data record) that has been assigned to users or resources, you may get unexpected results during MLS validation. Before changing or removing a security label from the system, check whether it has been assigned to any users or resources. If it has, confirm that the change or deletion is intended. If it is, make any necessary changes to user and resource Seclabel profile records that are using the security label. Likewise, if you delete a security level or category that is used in any existing security label, before removing the level or category from the system, confirm that the deletion is intended. If it is, make any necessary changes to existing security labels, and any user and resource Seclabel profile records that are using the security labels.*

System-defined Labels

eTrust CA-ACF2 provides three system-defined, internal security labels that can never be directly created or modified by a user but can be assigned to users: **SYSHIGH**, **SYSLOW**, and **SYSMULTI**.

For more information on how to implement MLS on a system, including defining, assigning, using security classifications, and using system-defined security labels, see the *Multilevel Security Planning Guide*.

Command Syntax

```
{Insert|Change|List|Delete}{logonid|logonidmask}
    seclabel(seclabel1,…seclabeln)
    deflabel(seclabel)
{add|rep|del}
```

Record ID	Fields
logonid|*logonidmask*	SECLABEL(*seclabel1,…seclabeln*) DEFLABEL(*seclabel*|SYSLOW)

Field Descriptions

logonid | logonidmask
Specifies one or more userids that are to be assigned security labels. This field is required and can be masked. The asterisk(*) is the only valid masking character. You cannot use the dash(-) as a masking character. When a dash occurs, eTrust CA-ACF2 treats it as a literal character. Masking of the logonid follows the same rules as masking for resource rules. You can use masking to ease administration by assigning security labels to many users at one time.

Seclabel(*seclabel1,...seclabeln*)
Specifies the security labels which a user is authorized to use when entering a system and that will be used during validation to determine whether access to classified MLS data sets and resources will be allowed or denied. The *seclabel* value is the 1- to 8-character uppercased name of an existing SECLABEL Profile Data record segment that contains the security label data. You may assign more than one security label to a user, but only one label may be active at a time and used to validate MLS access to data sets and resources. If multiple security labels are assigned, any of these are available to the user to signon to a system. This field is required and cannot be masked. A comma or blank is the only valid delimiter between specified security label values. The system-defined security label SYSNONE is not valid for a user.

Deflabel(*seclabel* | SYSLOW)
Specifies the name of a security label that will be active and used to validate MLS access if a security label is not specified at system entry when MLS is active. The *seclabel* value is the 1- to 8-character name of an existing SECLABEL Profile Data Record segment that contains the security label data. This field is required. The default value is the system-defined label, SYSLOW, which is always the lowest security label defined by the system and will be dominated by all other security labels.

Creating a User SECLABEL Profile Record

To create a User SECLABEL Profile Record, you must have the SECURITY privilege in your logonid. Issue the following commands to create the record:

```
acf
 ACF
set profile(user) division(seclabel)
 PROFILE
insert usera seclabel(label2) deflabel(syslow)
 SECLABEL / USERA LAST CHANGED BY M1ADMN ON 04/22/04-15:14
                      DEFLABEL(SYSLOW) SECLABEL(LABEL2)
PROFILE
```

Note: Any security label specified in the record must be valid (defined in the system) for the record to be successfully inserted.

Viewing a User SECLABEL Profile Record

To view a User SECLABEL Profile Record, you must have the SECURITY privilege in your logonid. Issue the following commands to view the record:

```
acf
 ACF
set profile(user) division(seclabel)
 PROFILE
list usera
 SECLABEL / USERA LAST CHANGED BY M1ADMN ON 04/22/04-15:14
                      DEFLABEL(SYSLOW) SECLABEL(LABEL2)
PROFILE
```

Changing a User SECLABEL Profile Record

To change a User SECLABEL Profile Record, you must have the SECURITY privilege in your logonid. Issue the following commands to change the record:

```
acf
 ACF
set profile(user) division(seclabel)
 PROFILE
change usera seclabel(label30) rep
 SECLABEL / USERA LAST CHANGED BY M1ADMN ON 04/22/04-15:17
                      DEFLABEL(SYSLOW) SECLABEL(LABEL30)
PROFILE
```

Note: Any security label specified in the record must be valid (defined in the system) for the record to be successfully changed.

Deleting a User SECLABEL Profile Record

To delete a User SECLABEL Profile record, you must have the SECURITY privilege in your logonid. Issue the following commands to delete the record:

```
acf
 ACF
set profile(user) division(seclabel)
 PROFILE
delete usera
 ACF6D073 SECLABEL /USERA RECORD DELETED
 PROFILE
```

Activating a User SECLABEL Profile Record

To activate the security labels assigned to users, issue the following commands:

```
set control(gso)
 CONTROL
 change infodir types(r-pusr)
 f acf2,refresh(infodir)
 f acf2,rebuild(usr),c(p)
```

PROXY Profile Data Records

Specifies information that the z/OS LDAP Server will use when acting as a proxy on behalf of a requester. The R_proxyserv SAF callable service will attempt to retrieve this information when it is not explicitly supplied through invocation parameters. Applications or other services that use the R_proxyserv callable service, such as IBM Policy Director Authorization Services for z/OS, may instruct their invokers to define PROXY segment information.

Record ID	Fields
Recid	BINDDN(*binddn*) BINDPTOD(00/00/00-00:00) BINDPW(*password*) LDAPHOST(*ldapurl*)

Field Descriptions

BINDDN(*binddn*)

Specifies the distinguished name (DN) which will be used in conjunction with the BIND password if the LDAP Server needs to supply an administrator or user identity to BIND with another LDAP Server. The *binddn* value can be 1-1023 characters in length. A DN is made up of attribute value pairs, separated by commas. For example:

'cn=Tom Brady,ou=Quarterback,o=New England,c=US'

'cn=Sammy Sosa,ou=Slugger,o=Chicago Cubs,c=US'

Note: The list of attribute pairs specified in the BINDDN field are considered a single value, therefore, it must be enclosed in single quotes. The single quotes will qualify the entire list of attribute value pairs as one value.

eTrust CA-ACF2 does not validate a valid BIND DN was entered.

BINDPTOD(00/00/00-00:00)

Specifies the date and time the BINDPW field was last changed. eTrust CA-ACF2 lists the date in the format mm/dd/yy, dd/yy, depending on the DATE field of the GSO OPTS record. You cannot set this field. eTrust CA-ACF2 maintains and displays it. (Eight-byte binary filed)

BINDPW(*binddn*)

Specifies the password for the DN defined in the BINDDN parameter. The BIND password can contain 1-128 characters. Both uppercase and lowercase are accepted and maintained in the case in which they are entered.

eTrust CA-ACF2 does not validate that a valid BIND password was entered.

LDAPHOST(*ldapurl*)

Specifies the URL of the LDAP server that the z/OS LDAP Server will contact when acting as a proxy on behalf of a requester. An LDAP URL has a format such as ldap://123.45.6:389 or ldaps://123.45.6:636, where ldaps indicates that an SSL connection is desired for a higher level of security. LDAP will also allow you to specify the host name portion of the URL using either the text form (LDAP.HOST.CA.COM) or the dotted decimal address (111.222.33.44). The port number is appended to the host name, separated by a colon ':'. See your LDAP server documentation for information on how to set up the LDAP server for SSL connections.

This value must be at least 10 bytes long and can be up to 1023 bytes long. It must start with either ldap:// or ldaps://. Any characters may be entered in the remaining portion of the URL, but you should ensure that the URL conforms to TCP/IP conventions. Normally, characters such as commas, blanks, parenthesis, semicolons and single quotes are not allowed in a host name. These characters will be accepted if the LDAPHOST is enclosed in single quotes.

eTrust CA-ACF2 does not validate that the contents of the URL are valid.

WORKATTR Profile Data Records

WORKATTR profile data information is extracted by APPC/MVS for TAILOR_SYSOUT and TAILOR_ACCOUNT processing. Name and location information is displayed on the banner pages of transaction program generated SYSOUT. Account information is passed to z/OS exits and used by SMF.

Record ID	Fields
Recid	WANAME(*name*) WABLDG(*bldg*) WADEPT(*dept*) WAROOM(*room*) WAADDR1(*addr1*) WAADDR2(*addr2*) WAADDR3(*addr3*) WAADDR4(*addr4*) WAACCNT(*accnt*)

Field Descriptions

recid

A one to eight-character maskable userid. Normal logonid masking conventions apply, except that the mask can contain only asterisks (*), not a dash (-).

- **WANAME**—A one to 60-character user name.
- **WABLDG**—A one to 60-character building name.
- **WADEPT**—A one to 60-character department name.
- **WAROOM**—A one to 60-character room name.
- **WAADDR1**—A one to 60-character address line.
- **WAADDR2**—A one to 60-character address line.
- **WAADDR3**—A one to 60-character address line.
- **WAADDR4**—A one to 60-character address line.
- **WAACCNT**—A one to 255-character account number.

Example

The following example shows the creation of a WORKATTR profile data record:

```
SET PROFILE(USER) DIVISION(WORKATTR)

INSERT userid WANAME(username) WAROOM(roomid) WAACCNT(123,456)
```

Using the ISPF Panels

This section provides a brief overview of how to use the eTrust CA-ACF2 ISPF panels to maintain logonid records.

Note: To use the ISPF panels that are shipped with eTrust CA-ACF2, you must install them. When they are active, select the option from the ISPF/PDF Primary Option Menu.

To create a logonid record, select option 2 LOGONIDS from the eTrust CA-ACF2 ISPF Option Selection Menu.

```
----------------- eTrust CA-ACF2 ISPF OPTION SELECTION MENU ------------------
OPTION  ===>
   1  RULES     - PROCESS eTrust CA-ACF2 ACCESS AND GENERALIZED RESOURCE RULES
   2  LOGONIDS  - eTrust CA-ACF2 LOGONID CREATION/MAINTENANCE FACILITY
   3  SYSTEM    - eTrust CA-ACF2 SHOW COMMANDS
   4  REPORTS   - eTrust CA-ACF2 REPORT PROGRAM PROCESSOR
   5  UTILITIES - PROCESS eTrust CA-ACF2 UTILITIES
   6  GSO       - GLOBAL SYSTEM OPTIONS SERVICES
   7  NET       - NETWORKING SYSTEM OPTIONS SERVICES
   8  CAC       - MVS DATABASE CACHE RECORD SERVICES
   9  XREF      - SOURCE/RESOURCE AND GROUP RECORD SERVICES
  10  MAC       - MANDATORY ACCESS CONTROL ADMINISTRATION
  11  CPF       - COMMAND PROPAGATION FACILITY SERVICES
  12  FIELD     - RECORD LEVEL PROTECTION CONTROLS
  13  TARGETS   - SET CPF TARGET NODES, DEFAULTS IN USE
  14  PROFILE   - PROCESS PROFILE INFORMATION RECORDS
  15  SMS       - PROCESS DFSMS SUPPORT RECORDS
  16  ENTRY     - PROCESS ENTRY SOURCE RECORDS
  17  SHIFT     - PROCESS SHIFT/ZONE RECORDS
  18  RACDCERT  - PROCESS KEYRING/CERTIFICATE COMMANDS
  19  C-CIC     - PROCESS C-CIC CICS INITIALIZATION RECORDS
  20  LDS       - PROCESS LDAP DIRECTORY SERVICES
```

The eTrust CA-ACF2 Logonid Maintenance panel is displayed.

```
----------------- eTrust CA-ACF2 SECURITY LOGONID MAINTENANCE -----------------
OPTION  ===>

   1  ADD     - INSERT A NEW LOGONID (TSO, BATCH, ETC.)

   2  ADD     - INSERT A NEW CICS/IMS LOGONID

   3  CHANGE  - CHANGE A LOGONID

   4  DELETE  - DELETE AN EXISTING LOGONID

   5  LIST    - LIST A LOGONID

   6  SUSPEND - CANCEL/SUSPEND A LOGONID

   7  RESET   - RESET A LOGONID PASSWORD/VIOLATIONS

   8  SCOPE   - CREATE/MAINTAIN SCOPE RECORDS FOR LOGONIDS

   9  TARGETS - SET CPF TARGET NODES, DEFAULTS IN USE
```

The sections that follow describe the panels for options 1-7. Option 8, SCOPE is described in the "Maintaining Scope Records" chapter. Option 9, TARGETS is described in the "Using the Command Propagation Facility" chapter.

Note: To process an ACF command at a node other than the default node or the current target setting, select the TARGET option and specify the target nodes **before** you select the ADD, CHANGE, LIST, DELETE, SUSPEND, or RESET options.

Creating Logonid Records

To add a logonid record to the Logonid database, select option 1 ADD from the eTrust CA-ACF2 Logonid Maintenance panel. The Add A New Logonid panel is displayed. Use the tab keys and type the appropriate values in the fields.

```
------------------------------ ADD A NEW LOGONID ------- ENTER REQUIRED FIELD
COMMAND ===>

IDENTIFICATION SECTION

          LID ===>
USING LOGONID ===>
    USER NAME ===>
        PHONE ===>
     PASSWORD ===>

CANCEL/SUSPEND SECTION
(ENTER Y OR N FOR THE FOLLOWING FIELDS, BLANK DEFAULTS TO N)

       CANCEL ===>                  SUSPEND ===>
      MONITOR ===>                  MON-LOG ===>
       PP-TRC ===>                  PP-TRCV ===>
        TRACE ===>                  TSO-TRC ===>

PRESS ENTER TO CONTINUE OR END TO CANCEL LOGONID CREATION.
```

Panel Field Descriptions

The following list describes the fields in the IDENTIFICATION and CANCEL/SUSPEND sections of this panel and how to specify them:

LID
Specify the one to eight-character logonid of the user.

USING LOGONID
Specify a model logonid. When you specify a model logonid, you are telling eTrust CA-ACF2 that you want to create a new logonid based on the model. Specify only those fields that differ from the model logonid for the new logonid.

USER NAME
Specify the name of the user.

PHONE
Specify the user's phone number.

PASSWORD

Specify a password for the user. Passwords are suggested for all logonids. If you do not specify a password when creating a logonid, eTrust CA-ACF2 issues a message informing you of this requirement and the insert fails if PSWDREQ is set in the GSO PSWD record. At most sites, this password is used only once—the first time the user logs on to the system. After that the user can change his password so that he is the only person who knows it.

Beginning with the CANCEL/SUSPEND section and continuing for the next several panels, enter **Y** to activate the field. If you do not want a user to have the privilege, leave the field blank. For descriptions of the fields, see Logonid Record Fields in this chapter.

To continue selecting privileges, press Enter.

```
------------------------------ Add a New Logonid ---------------------------

COMMAND ===>

 Privileges Section for LOGONID

  ACTIVE   ===>             DSNSCOPE ===>             EXPIRE   ===>
  LIDSCOPE ===>             NOSPOOL  ===>             PROGRAM  ===>
  SCPLIST  ===>             SYNCNODE ===>             SYNERR   ===>
  UIDSCOPE ===>

Enter Y to allow the following privileges or N to disallow. N is the default

 ACCOUNT  ==>   AUDIT    ==>   AUTOALL  ==>   AUTODUMP ==>   AUTONOPW  ==>
 AUTOONLY ==>   BDT      ==>   CICS     ==>   CMD-PROP ==>   CONSULT   ==>
 DG84DIR  ==>   DIALBYP  ==>   DUMPAUTH ==>   GRP-OPT  ==>   GRPLOGON  ==>
 IDMS     ==>   IMS      ==>   JOB      ==>   JOBFROM  ==>   LDS       ==>
 LDEV     ==>   LEADER   ==>   LOGSHIFT ==>   MAINT    ==>   MUSASS    ==>
 NO-INH   ==>   NO-OMVS  ==>   NO-SMC   ==>   NO-STORE ==>   NOMAXVIO  ==>
 NON-CNCL ==>   PPGM     ==>   PRIV-CTL ==>   READALL  ==>   REFRESH   ==>
 RESTRICT ==>   RSRCVLD  ==>   RULEVLD  ==>   SECURITY ==>   SRF       ==>
 STC      ==>   SUBAUTH  ==>   TAPE-BLP ==>   TAPE-LBL ==>   TDISKVLD  ==>
 TSO      ==>   VLDVMACT ==>   VM       ==>   VMD4AUTH ==>   VMD4RSET  ==>
 VMD4TARG ==>   VMSAF    ==>   VMSFS    ==>   VMXA     ==>   VSESRF    ==>

 Press ENTER to display next panel or END to redisplay previous panel.
```

After you complete the entries for this panel, press Enter and specify values for the other fields.

```
------------------------------ Add a New Logonid ----------------------------
 COMMAND ===>

  Password Section for LOGONID
   MAXDAYS  ===>          MINDAYS  ===>          PSWD-EXP ===>    ( Y OR N )

  TSO Section
   ATTR2    ===>          CHAR     ===>          DFT-DEST ===>
   DFT-PFX  ===>          DFT-SOUT ===>          DFT-SUBC ===>
   DFT-SUBH ===>          DFT-SUBM ===>          LINE     ===>
   TSOACCT  ===>
   TSOCMDS  ===>          TSOPERF  ===>          TSOPROC  ===>
   TSORGN   ===>          TSOSIZE  ===>          TSOTIME  ===>
   TSOUNIT  ===>
   ( Enter 'Y' for YES to allow TSO fields -  NO is the default )
   ACCTPRIV ===>       ALLCMDS  ===>       CMD-LONG ===>       CONSOLE  ===>
   INTERCOM ===>       JCL      ===>       LGN-ACCT ===>       LGN-DEST ===>
   LGN-MSG  ===>       LGN-PERF ===>       LGN-PROC ===>       LGN-RCVR ===>
   LGN-SIZE ===>       LGN-TIME ===>       LGN-UNIT ===>       MAIL     ===>
   MODE     ===>       MOUNT    ===>       MSGID    ===>       NOTICES  ===>
   OPERATOR ===>       PAUSE    ===>       PMT-ACCT ===>       PMT-PROC ===>
   PROMPT   ===>       RECOVER  ===>       TSOFSCRN ===>       UNICNTR  ===>
   VLD-ACCT ===>       VLD-PROC ===>       WTP      ===>
  Press ENTER to display next panel or END to redisplay previous panel.
```

After you complete the entries for this panel, press Enter and specify values for the other fields.

```
------------------------------ Add a New Logonid ----------------------------
 COMMAND ===>

 CICS Section for LOGONID
   ACF2CICS ===>   (Y or N) CICSOPT  ===>            CICSCL   ===>
   CICSID   ===>            CICSKEY  ===>            CICSKEYX ===>
   CICSPRI  ===>            CICSRSL  ===>            IDLE     ===>

 IDMS Section
   IDMSPROF ===>                                     IDMSPRVS ===>

 MUSASS Section
   MUSID    ===>            MUSOPT   ===>            MUSPGM   ===>
   MUSUPDT  ===>   (Y or N) MUSIDINF ===>   (Y or N) MUSDLID  ===>
   NO-STATS ===>   (Y or N)

 Restrictions Section
   GROUP    ===>            PREFIX   ===>            SHIFT    ===>
   SOURCE   ===>            VMACCT   ===>            ZONE     ===>
   VMIDLEMN ===>            VMIDLEOP ===>
   ( Enter 'Y' for YES to specify restrictions - NO is the default )
   AUTHSUP1 ===>      AUTHSUP2 ===>      AUTHSUP3 ===>      AUTHSUP4 ===>
   AUTHSUP5 ===>      AUTHSUP6 ===>      AUTHSUP7 ===>      AUTHSUP8 ===>

User Fields===>
```

Press ENTER to add logonid or END to redisplay previous panel.After you complete the entries for this panel, press Enter to add the logonid.

Creating a CICS or IMS Logonid

To create a logonid for the eTrust CA-ACF2 CICS or eTrust CA-ACF2 IMS interfaces, select option 2 Add A New CICS/IMS Logonid from the eTrust CA-ACF2 Logonid Maintenance panel. The Add A New CICS Or IMS Logonid panel is displayed. This panel is exactly like the one for Add A New Logonid. Use the tab keys and type the appropriate values in the fields.

```
------------------------ ADD A NEW CICS OR IMS LOGONID -----------------------

COMMAND ===>

IDENTIFICATION SECTION

          LID ===>
USING LOGONID ===>
    USER NAME ===>
        PHONE ===>
     PASSWORD ===>

CANCEL/SUSPEND SECTION

(ENTER Y OR N FOR THE FOLLOWING FIELDS, BLANK DEFAULTS TO N)

       CANCEL ===>    MON-LOG ===>    MONITOR ===>    SUSPEND ===>
       PP-TRC ===>    PP-TRCV ===>    TRACE   ===>    TSO-TRC ===>

PASSWORD SECTION

     MAXDAYS  ===>     MINDAYS  ===>     PSWD-EXP ===>    ( Y OR N )

PRESS ENTER TO CONTINUE OR END TO CANCEL LOGONID CREATION.
```

Panel Field Descriptions

The following list describes the fields in the panel and how to specify them:

LID
Specify the one to eight-character logonid of the user here.

USING LOGONID
Specify a model logonid. When you specify a model logonid, you are telling eTrust CA-ACF2 that you want to create a new logonid based on the model. Specify only those fields that differ from the model logonid for the new logonid.

USER NAME
Specify the name of the user.

PHONE
Specify the user's phone number.

PASSWORD

Specify a password for the user. Passwords are required for all logonids if PSWDREQ is set on in the GSO PSWD record. If you do not specify a password when creating a logonid, eTrust CA-ACF2 issues a message informing you of this requirement and the insert fails. At most sites, this password is used only once—the first time the user logs on to the system. After that the user can change his password so that he is the only person who knows it.

Beginning with the CANCEL/SUSPEND section and continuing for the next several panels, enter **Y** to activate the field. If you do not want a user to have the privilege, leave the field blank. For descriptions of the fields, see Logonid Record Fields later in this chapter.

The fields on the remaining panels are exactly the same as those for Add A New Logonid. See the previous section to view sample panels.

After you complete the entries for this panel, press Enter and specify values for the other fields.

Changing a Logonid

To change a logonid record, select option 3 Change A Logonid from the eTrust CA-ACF2 Logonid Maintenance panel. The Change A Logonid panel is displayed. Use the tab keys and type the appropriate values in the fields.

```
------------------------------ CHANGE A LOGONID -------------------------------
COMMAND ===>

IDENTIFICATION SECTION

     LOGONID  ===>          - OR - UID STRING ===>
                  (LOGONID OR UID STRING Can be masked)
           IF ===>
    User Name ===>
        Phone ===>
     Password ===>                           Verify Password ===>

Cancel/Suspend Section
(ENTER Y or N for the following fields, blank leaves value unaltered)

       CANCEL ===>    MON-LOG ===>    MONITOR ===>      SUSPEND ===>
       PP-TRC ===>    PP-TRCV ===>      TRACE ===>      TSO-TRC ===>

Violation Statistics Section

     PSWD-DAT ===>          PSWD-INV ===>         PSWD-VIO ===>
      SEC-VIO ===>

Press Enter to continue or End to cancel logonid update.
```

Panel Field Descriptions

The following list describes the fields in the panel and how to specify them:

LOGONID
Specify the logonid record you want to change or a mask. If you specify a mask, any other fields you specify apply to all logonid records that match the mask.

UID STRING
Specify a UID you want to change or a mask. If you specify a mask, any other fields you specify apply to all UIDs that match the mask.

IF
Specify logonid record bit fields here only if you specified a logonid or UID mask. When you specify bit fields here, you indicate that you want to change all logonid records that match the mask if they also contain the fields you specify here. If they do not, eTrust CA-ACF2 does not change the logonid records.

USER NAME
Specify the name of the user.

PHONE
Specify the user's phone number.

PASSWORD
Specify a password for the user. Passwords are required for all logonids. If you do not specify a password when changing a logonid, eTrust CA-ACF2 issues a message informing you of this requirement and the change fails. At most sites, this password is used only once—the first time the user logs on to the system. After that the user can change his password so that he is the only person who knows it.

Beginning with the CANCEL/SUSPEND section and continuing for the next several panels, enter **Y** to activate the field. If you do not want a user to have the privilege, leave the field blank. For descriptions of the fields, see Logonid Record Fields later in this chapter. After you complete the entries for this panel, press Enter and specify values for the other fields.

VERIFY PASSWORD
Specify a password verification whenever a user requests a new password.

Deleting Logonid Records

To remove a logonid record from the Logonid database, select option 4 DELETE from the eTrust CA-ACF2 Logonid Maintenance panel. The Delete A Logonid panel is displayed. Use the tab keys and type the appropriate values in the fields.

```
------------------------------ DELETE A LOGONID ------------------------------
COMMAND ===>

ENTER DESIRED DELETE FUNCTION:

LOGONID ===>                           ( LOGONID OR LOGONID MASK )
 - OR -
UID     ===>                           ( UID OR UID MASK )

IF      ===>                           ( NAMES OF BIT FIELDS )


RULE/NORULE   ===> R            ENTER N TO NOT DELETE RULE SETS
DEFAULT VALUE: RULE                   FOR USER WITH THIS HIGH LEVEL INDEX
```

Panel Field Descriptions

The panel fields are described in the following:

LOGONID
Specify the logonid record you want to delete or a mask. If you specify a mask, eTrust CA-ACF2 deletes all logonid records that match the mask.

UID
Specify the UID for the logonid records you want to delete or a mask. If you specify a mask, eTrust CA-ACF2 deletes all records that match the mask. If you also specify the IF field, eTrust CA-ACF2 deletes the logonid records for the UIDs that match the mask if they have the fields you specified for the IF field.

IF
Specify logonid record bit fields here only if you specified a logonid or UID mask. When you specify fields here, you indicate that you want to delete all logonid records that match the mask if they also contain the fields you specify here. If they do not, eTrust CA-ACF2 does not delete the logonid records.

RULE/NORULE
Specify Y to delete all rules from the Rule database if the high-level qualifier matches the logonid you want to delete. RULE is the default. Specify N to leave the rule in place.

After you complete the entries for this panel, press Enter. eTrust CA-ACF2 deletes the logonid or group of logonids you specified.

Displaying Logonid Records

To display a logonid record in the Logonid database, select option 5 LIST from the eTrust CA-ACF2 Logonid Maintenance panel. The List A Logonid panel is displayed. Use the tab keys and type the appropriate values in the fields.

```
-------------------------------- LIST A LOGONID -----------------------------
COMMAND ===>


ENTER DESIRED LIST FUNCTION:

LOGONID ===>                                    LOGONID OR LOGONID MASK
IF      ===>                                    NAMES OF BIT FIELDS
UID     ===>                                    UID OR UID MASK
SECTION ===>                                    LOGONID SECTIONS TO LIST
PROFILE ===>                                    USER PROFILES TO LIST

ENTER SET MODE FUNCTION:

VERBOSE/TERSE   ===> V            ENTER: V - VERBOSE OR T - TERSE
TRIVIA/NOTRIVIA ===> T                   T - TRIVIA  OR N - NOTRIVIA
DEFAULT VALUES: VERBOSE AND TRIVIA
```

Panel Field Descriptions

The following list describes the fields in the panel and how to specify them:

LOGONID
Specify the logonid record you want to view or a mask. If you specify a mask, eTrust CA-ACF2 displays all logonid records that match the mask.

IF
Specify logonid record fields here only if you specified a logonid or UID mask. When you specify fields here, you indicate that you want to view all logonid records that match the mask if they also contain the fields you specify here. If they do not, eTrust CA-ACF2 does not display the logonid records. This field can be bit fields or value fields such as MAXDAYS=0.

UID
Specify a UID you want to view or a mask. If you specify a mask, eTrust CA-ACF2 displays all logonid records that match the mask. If you also specify the IF field, eTrust CA-ACF2 displays the logonid records for the UIDs that match the mask and have the fields you specified for the IF field.

SECTION(*name,...name*)
Specify the name of one or more sections to be displayed from the logonid or logonids selected. name can be one of the following:

- **ALL**—displays the logonid header information and information from both the logonid record and profile record for the selected logonid.
- **HEADER**—displays only header information for the selected logonid. Enter this keyword only if just the header section is desired. For all other requests, header information displays automatically.

- ***logonid group name***—displays the identifying header information for the selected logonid and information contained in the specified section of the Logonid record.

If you do not specify one of these options for SECTION, eTrust CA-ACF2 displays header and logonid information.

PROFILE(*name,...name*)
Specifies the name of one or more user profile data records to be displayed for the logonid or logonids selected. name can be one of the following:

- **ALL**—displays information contained in all user profile records for the selected logonid.
- ***user profile type***—displays only the specified user profile data records for the selected logonid.

VERBOSE|TERSE
Specify T to view only the first line of each logonid record. You specify which fields appear in this display in the @HEADER macro of the ACFFDR. See Appendix A, "eTrust CA-ACF2 Field Definition Record Generation," in the *Getting Started* guide for more information.

Specify V to view the full logonid record. V is the default.

TRIVIA|NOTRIVIA
Specify T to view the full logonid record, if you also specify VERBOSE. If you specify TERSE, eTrust CA-ACF2 displays only the header information.

Specify N to view only those logonid fields that you specify in the FLAGS=LIMIT operand of the @CFDE macro in the ACFFDR. See Appendix A, "eTrust CA-ACF2 Field Definition Record Generation," in the *Getting Started* guide for more information. T is the default.

After you complete the entries for this panel, press Enter. eTrust CA-ACF2 displays the logonid or group of logonids you specified.

Suspending or Canceling a Logonid

When a user goes on vacation, sick leave, or resigns, you should suspend his logonid record. You should not immediately delete his logonid. Depending on your site, you might be asked to determine what types of access he had and to assign that access to another user. You might also want to suspend the logonid of a user after you determine he is trying to circumvent your security implementation.

No matter what the case, you can always reactivate a suspended logonid by changing the logonid record. If you delete a logonid record, you must create it again.

To suspend a logonid, select option 6 SUSPEND from the eTrust CA-ACF2 Logonid Maintenance panel. The Cancel/Suspend A Logonid panel is displayed. Use the tab keys and type the appropriate values in the fields.

```
-------------------------- CANCEL/SUSPEND A LOGONID -------------------------
COMMAND ===>

ENTER DESIRED OPTION:                    C - CANCEL     S - SUSPEND

LOGONID ===>                             (LOGONID OR LOGONID MASK)
 - OR -
UID     ===>                             (UID OR UID MASK)
```

Panel Field Descriptions

The panel fields are described in the following:

LOGONID
Specify the logonid record you want to cancel or a mask. If you specify a mask, eTrust CA-ACF2 cancels all logonid records that match the mask.

UID
Specify the UID for the logonid records you want to cancel or a mask. If you specify a mask, eTrust CA-ACF2 cancels all records that match the mask.

After you complete the entries for this panel, press Enter. eTrust CA-ACF2 cancels the logonid or group of logonids you specified.

Resetting a Password

From time to time, users enter expired or incorrect passwords and can be suspended from entering the system. Only a security administrator or account manager with the appropriate scope can reset their password.

To reset a password for a logonid record, select option 7 RESET from the eTrust CA-ACF2 Logonid Maintenance panel. The Reset A Logonid panel is displayed. Use the tab keys and type the appropriate values in the fields.

```
-------------------------------- RESET A LOGONID ----------------------------
OPTION  ===>

    1  PASSWORD   - RESET USER'S PASSWORD
    2  VIOLATIONS - RESET USER'S PASSWORD VIOLATION COUNT


    LOGONID ===>               PASSWORD ===>
```

Panel Field Descriptions

The panel fields are described in the following:

OPTION
Type 1 to reset a password for a logonid. Type 2 to reduce the password violation count to zero if that user has surpassed the threshold value for your site.

LOGONID
Specify the logonid of the person who asked you to reset his password.

PASSWORD
Specify the password that the user must enter to gain access.

After you complete the entries for this panel, press Enter. eTrust CA-ACF2 resets the password or reduces the password violation count for the logonid or group of logonids you specified.

Using the ACF Command

The ACF command has many command settings. The command setting determines the type of eTrust CA-ACF2 record you can process. You can process logonid records and user profile records by entering the TSO ACF command or by establishing the LID setting of the ACF command. After you establish the ACF command setting you can issue any of the following ACF subcommands:

- ACCESS
- *CHANGE
- CHKCERT
- CONNECT
- *DELETE
- END
- EXPORT
- GENCERT
- GENREQ
- HELP
- *INSERT
- *LIST
- MLSLABEL
- MLWRITE

- QUIT
- REKEY
- REMOVE
- ROLLOVER
- SET or T
- SHOW
- SN
- *SYNCH

The common subcommands ACCESS, CHKCERT, CONNECT, END, EXPORT, QUIT, GENCERT, GENREQ, MLSLABEL, MLWRITE, REKEY, REMOVE, ROLLOVER, HELP, SET, SHOW, and SN operate under all settings. The other subcommands, marked by asterisks (*), are described in the following sections.

You can also execute the ACF command from TSO or in batch using ACFBATCH or the batch TMP (IKJEFT01). These options are described in The ACF Command in Batch section in the "Overview of eTrust CA-ACF2" chapter.

This section describes how to use the INSERT, SYNCH, CHANGE, LIST, and DELETE subcommands to maintain logonid records.

Logonid and User Profile Administration

eTrust CA-ACF2 allows for the processing of both logonid and user profile records in LID setting. User profile records can be created and deleted with a logonid request. eTrust CA-ACF2 distinguishes the particular logonid or user profile records to process and performs the appropriate administrative request.

User profile records are associated with a user of the system. User profile CERTDATA and KEYRING records might contain suffixes if a user is defined with more than one record. The record key contains a userid with a distinguishing suffix separated with a period. Because a user profile record with a suffix cannot be distinguished in LID mode, CERTDATA and KEYRING records cannot be inserted in LID mode. The records can be directly administered in user profile administration mode. For more information see the User Profile Records section in the "Maintaining Profile Records" chapter of this guide.

Administrators with only the ACCOUNT privilege can insert or change the information in logonid records that are within their scope, but cannot insert or change the information in user profile records. Administrators with only the SECURITY privilege can only change certain fields in the logonid records that are within their scope and can insert or change the information in user profile records that are within their scope, but cannot insert logonid records.

Administrators must have both the ACCOUNT and SECURITY privileges to insert a logonid record and the corresponding user profile records in a single command line, and the logonid and the user profile records being inserted must be within his scope.

INSERT Subcommand

The INSERT subcommand lets a logonid with the ACCOUNT privilege level add a new logonid record to the Logonid database. Similarly, a logonid with the SECURITY privilege level can add a new user profile data record to the Infostorage database. The syntax of the INSERT subcommand is:

```
Insert  {*|logonid [field,...,field]|{Using(modellogonid) newlogonid
         [field,...,field]} [TARGET(null|=|?|nodemask1,...,nodemask100)]
```

Note: Do not specify the characters –, *, &,), (, /, &, or @ in the logonid field. If you specify ABC– as a logonid and you attempt to display ABC–, eTrust CA-ACF2 interprets the dash as a continuation character. Nor should the logonid contain embedded periods due to restrictions at current levels of MVS.

Parameter Descriptions

The parameters of the INSERT subcommand are described in the following:

*** (asterisk)**
Indicates that you want to insert the last logonid referenced. This can be especially useful if you just deleted one logonid and want to recreate a similar one without your site's privileges, zero fields, or statistics.

logonid
Specifies the one to eight character name of the logonid record or user profile records you want to create. You can also specify *field,…field* with the logonid parameter. See the description of the USING parameter for more details.

USING(*modellogonid*) *newlogonid*
Specifies an existing logonid record that you want to use as a model to create a new logonid record. This reduces the number of fields you must enter for a new logonid record. Enter the USING parameter with the one to eight-character logonid for the record that you want to use as the model. Then specify newlogonid, where newlogonid specifies the one to eight character name for the new logonid record that you are adding. You must specify this parameter with the USING parameter.

Note: If you specify the USING parameter, eTrust CA-ACF2 requires that a logonid record field also be specified.

A site can specify fields that cannot be copied from a model logonid record to a new logonid record in the @CFDE macro of the eTrust CA-ACF2 Field Definition Record (ACFFDR). Use the SHOW ZEROFLDS command of the eTrust CA-ACF2 subsystem to display the list of specified fields for your site. See the appendix, "eTrust CA-ACF2 Field Definition Record Generation," in the *Getting Started* guide for more information about this macro.

Note: User profile records cannot be copied from model user profile records. The USING parameter refers only to the new logonid record created and not the user profile records. Specification of the user profile record fields is necessary to create a new user profile record.

field,...,field
Specifies the fields you want in the new logonid record or user profile records and, if necessary, the corresponding value for the field. See the How to Specify Logonid and User Profile Record Fields section earlier in this chapter. Specify logonid fields in any order, but separate them with commas or blank spaces.

If you specify the USING parameter, eTrust CA-ACF2 adds any field that you specify to the new logonid record. If a particular field is already in the model logonid record, eTrust CA-ACF2 changes that field or deletes it.

TARGET(null|=|?|*nodemask1,...,nodemask100*)
Specifies that you want to process logonid or user profile records for a target node or group of nodes. Valid values are:

- **null**—TARGET() processes CPF commands only on the HOME node specified in the CPF OPTIONS record.
- **=** (equal sign)—TARGET(=) processes CPF commands only on the HOME node specified in the CPF OPTIONS record.
- **?** (question mark)—TARGET(?) processes CPF commands on the nodes listed in the DFTCMD parameter of the CPF OPTIONS record.
- ***nodemask1,...,nodemask100***—TARGET(*nodemask1,...,nodemask100*) processes at these specific nodes. These node masks can be individual nodes or they can be masked with an asterisk. Up to 100 nodes are allowed in this list.

Field names can be abbreviated when specified on the INSERT subcommand. Fields might be abbreviated to the minimum number of letters required to differentiate them from other fields in the logonid record and user profile records. If abbreviations match more than one field name or keyword, they are treated as invalid.

Use the INSERT subcommand to define the record name and field specifications. To activate newly created profile records, issue the REBUILD command:

```
F ACF2,REBUILD(USR),CLASS(P)
```

INSERT Subcommand Example 1

The following INSERT subcommand creates a logonid record for the logonid PAY7777:

```
 ACF
set lid
 LID
insert pay7777 name(jane doe) leader scplist(finance) phone(ext.458) -
password(xxxx) tso tsoacct(acct01) mail notices smsinfo(defpay)
```

Note: eTrust CA-ACF2 processes the data you input for the new logonid in the order in which it is specified on the INSERT.

When listed, the new logonid record looks like this:

```
list pay7777
PAY7777       PAY7777 JANE DOE EXT.  458
PRIVILEGES    LEADER SCPLIST(FINANCE) TSO
ACCESS        ACC-CNT(0) ACC-DATE(0) ACC-TIME(0)
PASSWORD      PSWD-TOD(00/00/00-00:00)
TSO           TSOACCT(ACCT01) MAIL NOTICES
STATISTICS    UPD-TOD(00/00/00-00:00)
RESTRICTIONS  PREFIX(PAY7777) DFP SMSINFO(DEFPAY)
```

Note: Passwords are required for all logonids. If you do not specify a password when inserting a logonid, eTrust CA-ACF2 issues a message informing you of this requirement and the insert fails.

INSERT Subcommand Example 2

To create user profile data records, enter logonid administration mode by issuing the SET LID command. The following INSERT subcommand creates an OMVS profile data record for the logonid JANEDOE:

```
 ACF
set lid
 LID
insert janedoe uid(199) home(/u/janedoe) omvspgm(/bin/sh)
```

The information returned will look like the following:

```
OMVS / JANEDOE LAST CHANGED BY MASTER ON 08/23/00-10:19
                    HOME (/u/janedoe) OMVSPGM(/bin/sh) UID(199)
```

INSERT Subcommand Example 3

To create a logonid and user profile data records, enter logonid administration mode by issuing the SET LID command. The following INSERT subcommand creates a logonid record, CICS profile data record, and LNOTES profile data record for the logonid JOHNDOE:

```
 ACF
set lid
 LID
insert johndoe name(john doe) tso opident(chi) sname(johndoe)
```

The information returned will look like the following:

```
JOHNDOE         JOHNDOE JOHNDOE
                COMPANY() DEPT()  IDNUM()  LEVEL()  LOCATION()  OLDLID()
                POSITION() PROJECT()  SITE()
PRIVILEGES      TSO
ACCESS          ACCT-CNT(0)  ACCT-DATE(00/00/00)  ACCT-TIME(00:00)
PASSWORD        PSWD-DAT(00/00/00)  PSWD-INV(0)  PSWD-TOD(00/00/00-00:00)
                PSWD-VIO(0)
TSO             DFT-PFX(JOHNDOE)
STATISTICS      SEC-VIO(0)  UPD-TOD(08/24/00-08:55)
RESTRICTIONS    PREFIX(JOHNDOE)
CICS / JOHNDOE LAST CHANGED BY MASTER ON 08/24/00-08:55
                OPIDENT(CHI)  TIMEOUT(0)
LNOTES / JOHNDOE LAST CHANGED BY MASTER ON 08/24/00-08:55
                SNAME(johndoe)
```

Synchronizing New Logonid Records

When a logonid is inserted, it is normally synchronized with the BRODCAST data set; however, when the IKJTSO*xx* parmlib member indicates that SYS1.BRODCAST is not to be used, the synchronization is bypassed.

SYNCH Subcommand

The SYNCH subcommand synchronizes the Logonid database with the TSO SYS1.BRODCAST data set. Issue this subcommand for TSO users when you create their logonid records.

You can synchronize one logonid record or a group of records. Each time you perform the synchronization, eTrust CA-ACF2 rebuilds the SYS1.BRODCAST data set. Before you issue the SYNCH command, examine your site's synchronization standards to ensure that the appropriate users are included in the BRODCAST data set.

Only **unrestricted** users with the ACCOUNT privilege level can issue this subcommand. This means that the logonid has no SCPLIST field specified in his logonid record. However, a logonid with the ACCOUNT privilege cannot add a user with the SECURITY privilege to the SYS1.BRODCAST data set. To synchronize logonids with the SECURITY privilege, the logonid that executes the SYNCH subcommand must have the SECURITY and ACCOUNT privileges.

You can also use the ACFBSYNC utility to synchronize logonid records in batch. See the *Reports and Utilities Guide* for information about ACFBSYNC.

Note: You must run the TSO SYNC command at least once before running the ACFBSYNC to initialize and build a new BRODCAST data set or to rebuild the data set when it becomes full or is damaged. Use the UADS option when issuing the SYNC command for a eTrust CA-ACF2 system (that is, SYNC UADS). To run the TSO SYNC command, you must have the TSO ACCTPRIV privilege.

The TARGET parameter is not available for SYNCH processing. You cannot issue the SYNCH subcommand or use the ACFBSYNC utility to synchronize the Logonid database with SYS1.BRODCAST on remote nodes.

The SYNCH subcommand has the following syntax:

```
SYNCH  {Like(lidmask) [If(field,...field)]}
       {Uid(uidmask) [If(field1,...field)]}
       {If(field1,...field)}
```

Parameter Descriptions

You must specify one of the following parameters with the SYNCH subcommand:

Like(*lidmask*)
Specifies a mask for the logonid records you want to synchronize. The default is all logonids.

Uid(*uidmask*)
Specifies a mask for the UIDs whose logonid records you want to synchronize.

If(*field1,...field*)
Lets you synchronize all logonid records with the specified fields. For example, the following command synchronizes all logonid records with the TSO privilege:

```
synch if(tso)
```

You can also use the IF parameter to synchronize records for logonids with particular combinations of privileges. For example, execute the following to synchronize the logonid records of those users who have the TSO and NON-CNCL privileges:

```
synch if(tso,non-cncl)
```

eTrust CA-ACF2 places only those logonids that match these attributes on BRODCAST. Use care when you combine attributes using the IF parameter.

With the IF parameter, you can specify any bit field of the logonid record. To specify logonids without a specific privilege, prefix the field with NO.

In addition, this parameter can be used with the LIKE or UID parameters to further refine the selection of logonids.

After you issue the SYNCH subcommand, you must add additional TSO logonids to the BRODCAST data set using the INSERT subcommand if LOGONCK is set in the GSO TSO record. Enter any eight-character logonid on the SYS1.BRODCAST data set under only its first seven characters. The last character is truncated. Otherwise, add all logonids (for CA-ROSCOE, VM, and so forth) to the BRODCAST data set in their entirety with the INSERT subcommand.

CHANGE Subcommand

The CHANGE subcommand lets a logonid with the ACCOUNT or SECURITY privilege add, change, or delete fields of selected logonid records. Logonids with LEADER and CONSULT privileges can also change some of the logonid record fields. The CHANGE subcommand lets a logonid with the SECURITY privilege change user profile data information located in the infostorage database that is within his scope. The syntax of the CHANGE subcommand is:

```
CHAnge    {*}
          {logonid}
          {Like(lidmask) [If(field,...,field)]}
          {Uid(uidmask) [If(field,...,field)]}
          {If(field,...,field)}
          [TARGET(null|=|?|nodemask1,...,nodemask100)]
          [(field,...,field)
          [ADD|REP|DEL]
```

Parameter Descriptions

The parameters of the CHANGE subcommand are described in the following. You must specify one of the following parameters with the CHANGE subcommand:

*** (asterisk)**
Indicates that you want to change the last logonid processed since the LID setting was established.

logonid
Specifies the one to eight character name of the logonid record or user profile records you want to change.

Like(*lidmask*)
Specifies a one to eight-character mask that identifies a group of logonid records you want to change. For example, the mask PAY*** identifies records for all logonids that begin with the letters PAY and end with any zero to three characters, excluding imbedded blanks. PAY–. matches all logonids that begin with PAY regardless of length. If you use this parameter to add or delete the TSO field to a group of logonids, you must also synchronize the SYS1.BRODCAST data set. To recreate the SYS1.BRODCAST data set, use the SYNCH subcommand or the ACFBSYNC utility.

Note: The Like parameter cannot be specified when issuing a CHANGE subcommand with user profile records.

UID(*uidmask*)
Specifies a UID mask that identifies the logonids that you want to change.

Note: The UID parameter cannot be specified when issuing a CHANGE subcommand with user profile records.

IF(*field,...,field*)
Specifies that you want to change a group of logonids if they have a certain field or group of fields. You can change only bit fields with this parameter. Prefix the bit field name with NO to indicate logonids without that option.

For example, the following command changes all the logonid records of users who have the ACCOUNT privilege, but **not** the SECURITY privilege:

```
change if(account,nosecurity)
```

If you specify multiple fields in one IF command, eTrust CA-ACF2 treats them as AND fields. In other words, records are changed only if all the fields you specify match the record.

You can also use this parameter with the LIKE or UID parameters to further refine the selection of logonids.

Note: The IF parameter cannot be specified when issuing a CHANGE subcommand with user profile records.

The following parameters are optional with the CHANGE subcommand:

TARGET(null|=|?|*nodemask1,...,nodemask100*)
Specifies that you want to process logonid records or user profile records for a target node or group of nodes. Valid values are:

- **null**—TARGET() processes CPF commands only on the HOME node specified in the CPF OPTIONS record.
- **=** (equal sign)—TARGET(=) processes CPF commands only on the HOME node specified in the CPF OPTIONS record.

- **?** (question mark)—TARGET(?) processes CPF commands on the nodes listed in the DFTCMD parameter of the CPF OPTIONS record.
- ***nodemask1,...,nodemask100***—TARGET(*nodemask1,...,nodemask100*) processes at these specific nodes. These node masks can be individual nodes or they can be masked with an asterisk. Up to 100 nodes are allowed in this list.

If you want to process an ACF command at a node other than the default node or the current target setting, you must select the TARGET parameter **before** any other parameters. For details on how to use the TARGET parameter and examples, see the "Using the Command Propagation Facility" chapter in this guide.

field,...,field

Specifies the fields and any values that you want to add, replace, or delete (see the following) to the new record. For the specific field names in each record, see chapter where the record is described.

These general rules apply to field names:

- Turn on bit fields by stating the field name. Turn them off by prefixing the field name with NO. For example, to remove the STC option found in the OPTS record, use NOSTC.
- Specify the desired value in parentheses for fields with variable or character values, for example, TIME(12:30). To nullify a field, specify the field with no value, as in TIME().
- Specify the multiple values in parentheses for fields that are defined with the capacity to contain multiple values. For example, PGMS(IMASPZAP AMASPZAP SUPERZAP). Use a space or comma as a valid delimiter in the parentheses.

ADD

Indicates that you want to add the specified fields and values to those in the existing record. This parameter applies only to multi-value fields. ADD is the default.

REP

Indicates that you want to replace the fields and values in an existing record with the fields and values you specify. This parameter applies only to multi-value fields. If any field you specify is not part of the existing record, eTrust CA-ACF2 adds that field and its value to the record.

DEL

Indicates that you want to delete the specified field values from the existing record. This parameter applies only to multi-value fields.

Field names can be abbreviated when specified on the CHANGE subcommand. If abbreviations match more than one field name in the logonid record and in any user profile record, they are treated as invalid.

Use the CHANGE subcommand to redefine the record field specifications. To activate changed profile records, issue the REBUILD command.

```
F ACF2,REBUILD(USR),CLASS(P)
```

CHANGE Subcommand Example 1

Consider the following logonid record for the logonid PAY7777:

```
list pay7777
PAY7777                 PAY7777 JANE DOE EXT.458
PRIVILEGES              LEADER SCPLIST(FINANCE) TSO
ACCESS                  ACC-CNT(0) ACC-DATE(0) ACC-TIME(0)
PASSWORD                PSWD-TOD(01/04/02-12:01)
TSO                     MAIL NOTICES TSOACCT(ACCT01)
STATISTICS              UPD-TOD(01/04/02-12:01)
RESTRICTIONS            PREFIX(PAY7777)
DFP                     SMSINFO(DEFPAY)
```

To add the CICS attribute and remove the TSO and other TSO-related attributes, use the following CHANGE subcommand:

```
change pay7777 cics notso tsoacct() nomail nonotices
```

In this subcommand, the field name CICS lets the user log on to CICS. The letters NO precede the field name to remove the TSO, MAIL, and NOTICES privileges. Because the TSOACCT field has a value associated with it, you remove the field by specifying the field name followed by no value in parentheses. You can specify a value to change the current value of the field.

After this change, the logonid record looks like the following:

```
PAY7777                 PAY7777 JANE DOE EXT.458
PRIVILEGES              CICS LEADER SCPLIST(FINANCE)
ACCESS                  ACC-CNT(0) ACC-DATE(0) ACC-TIME(0)
PASSWORD                PSWD-TOD(01/04/02-12:01)
STATISTICS              UPD-TOD(01/04/02-12:01)
RESTRICTIONS            PREFIX(PAY7777)
DFP                     SMSINFO(DEFPAY)
```

Note: Passwords are required for all logonids. If you do not specify a password when changing a logonid, eTrust CA-ACF2 issues a message informing you of this requirement and the change fails.

CHANGE Subcommand Example 2

Consider the following OMVS profile data record for the logonid TESTGUY:

```
ACF
set lid
 LID
list testguy profile(omvs)
JOHNDOE          TESTGUY            TESTGUY
                COMPANY() DEPT()  IDNUM()  LEVEL()  LOCATION()  OLDLID()
                POSITION() PROJECT()  SITE()
OMVS / TESTGUY   HOME(/u/user) OMVSPGM(/bin/sh) UID(200)
```

The following CHANGE command alters the UID and HOME fields on the OMVS profile data record for the logonid TESTGUY:

```
Change testguy uid(101) home(/u/testguy)
```

The record information returned will look like the following:

```
OMVS / TESTGUY LAST CHANGED BY MASTER ON 08/24/00-13:48
                    HOME (/u/testguy) OMVSPGM(/bin/sh) UID(101)
```

CHANGE Subcommand Example 3

Consider the following LOGONID record, CICS profile data record, and LNOTES profile data record for the logonid TESTER1:

```
TESTER1           TESTER1            TESTER1
                COMPANY() DEPT()  IDNUM()  LEVEL()  LOCATION()  OLDLID()
                POSITION() PROJECT()  SITE()
PRIVILEGES      CICS TSO
ACCESS          ACCT-CNT(0)  ACCT-DATE(00/00/00)  ACCT-TIME(00:00)
PASSWORD        PSWD-DAT(00/00/00)  PSWD-INV(0)  PSWD-TOD(00/00/00-00:00)
                PSWD-VIO(0)
TSO             DFT-PFX(TESTER1) MAIL
STATISTICS      SEC-VIO(0)  UPD-TOD(08/24/00-09:39)
RESTRICTIONS    PREFIX(TESTER1)
CICS / TESTER1 LAST CHANGED BY MASTER ON 08/24/00-09:39
                OPIDENT(CHI)  TIMEOUT(0)
LNOTES / TESTER1 LAST CHANGED BY MASTER ON 08/24/00-09:39
                SNAME(tester1)
```

The following CHANGE command alters the NAME and the MAIL attributes on the logonid record, the OPIDENT field on the CICS profile data record, and the SNAME field on the LNOTES profile data record for TESTER1:

```
Change tester1 name(jane doe) mail opident(nyc) sname(janedoe)
```

The record information returned will look like the following:

```
TESTER1          TESTER1           JANE DOE
                 COMPANY() DEPT()  IDNUM()  LEVEL()  LOCATION()  OLDLID()
                 POSITION() PROJECT()  SITE()
PRIVILEGES       CICS TSO
ACCESS           ACCT-CNT(0)  ACCT-DATE(00/00/00)  ACCT-TIME(00:00)
PASSWORD         PSWD-DAT(00/00/00)  PSWD-INV(0)  PSWD-TOD(00/00/00-00:00)
                 PSWD-VIO(0)
TSO              DFT-PFX(TESTER1)
STATISTICS       SEC-VIO(0)  UPD-TOD(08/24/00-12:03)
RESTRICTIONS     PREFIX(TESTER1)
CICS / TESTER1 LAST CHANGED BY MASTER ON 08/24/00-12:03
                 OPIDENT(NYC)  TIMEOUT(0)
LNOTES / TESTER1 LAST CHANGED BY MASTER ON 08/24/00-12:03
                   SNAME(janedoe)
```

LIST Subcommand

The LIST subcommand lets a logonid with the ACCOUNT, SECURITY, or AUDIT privilege display logonid records. In addition, logonids with the CONSULT or LEADER privilege can list logonids within their scope. Any user can list his logonid record. You can permit users to display other logonid records by specifying privilege fields or restrict these privileges with scope records.

A logonid with the SECURITY privilege level can display user profile data. All or specified user profile records including suffixed profile records can be displayed for a logonid or group of logonids.

Four SET subcommand parameters control which fields eTrust CA-ACF2 displays when you issue a LIST subcommand. These are the TERSE|VERBOSE and TRIVIA|NOTRIVIA parameters:

- If you issue a SET TERSE subcommand, eTrust CA-ACF2 displays only the first line of each logonid record. You specify which fields appear in this display in the @HEADER macro of the ACFFDR. See the appendix, "eTrust CA-ACF2 Field Definition Record Generation," in the *Getting Started* guide for more information.
- If you issue a SET VERBOSE subcommand, eTrust CA-ACF2 displays the full logonid record.
- If you issue a TRIVIA subcommand, eTrust CA-ACF2 displays all fields of the logonid record, if you also specify VERBOSE.
- If you issue a NOTRIVIA subcommand, eTrust CA-ACF2 displays only those fields you specify for the FLAGS=LIMIT operand in the @CFDE macro of the ACFFDR.

The syntax for the LIST subcommand follows:

```
List   {*}
       {logonid}
       {Like(lidmask) [If(field,...,field)]}
       {Uid(uidmask) [If(field,...,field)]}
       {If(field,...field)}
       [SECTION(name,...,name)]
       [PROFILE(name,...,name)]
       [TARGET(null|=|?|nodemask1,...,nodemask100)]
```

Parameter Descriptions

The parameters of the LIST subcommand are described in the following. You must specify one of the following parameters with the LIST subcommand:

*** (asterisk)**
Specifies that you want to display the last logonid record that you referenced in this session. eTrust CA-ACF2 displays your logonid record if you have not specified an individual logonid in any subcommands since you issued the ACF command.

logonid
Specifies the one to eight-character name of the logonid record or user profile record you want to display.

Like(*lidmask*)
Specifies a one to eight-character mask for the group of logonid records or user profile records you want to display. You can specify a dash to display all logonid records in your scope. You can also specify the IF(*field,...,field*) parameter with the LIKE parameter.

UID(*uidmask*)
Specifies that you want to display a group of logonid records or user profile records that match the mask. You can also specify the IF(*field,...,field*) parameter with the UID parameter.

If(*field,...,field*)
Lets you define flexible record selection criteria. This parameter is formatted similarly to a high-level programming language IF statement. The variables available for processing are the various logonid record field names or constants (defined in the following). The available operators are also defined in the following. The full IF expression is evaluated as an algebraic expression yielding a TRUE (that is, select record) or FALSE (that is, bypass record) value. If the result of the IF expression is a quantity, NONZERO is considered TRUE (selected) and a zero value is considered FALSE (not selected). Parentheses can group expressions to override the normal precedence order.

When specifying a constant (specific value) in the IF statement, use the following formats:

Format	Contents	Type
'aaaa' or C'aaaa'	Alphanumerics	Character fields
Nnnn	Numerics	Binary number fields
X'xx'	Hex numbers	Hex fields
B'n'	1 or 0 (1=on, 0=off)	Bit (flag) fields
P'nn'	Numerics	Packed decimal fields
D'mm/dd/yy' or D'dd/mm/yy' or D'yy/mm/dd'	Numerics with dividing Date field slashes (the format used is based on local system option). See the following note. Use single quotes before and after the date.	
U'xxxx'	Any of the previous contents can be used. eTrust CA-ACF2 resolves the data type whatever the field is defined as in the @CFDE.	

Note: You can designate the DATE field as a TOD clock field or you can store the DATE field as a packed decimal in the logonid record; however, the output is a date. Time-of-day fields are treated as date fields only. No comparison is made of the time portion of the field by the IF processor.

The IF expression operators (in order of precedence) are as follows:

Precedence	Character	Symbol	Meaning
1.	NOT	^	Not
2.	OR	|	Or
3.	AND	& or ,	And
4.a.	EQ	=	Equal
4.b.	NE	^=	Not equal
4.c.	LE	<=	Less than or equal to
4.d.	GE	>=	Greater than or equal to

Precedence	Character	Symbol	Meaning
4.e.	LT	<	Less than
4.f.	GT	>	Greater than
5.		-	Negative value
6.a.		*	Times
6.b.		/	Divided by
7.a.		+	Plus
7.b.		-	Minus
8.		||	Concatenated to (that is, used between field names, in sequence, to show concatenation of fields). Since UID is not an actual field in the logonid record and cannot be referenced directly, this can be used to build a UID.

The symbols or the character abbreviations are acceptable to the program.

The command accepts only one IF statement. However, you can specify multiple criteria in one statement. If the criterion needs to continue onto more than one line, use a dash (-) as the last non-blank character on the line. eTrust CA-ACF2 recognizes a continuation character unconditionally. For example:

```
l if(security and noims and account and nocics and -tsoproc='tsoproc' and –
(acc-date le u'01/01/02') and (acc-cnt ge 123) and (group = 'acfgroup') and –
maxdays = 0 and (tsorba le u'1234'))
```

For details, see How to Specify Logonid and User Profile Record Fields earlier in this chapter.

You can also specify the IF parameter with the LIKE or UID parameter. To display logonids without a specific bit field, prefix the field name with NO.

The following parameters are optional with the LIST subcommand:

SECTION(*name,...,name*)

Specifies the name of one or more sections to be displayed from the logonid or logonids selected. Name can be one of the following:

- **ALL**—displays the header and Logonid record information for the selected logonid.
- **HEADER**—displays only the header information for the selected logonid. Enter this keyword only if just the header section is desired. For all other requests, header information automatically displays.

- ***logonid group name*** — displays the identifying header information for the selected logonid and information contained in the specified sections of the Logonid record.

If you do not specify one of these options for SECTION, eTrust CA-ACF2 displays header information only.

PROFILE(*name,...,name*)
Specifies the name of one or more types of user profile information to be displayed for the logonid or logonids selected. name can be one of the following:

- **ALL** — displays information contained in any user profile record for the selected logonid.
- ***user profile type*** — displays only the user profile information of the specified types for the selected logonid.

TARGET(null|=|?|*nodemask1,...,nodemask100*)
Specifies that you want to process logonid records for a target node or group of nodes.

- **null** — TARGET() processes CPF commands only on the HOME node specified in the CPF OPTIONS record.
- **=** (equal sign) — TARGET(=) processes CPF commands only on the HOME node specified in the CPF OPTIONS record.
- **?** (question mark) — TARGET(?) processes CPF commands on the nodes listed in the DFTCMD parameter of the CPF OPTIONS record.
- ***nodemask1,...,nodemask100*** — TARGET(*nodemask1,...,nodemask100*) processes at these specific nodes. These node masks can be individual nodes or they can be masked with an asterisk. Up to 100 nodes are allowed in this list.

Logonid field names can be abbreviated when specified on the LIST subcommand. If abbreviations match more than one field name, they are treated as invalid.

List Subcommand Example 1

To display user profile data records, enter logonid administration mode by issuing the SET LID command. The following LIST command displays all user profile data records for the group of logonids TESTER-:

```
ACF
set lid
 LID
list like(tester-) profile(all)
```

The record information returned will look like the following:

```
TESTER1               TESTER1        JANE DOE
                      COMPANY() DEPT()  IDNUM()  LEVEL()  LOCATION()  OLDLID()
                      POSITION() PROJECT()  SITE()
CICS / TESTER1        OPIDENT(NYC)  TIMEOUT(0)
LNOTES / TESTER1      SNAME(janedoe)

TESTER2                 TESTER2       JOHN DOE
                      COMPANY() DEPT()  IDNUM()  LEVEL()  LOCATION()  OLDLID()
                      POSITION() PROJECT()  SITE()
CICS / TESTER2        OPIDENT(LA)  TIMEOUT(0)
KEYRING / TESTER2     DEFAULT()  RINGNAME(JOHN DOE)
No certificates are connected to this key ring
KEYRING / TESTER2.A   DEFAULT()  RINGNAME()
No certificates are connected to this key ring

KEYRING / TESTER2.B   DEFAULT()  RINGNAME()
No certificates are connected to this key ring

LANGUAGE / TESTER2    PRIMARY(ENU)  SECONDARY(GER)
LNOTES / TESTER2      SNAME(johndoe)
NDS / TESTER2         UNAME(johndoe)
NETVIEW / TESTER2     NTVCLASS(1 2 3)
OPERPARM / TESTER2    KEY(JOHNDOE)
```

List Subcommand Example 2

To display specific user profile data records, enter logonid administration mode by issuing the SET LID command. The following LIST command displays corresponding KEYRING, LANGUAGE, and OPERPARM profile data records for the userid TESTER2:

```
ACF
set lid
 LID
list tester2 profile(keyring, language, operparm)
```

The record information returned will look like the following:

```
TESTER2               TESTER2        JOHN DOE
                      COMPANY() DEPT()  IDNUM()  LEVEL()  LOCATION()  OLDLID()
                      POSITION() PROJECT()  SITE()
KEYRING / TESTER2     DEFAULT()  RINGNAME(JOHN DOE)
No certificates are connected to this key ring
KEYRING / TESTER2.A   DEFAULT()  RINGNAME()
No certificates are connected to this key ring

KEYRING / TESTER2.B   DEFAULT()  RINGNAME()
No certificates are connected to this key ring

LANGUAGE / TESTER2    PRIMARY(ENU)  SECONDARY(GER)
OPERPARM / TESTER2    KEY(JOHNDOE)
```

DELETE Subcommand

The DELETE subcommand lets users with the ACCOUNT privilege delete logonid records. By default, this subcommand also deletes any access rule set whose key matches that of any deleted logonid record. Additionally, this subcommand also deletes any entry for the logonid on the SYS1.BRODCAST data set. When a logonid is deleted, associated user profile records are deleted from the infostorage database. Only users with the ACCOUNT privilege level can delete logonid records. You can restrict this authority through the use of scopes.

The syntax of the DELETE subcommand is:

```
DELete    {*}
          {logonid}
          {LIke(lidmask) [If(field,...,field)]}
          {Uid(uidmask) [If(field,...,field)]}
          {If(field,...,fieldn)}
          [NORule]
          [TARGET(null|=|?|nodemask1,...,nodemask100)]
```

Parameter Descriptions

The parameters for the DELETE subcommand are described in the following. You must select one of the following parameters with the DELETE subcommand.

*** (asterisk)**
Specifies that you want to delete the last logonid record that you referenced in your current TSO session. You cannot delete your own logonid.

logonid
Specifies the one to eight-character name of the logonid record you want to delete.

LIke(*lidmask*)
Specifies that you want to delete a group of logonid records that match the mask. Masks can be one to eight characters long. When you issue an online command that deletes multiple records from the eTrust CA-ACF2 databases, eTrust CA-ACF2 asks you to confirm the DELETE LIKE request. If you respond **Y**, eTrust CA-ACF2 performs the delete. If you specify **N** or any other response, eTrust CA-ACF2 ignores the command and prompts for the next subcommand.

Uid(*uidmask*)
Specifies that you want to delete a group of logonid records whose UIDs match the mask.

If(*field,...,field*)
Specifies that you want to delete a group of logonid records with the specified fields. You can specify up to 16 fields. For example, the following command deletes all logonids with the ACCOUNT but not the SECURITY privilege:

```
delete if(account,nosecurity)
```

To delete logonids without a specific bit privilege, prefix the field name with NO. You can specify this parameter for bit fields only.

You can also use this parameter with the LIKE or UID parameter to further refine the selection of logonids.

The following parameters are optional with the DELETE subcommand:

NORule
Specifies that you want to delete the logonid record but **not** the access rules whose high-level qualifier matches the logonid. By default, eTrust CA-ACF2 deletes these rules whenever you delete a logonid. If you specify target nodes, the RULE parameter deletes the access rules for the corresponding logonid in the Rule database on each node specified. For example, the following deletes the logonid record and corresponding access rule set for the logonid PAY777:

```
delete PAY7777
```

After you issue the DELETE subcommand, eTrust CA-ACF2 returns the message DELETED.

TARGET(null|=|?|*nodemask1,...,nodemask100*)
Specifies that you want to process logonid records for a target node or group of nodes.

- **null**—TARGET() processes CPF commands only on the HOME node specified in the CPF OPTIONS record.
- **=** (equal sign)—TARGET(=) processes CPF commands only on the HOME node specified in the CPF OPTIONS record.
- **?** (question mark)—TARGET(?) processes CPF commands on the nodes listed in the DFTCMD parameter of the CPF OPTIONS record.
- ***nodemask1,...,nodemask100***—TARGET(*nodemask1,...,nodemask100*) processes at these specific nodes. These node masks can be individual nodes or they can be masked with an asterisk. Up to 100 nodes are allowed in this list.

If you enter a command to delete more than one logonid record, eTrust CA-ACF2 responds with the following prompt:

```
ARE YOU SURE?
```

You must respond Y or YES to process the delete request.

Chapter

4

Maintaining Scope Records

You create *scope records* to limit a user's administrative authority over the eTrust CA-ACF2 Logonid, Rule, and Infostorage databases. Although they are used to control the actions of logonids, eTrust CA-ACF2 stores scope records in the Infostorage database. The SCPLIST field of the logonid record points to the name of the scope record. We sometimes see scope records as *scope lists*.

This chapter describes the following topics:

- Specifying a scope for a logonid
- Parameters and fields that affect scopes
- Using the ISPF panels to create scope records
- Using the ACF command to create scope records
- Activating changes to scope records

Specifying a Scope for a Logonid

A scope record specifies a list of data set high-level indexes, logonids, UIDs, or infostorage keys to be the authorized limits of a privileged logonid. When you assign a scope record for a logonid, you limit its access to the eTrust CA-ACF2 databases. Scope records grant no special privileges to a user. They provide you with a means to delegate security administration to other logonids and limit the power of those logonids.

To maintain scope records and assign a scope record to a logonid, you must understand the following concepts:

- Naming scope records
- Specifying scope record fields
- Scope field summary chart
- Preventing access to a database in a scope record

Naming Scope Records

Unlike logonid records, which are stored in the Logonid database, scope records are stored in the Infostorage database. eTrust CA-ACF2 stores scope records using the three-character type code SCP and a one to eight character record name. To assign a scope to a logonid, specify a value for the SCPLIST field of the logonid record for that user. The value you specify is the one to eight-character name of the scope record.

For example, suppose you wanted to assign a scope to a logonid with the SECURITY privilege so that the user could administer payroll-related data only. You must change the user's logonid record by specifying a value, such as PAYSCOPE, for the SCPLIST field. (The procedure is described later in this section.) This value PAYSCOPE instructs eTrust CA-ACF2 to check the Infostorage database for a type SCP record named PAYSCOPE.

eTrust CA-ACF2 displays the following when you issue the LIST subcommand in the SCP setting:

```
 ACF
set scope(scp)
 SCOPE
list payscope
 ACF60062 SCOPE PAYSCOPE STORED BY PAYSDH ON 09/15/01-11:43
 DSN(PAYWORK,PAYTEST) LID(PAY-) UID(FIN-) INF(RCKCPAYT,RITRPAYUPD)
 NEXTKEY(PAYSCOP2)
```

This record indicates that a security administrator with a scope of PAYSCOPE can do the following tasks:

- Write access rules that govern the two groups of payroll work files with the high-level indexes PAYWORK and PAYTEST.
- Write resource rules for CICS and IMS payroll transactions (PAYT on CICS and PAYUPD on IMS).
- Update logonid records that begin with PAY and whose UIDs begin with FIN.
- If access is denied from this scope record, redrive the scope checking with PAYSCOP2 scope record.

Specifying Scope Record Fields

The four scope record parameters DSN, INF, LID, and UID define a privileged logonid's access to the eTrust CA-ACF2 databases. If you do not specify a value for one of these parameters, eTrust CA-ACF2 denies the scoped logonid access to the corresponding database. You can specify a single value or a list of values using the eTrust CA-ACF2 masking characters asterisk (*) and dash (-).

Note: If the dash (-) mask character is used anywhere except at the end of an INF parameter, it is taken as a literal character. In the following example, the dash (-) in the INF parameter is taken literally and not as a mask. However, in the DSN, UID, and LID parameters, the dash is taken as a masking character:

```
DSN(PAY-,ACCT-) UID(PAY-) LID(PAY-) INF(RSAFTEST.-.RESOURCE)
```

To use masking in the middle of an INF parameter, you must use the asterisk (*). For example, INF(RSAFTEST.****.RESOURCE) can be used to mask a four-character second level resource.

DSN

Specifies the high-level index or $KEY of the data sets you want to include in the scope of a logonid. You can specify multiple data set high-level index names or masks.

By specifying a high-level index here, you limit the rule writing and data set access privileges of logonids with this scope.

DSN also relates to the LID and UID fields. A scoped security administrator or account manager can change the DFT-PFX field of a logonid record only if the value specified for DSN is in the range specified for the LID and UID fields.

INF

Specifies multiple one to 44-character entries that indicate the limits placed on the Infostorage database records that the scoped logonid can insert, alter, or delete and the resources that can be accessed through the SECURITY privilege. The format of the entries for the INF parameter are described in the following:

```
INF(ctttk)
```

c

Storage class (for example, R for resource rules)

ttt

Type code (for example, CKC for CICS transaction rules)

k

Name, consisting of 1 to 256 characters (for example, PAYT for the name of a specific CICS transaction).

For example, if the resource rule storage key consists of the following:

R

Storage class for a resource rule set

CKC

Type code for CICS transactions

ACFM

Name of the transaction

Then, the entry in the INF parameter is INF(RCKCACFM-).

Unlike other infostorage records, cross-reference records are sysid-dependent. This means that in specifying the scope for cross-reference records, you must include the system identifier:

```
INF(ctttsssssssk)
```

C
Storage class (X for cross-reference records)

ttt
Type code (for example, RGP for resource group)

ssssssss
Sysid name on which cross-reference records reside

k
Record ID name (for example, a resource group name specified in your XRGP record).

For example, if the Cross-Reference record storage key consists of the following:

X
Storage class for Cross-Reference records

RGP
Type code for Cross-Reference Resource Group records

CPU1bbbb
Eight-character system identifier for the record name, where bbbb are blank spaces

CRSREF1
Record ID name

Then, the entry in the INF parameter would be INF(XRGPCPU1bbbbCRSREF1-).

LID
Specifies logonids or logonid masks that the scoped logonid can create, list, alter, or delete. You must also specify a UID entry to permit access to records in the Logonid database.

UID
Specifies the one to 24-character UID or UID mask to be placed in the scope of the logonid. You must also specify a LID entry to permit access to the Logonid database.

The effect of a scope relates directly to the authority privileges of the logonid. Multiple entries or fields can exist on a scope record, such as:

```
DSN(PAY-,ACCT-) UID(PAY-) LID(PAY-) INF(CGSO-,FEXP-)
```

NEXTKEY
Specifies the key of an alternate scope record that eTrust CA-ACF2 should check if access is denied based on the current scope record. A maximum of ten nextkey records are processed. Masking is not allowed in this field.

Note: When entering resource names in the INF(…) portion of a scope record, be aware of mixed case resource names that must be entered in mixed case. Use the SHOW CLASMAP subcommand of the ACF command to find out which resource types are mixed case. See the CLASMAP Record section in the "Understanding SAF" chapter for further information on mixed case resource names.

Scope Field Summary Chart

The following chart describes each field of the scope record and its effect on the specific privilege type of a logonid. There is a distinction made between the ability to update or create records. This authority is granted by the special privilege, not the scope. See the "Maintaining Logonid Records" chapter for descriptions of the privilege fields of the logonid record.

Authority Level	LID	UID	DSN	INF
SECURITY[1]	Update and list matching logonids	Update and list matching logonids	Compile, store, test, or delete matching rule sets	Create, update, and delete matching infostorage records
ACCOUNT[2]	Create, update, list, and delete matching logonids	Create, update, and list matching logonids	Create, update, or delete logonids with a matching PREFIX field	None
AUDIT	Display matching logonids	Display matching logonids	Decompile matching rule sets	Display matching infostorage records
LEADER	Update and display selected fields in matching logonids	Update and display selected fields in matching logonids	None	None

[1] After you scope a SECURITY logonid, the scoped security administrator cannot change certain fields of the logonid record. These fields are indicated by the RESTRICT flag in the @CFDE macro. For example, a scoped SECURITY logonid can no longer update the SCPLIST fields of logonid records. However, he can create scope records if they are in his INF scope. (See the description of the CA-ACF2 Field Definition Record (ACFFDR) in the *Getting Started* guide.)

[2] A scoped ACCOUNT logonid cannot use the ACF SYNCH subcommand.

Authority Level	LID	UID	DSN	INF
CONSULT	Update and display selected fields in matching logonids	Update and display selected fields in matching logonids	None	None
USER	None	None	None	None

The LID and UID fields interact together. If you scope a user with the ACCOUNT privilege, both the LID and UID fields require an entry. This way the user with the ACCOUNT privilege can create other logonids with the UID named in the scope. These two fields are required to scope any privileged user to permit access to other logonid records.

If the logonid with the ACCOUNT privilege is going to assign different PREFIX values to logonids, the values must be in the list or masks you specify in the DSN field.

Preventing Access in a Scope Record

To prevent a privileged user from accessing a particular eTrust CA-ACF2 database, do not specify a value for the scope record field that applies to that database. For example, a logonid with the SECURITY privilege might have a scope record with INF entries for entry records:

```
SCOPE LIMIT1 INF(ESRC-,XSGP-)
```

Since no entries were made in the LID, UID, or DSN fields, the logonid with the SECURITY privilege does not have the authority to update logonids or access rule sets. (Default equals no access.) The authority of his SECURITY privilege, as it relates to the eTrust CA-ACF2 databases, is to create and update the source entry list records and cross-reference source group records in the Infostorage database.

Parameters and Fields that Affect Scopes

This section describes how the following parameters and fields affect scope records:

- %CHANGE and %RCHANGE parameters in rules
- INFOLIST field of the GSO OPTS record
- DECOMP field of the GSO RULEOPTS record

%CHANGE and %RCHANGE Parameters in Rules

You can specify parameters in rule sets that override the values in the DSN and INF fields. These fields are %CHANGE and %RCHANGE. When you specify these parameters, you delegate change authority for the rule, regardless of any scope restrictions.

INFOLIST Field of GSO OPTS Record

The INFOLIST field of the GSO OPTS record lets users with the specified privilege read records in the eTrust CA-ACF2 Infostorage database, such as CONTROL, CPF, ENTRY, FIELD, IDENTITY, SCOPE, SHIFT, XREF, and ZONE records. INFOLIST privileges override any scope record limits. The INFOLIST field does not apply to resource rules.

DECOMP Field of GSO RULEOPTS Record

The DECOMP field on the GSO RULEOPTS record lets users with the specified privilege decompile (view) access and resource rules. DECOMP privileges override any scope record limits.

Using the ISPF Panels

To maintain scope records, select option 2 LOGONIDS from the eTrust CA-ACF2 ISPF Option Selection Menu, as follows:

```
------------------- eTrust CA-ACF2 ISPF OPTION SELECTION MENU -----------------
OPTION  ===>

   1  RULES     - PROCESS eTrust CA-ACF2 ACCESS AND GENERALIZED RESOURCE RULES
   2  LOGONIDS  - eTrust CA-ACF2 LOGONID CREATION/MAINTENANCE FACILITY
   3  SYSTEM    - eTrust CA-ACF2 SHOW COMMANDS
   4  REPORTS   - eTrust CA-ACF2 REPORT PROGRAM PROCESSOR
   5  UTILITIES - PROCESS eTrust CA-ACF2 UTILITIES
   6  GSO       - GLOBAL SYSTEM OPTIONS SERVICES
   7  NET       - NETWORKING SYSTEM OPTIONS SERVICES
   8  CAC       - MVS DATABASE CACHE RECORD SERVICES
   9  XREF      - SOURCE/RESOURCE AND GROUP RECORD SERVICES
  10  MAC       - MANDATORY ACCESS CONTROL ADMINISTRATION
  11  CPF       - COMMAND PROPAGATION FACILITY SERVICES
  12  FIELD     - RECORD LEVEL PROTECTION CONTROLS
  13  TARGETS   - SET CPF TARGET NODES, DEFAULTS IN USE
  14  PROFILE   - PROCESS PROFILE INFORMATION RECORDS
  15  SMS       - PROCESS DFSMS SUPPORT RECORDS
  16  ENTRY     - PROCESS ENTRY SOURCE RECORDS
  17  SHIFT     - PROCESS SHIFT/ZONE RECORDS
  18  RACDCERT  - PROCESS KEYRING/CERTIFICATE COMMANDS
  19  C-CIC     - Process C-CIC CICS Initialization Records
  20  LDS       - Process LDAP Directory Services
```

The eTrust CA-ACF2 Logonid Maintenance panel is displayed. If you want to process an ACF command at a node other than the default node or the current target setting, you must select the TARGETS option and specify the target nodes **before** you select the ADD, CHANGE, LIST, DELETE, SUSPEND, or RESET options. To create scope records, select option 8 SCOPE, as follows:

```
---------------------- eTrust CA-ACF2 LOGONID MAINTENANCE --------------------
OPTION  ===>


   1  ADD     - INSERT A NEW LOGONID (TSO, BATCH, ETC.)
   2  ADD     - INSERT A NEW CICS/IMS LOGONID
   3  CHANGE  - CHANGE A LOGONID
   4  DELETE  - DELETE AN EXISTING LOGONID
   5  LIST    - LIST A LOGONID
   6  SUSPEND - CANCEL/SUSPEND A LOGONID
   7  RESET   - RESET A LOGONID PASSWORD/VIOLATIONS
   8  SCOPE   - CREATE/MAINTAIN SCOPE RECORDS FOR LOGONIDS
   9  TARGETS - SET CPF TARGET NODES, DEFAULTS IN USE
```

The Scope Record Maintenance panel is displayed. Use the tab keys and type the appropriate values in the fields.

```
----------------------- SCOPE RECORD MAINTENANCE -----------------------------
COMMAND ===>

    OPERATION   ===>           I=INSERT, C=CHANGE, L=LIST, D=DELETE

    RECORD ID   ===>           RECORD NAME OR MASK

    USING RECID ===>           TEMPLATE RECORD ID

    OPTION      ===>           (ADD/REP/DEL FOR INSERT OR CHANGE)
                               (ALL/DSN/LID/UID/INF FOR LIST)

      THE FOLLOWING PARAMETERS CAN BE MASKED AND/OR LISTED:

    LOGONID     ===>

    UID STRING  ===>

    INDEX       ===>

    STORAGE KEY ===>

    NEXTKEY     ===>

ENTER ENTER TO CONTINUE OR END TO CANCEL REQUEST
```

To exit this panel, type END on the command line.

After you complete the entries for this panel, press Enter. eTrust CA-ACF2 stores the scope record.

Panel Field Descriptions

The fields for the Scope Record Maintenance panel are described in the following.

OPERATION
Specify one of the following values:

- **I**—insert a new scope record
- **C**—change a scope record
- **D**—delete a scope record
- **L**—list a scope record.

RECORD ID
Specify the name of the scope record you want to insert, change, or delete. You can also specify a mask. If you specify a mask, you can perform the action on all scope records that match the mask.

USING RECID
Specify the name of another scope record that you want to use as a model.

OPTION
Specify one of the following values for INSERT or CHANGE:

- **ADD**—adds the fields and their values to the scope record.
- **REP**—replaces any specified fields in the current version of the record or the model record with the specified values. If the fields do not currently exist, eTrust CA-ACF2 adds them to the scope record.
- **DEL**—deletes the specified fields and values from the current record or the model record.

Specify one of the following for LIST:

- **ALL**—display all four sections of a scope record.
- **DSN**—display only the data set section of a scope record.
- **LID**—display only the logonid section of a scope record.
- **UID**—display only the UID section of a scope record.
- **INF**—display only the infostorage section of a scope record.

ADD is the default. This field does not apply to DELETE requests.

LOGONID NAME
Specify logonids or logonid masks that are in this logonid's scope.

UID STRING
Specify UIDs or UID masks that are in this logonid's scope.

INDEX
Specify data set names or masks that are in this logonid's scope.

STORAGE KEY
Specify infostorage record names or masks that are in this logonid's scope.

NEXTKEY
Specify one scope list record for additional scope checking.

Using the ACF Command

To maintain scope records with ACF subcommands, enter the following to establish the SCOPE setting:

```
set scope(scp)
```

After you establish the SCOPE setting, you can issue any of the following ACF subcommands:

- ACCESS
- CHANGE
- CHKCERT
- CONNECT
- DELETE
- END or QUIT
- EXPORT
- GENCERT
- GENREQ
- HELP
- INSERT
- LIST
- REKEY
- REMOVE
- ROLLOVER
- SET or T
- SHOW
- SN
- SYNCH

The common subcommands ACCESS, CHKCERT, CONNECT, EXPORT, END, QUIT, GENCERT, GENREQ, REKEY, REMOVE, ROLLOVER, HELP, SET, SHOW, and SN are not described in this section. They operate under all settings. The CHANGE, DELETE, INSERT, and LIST subcommands are described in this section.

The name of a scope record is from one to eight characters long. To activate the record, specify its name in the SCPLIST field of the logonid record. The record name that you specify in the SCPLIST field of the user's logonid record points to the associated record in the Infostorage database. After you establish the SCOPE setting, you can insert a scope record by using a subcommand, such as:

```
set scope(scp)
 SCOPE
insert limit1 dsn(payroll) lid(pay-) uid(1h*pr-)
 ACF60062 SCOPE SCOPE2 STORED BY PAYSDH ON 07/15/04-11:43
 DSN(PAYROLL) LID(PAY-) UID(1H*PR-)
```

This example illustrates the entries for a scope named LIMIT1. When applied to a logonid with the SECURITY privilege, that logonid has authority to access the rule set PAYROLL, and update logonids that begin with PAY and that match the UID entry 1H*PR. This scoped logonid cannot access any other logonids, any other access rule sets, or any other infostorage records.

INSERT Subcommand

The INSERT subcommand lets you create new scope records. You must have the SECURITY privilege to create scope records.

The INSERT subcommand has the following syntax:

```
Insert      {*|recid|Using(modelscope) newscope}
            [ADD|DEL|REP]

            {[Dsn(entry,...,entry)]}
            {[Inf(entry,...,entry)]}
            {[Lid(entry,...,entry)]}
            {[Uid(entry,...,entry)]}
            [Nextkey(entry)]
            [TARGET(null|=|?|nodemask,...,nodemask)]
```

In its simplest form, this subcommand requires a scope record name of one to eight characters and one field, such as INF.

```
i limit3 i(esrc-,esgp-)
```

The LIMIT3 scope record contains INF entries. When associated with a SECURITY logonid, this scope record restricts the user's access to records in the Infostorage database to only entry records. By default, complete restriction is placed on access to logonids and access rules. If the scope contains DSN, LID, or UID fields, both the LID and UID entries are required to allow the appropriate access.

If you specify multiple values in the DSN, INF, LID, and UID fields, separate them with commas.

Parameter Descriptions:

The INSERT subcommand takes the following parameters:

*** (asterisk)**
Specifies that this subcommand applies to the last scope record processed since you established the SCP setting.

recid
Specifies a one to eight-character scope record name.

USing(*modelscope*)
Specifies the name of an existing model scope record. The fields are copied from that model record to create the new scope record. The ADD, REP, and DEL parameters alter the fields that are in the new record. If you omit these parameters when you specify the USING parameter, eTrust CA-ACF2 assumes the ADD parameter. Any additional values you specify are added to the new record along with those specified in the model record. For example:

```
 ACF
set scope(scp)
 SCOPE
insert using(limitl) limit2 add lid(act-)
 ACF60062 SCOPE SCOPE2 STORED BY PAYSDH ON 07/15/04-11:43
 DSN(payroll) LID(PAY-,act-) UID(1H*PR-)
```

This example subcommand uses the scope record LIMIT1 to create a new record named LIMIT2. LIMIT2 contains the same fields as LIMIT1. LIMIT2 contains PAY-1 and an additional LID mask, ACT-.

newscope
Specifies the name of the new scope record you want to insert. The name must be from one to eight characters in length. This parameter is required if you specify USING.

ADD
Adds any values you specify for the DSN, INF, LID, and UID fields to the fields of the model record. Use the ADD parameter when you want to insert a new scope record using a model record. eTrust CA-ACF2 defaults to ADD when you use the USING subcommand and do not specify ADD, REP, or DEL. For example, here are two INSERT subcommands:

```
insert scope1 dsn(payroll) lid(act-) uid(1h*pr-)
insert using(scope1) scope2 add dsn(account)

list scope2
 ACF60062 SCOPE SCOPE2 STORED BY PAYSDH ON 07/15/04-11:43
 DSN(PAYROLL,ACCOUNT) LID(ACT-) UID(1H*PR-)
```

The first subcommand inserts a scope record named SCOPE1. The second subcommand uses SCOPE1 to insert a scope record named SCOPE2. SCOPE2 contains DSNs of PAYROLL (as does SCOPE1) and ACCOUNT. The LID field value of ACT- and the UID field value of 1H*PR- are also copied from the SCOPE1 record to SCOPE2.

DEL

Indicates that any values you specify in the DSN, INF, LID, and UID fields of the INSERT subcommand are deleted from the corresponding fields of the model record. For example, here are two INSERT subcommands:

```
insert scope1 dsn(payroll) lid(pay-)
insert using(scope1) scope2 del lid(pay-)

list scope2
ACF60062 SCOPE SCOPE2 STORED BY PAYSDH ON 07/15/04-11:43
DSN(PAYROLL)
```

The first subcommand inserts a scope record named SCOPE1. The second INSERT subcommand uses SCOPE1 to insert a scope record named SCOPE2. SCOPE2 contains a DSN field with the value PAYROLL (as does SCOPE1), but the LID field that was copied has been deleted. You should only use the DEL parameter when you want to insert a new scope record with the USING parameter but must omit a field that is in the model record.

REP

Indicates that any values you specify in the DSN, INF, LID, and UID fields replace the corresponding fields of the model record. The REP parameter is meaningful only when you specify the USING parameter. For example, two INSERT subcommands:

```
insert scope1 dsn(payroll) lid(pay-) uid(1h*pr-)
insert using(scope1) scope2 rep lid(act-)

list scope2
 ACF60062 SCOPE SCOPE2 STORED BY PAYSDH ON 07/15/04-11:43
 DSN(PAYROLL) LID(ACT-) UID(1H*PR-)
```

The first subcommand inserts a scope record named SCOPE1. The second INSERT subcommand uses SCOPE1 to insert a scope record named SCOPE2. SCOPE2 contains a DSN field with the value PAYROLL and a UID field value of 1H*PR- (as does SCOPE1). SCOPE2 also contains a LID field with the value ACT- that replaces the value PAY- from SCOPE1.

Dsn(*entry,...,entry*)—Restricts the logonid's authority to access rule sets. Specify the rule sets with the high-level indexes or a mask of the high-level indexes. A mask can be from one to eight alphanumeric characters. You must separate multiple masks or high-level indexes with commas.

If you do not specify this field, access to the data set is completely restricted except for access granted through access rule sets or special privileges. Scoped users can decompile access and resource rules regardless of the limits imposed on their scopes if they are listed on the DECOMP field of the GSO RULEOPTS record. See Parameters and Fields That Affect Scopes earlier in this chapter for details about what can override the values for DSN.

Inf(*entry,...,entry*)

Restricts Infostorage database access so that it includes access to only the specified infostorage records included within the mask such as resource rule sets, entry records, scope records, shift records, zone records, control records, and so forth.

Identify the records with a one to 44-character storage key. This storage key consists of the following:

- A letter code indicating the storage class, as follows:
  - **C**—Control records
  - **D**—DB2 resource rule sets
  - **E**—Entry records
  - **F**—Field records
  - **I**—Identity records
  - **P**—Profile records
  - **R**—Resource rule sets
  - **S**—Scope records
  - **T**—Shift and zone records.
  - **X**—Cross-reference records
- The three-character type code that identifies the type of record or resource rule set. (This type code corresponds with the one used to establish the CONTROL, DB2, ENTRY, FIELD, IDENTITY, PROFILE, RESOURCE, SCOPE, SHIFT, or XREF setting.)
- A system or division identifier that corresponds to the APPLDIV field of the GSO APPLDEF record. The length of the division is specified in the APPLDLEN field. The size of the identifier is normally 8 bytes, but can be any value from 1 to 8 bytes in length. The division identifier defines the system ID (SYSID), network node, site routine, and so forth to which the record applies. Record classes E, F, R, S, and T do not have division names.
- The key of the resource rule set or the name of the Infostorage database record.

For example, if the resource rule storage key consists of the following:

- **R**—the storage class for a resource rule set
- **CKC**—the type code for CICS transactions
- **ACFM**—the name of the transaction

Then, the entry in the INF parameter is INF(RCKCACFM-). Or, if the control record storage key consists of the following:

- **C**—the storage class for control records
- **GSO**—the type code for global system options
- **CPU1bbbb**—the eight character division or system identifier for the recordname of the transaction, where bbbb are blank spaces
- **OPTS**—the name of the infostorage record

Then, the entry in the INF parameter is INF(CGSOCPU1bbbbOPTS-).

Separate multiple storage key masks with commas. If you do not specify this field and you specify the DSN, LID, or UID parameter, access to resources and to the Infostorage database is completely restricted.

However, access can be permitted by resource rules, special privileges, or pseudo data set names. Scoped users with similar authority, as specified in the INFOLIST field of the GSO OPTS record, are still able to list infostorage records regardless of the limits imposed on their scopes. See Parameters and Fields That Affect Scopes earlier in this chapter for details about what can override the values for INF.

Lid(*entry,...,entry*)
Restricts logonid record access so that it includes access to only the specified logonid records. These specified records are identified by logonids or logonid masks of one to eight characters. Separate logonids or masks using commas. If you do not specify the LID or UID field, Logonid database access is completely restricted.

Uid(*entry,...,entry*)
Restricts logonid record access so that it includes access to only the specified logonid records. These records are identified by UIDs or UID masks of 1 to 24 characters. Masking for this UID field works like masking for other eTrust CA-ACF2 UID fields. Trailing blanks are required if masking is not used. Use commas to separate multiple UID strings or masks. If the UID or LID field is not specified, Logonid database access is completely restricted.

Nextkey(entry)
Specifies the key of an alternate scope record that eTrust CA-ACF2 should check if access is denied based on the current scope record. A maximum of ten nextkey records are processed. Masking is not allowed in this field.

TARGET(null|=|?|*nodemask1,...,nodemask100*)
Specifies that you want to process scope records for a target node or group of nodes.

- **null**—Indicates that you want to process ACF subcommands at the home node only.
- **=** (equal sign)—Indicates that you want to process ACF subcommands at the home node only.

- ***? (question mark)*** — Indicates that you want to process ACF subcommands at the default target nodes.
- ***nodemask1,...,nodemask100*** — Indicates the target nodes where you want the ACF subcommands to be processed. You can specify up to 100 node names or masks. Separate entries with a comma.

CHANGE Subcommand

The CHANGE subcommand lets you change existing scope records. You must have the SECURITY privilege to change scope records. The CHANGE subcommand has the following syntax:

```
CHAnge     {*|recid|Like(recidmask)}
           [ADD|REP|DEL]

           {[Dsn(entry,...,entry)]}
           {[Inf(entry,...,entry)]}
           {[Lid(entry,...,entry)]}
           {[Uid(entry,...,entry)]}
            [Nextkey(entry)]
           [TARGET(null|=|?|nodemask1,...,nodemask100)]
```

In its simplest form, this subcommand requires a scope record name and at least one of the fields: DSN, INF, LID, or UID. For example, INSERT and CHANGE subcommands:

```
insert limit3 dsn(payroll) lid(alt-) uid(1h*pr)
change limit3 lid(pay-)
list limit3
 ACF60062 SCOPE LIMIT3 STORED BY PAYSDH ON 07/15/04-11:43
 DSN(PAYROLL) LID(ALT-,PAY-) UID(1H*PR)
```

The example CHANGE subcommand changes a scope record named LIMIT3 that was created with the INSERT subcommand. After the change, LIMIT3 has a DSN field with the value PAYROLL, a LID field with the value PAY- and a UID field with the value 1H*PR. The ADD parameter is the default.

Parameter Descriptions

The CHANGE subcommand takes the following parameters:

*** (asterisk)**
Specifies that the change applies to the last scope record processed since you established the SCP setting.

recid
Specifies the one to eight-character name of the existing scope record to be changed.

Like(*recidmask*)
Specifies a one to eight-character mask for the names of scope records to be changed.

ADD

Indicates that the values specified by the DSN, INF, LID, and UID fields of the CHANGE subcommand are added to the existing fields. For example, the following CHANGE subcommand changes the scope record named SCOPE1 that was created by the example INSERT subcommand:

```
insert scope1 dsn(payroll) lid(pay-) uid(1h*pr-)
change scope1 add dsn(account)

list scope1
 ACF60062 SCOPE SCOPE1 STORED BY PAYSDH ON 07/15/04-11:43
 DSN(PAYROLL,ACCOUNT) LID(PAY-) UID(1H*PR-)
```

As listed after this change, SCOPE1 contains a DSN field with both the values PAYROLL and ACCOUNT. The ADD parameter is assumed if you omit the ADD, REP, and DEL parameters. This parameter is positional.

REP

Indicates that the values specified in the DSN, INF, LID, and UID fields of a CHANGE subcommand replace the values in the corresponding fields of the existing scope record. If a field specified in the CHANGE subcommand does not currently exist in the scope record, it is added.

For example, the following CHANGE subcommand changes the scope record named SCOPE1 that was created by the example INSERT subcommand:

```
insert scope1 dsn(payroll) lid(pay-) uid(1h*pr-)
change scope1 rep lid(act-)

list scope1
 ACF60062 SCOPE SCOPE1 STORED BY PAYSDH ON 07/15/04-11:43
 DSN(PAYROLL) LID(ACT-) UID(1H*PR-)
```

As listed after this change, SCOPE1 contains a DSN field with the value PAYROLL, and a LID field with the value ACT- (which replaced the value PAY-). This parameter is positional.

DEL

Indicates that the values, specified by the DSN, INF, LID, and UID fields of the CHANGE subcommand, are deleted from the corresponding fields in the specified scope record. For example, the following CHANGE subcommand changes the scope record named SCOPE1 that was created by the example INSERT subcommand:

```
insert scope1 dsn(payroll) lid(pay-) uid(1h*pr-)
change scope1 del lid(pay-) uid(1h*pr-)
```

As listed after this change, the scope record SCOPE1 contains only a DSN field with the value PAYROLL. Both the LID and UID field values have been deleted. This parameter is positional.

Dsn(*entry,...,entry*)

Restricts any existing data set access so that it includes access to only the specified data sets (and any data sets permitted by access rule sets or other special privileges). Identify the data sets by high-level indexes or a mask. Use a one to eight-character alphanumeric mask. You must separate multiple masks or high-level indexes with commas.

If you do not specify the DSN field, data set access is completely restricted. However, access can be granted through access rule sets or special privileges. Scoped users with similar authority, as listed on the DECOMP field of the GSO RULEOPTS record, can still decompile rules, regardless of the limits imposed on their scope. See Parameters and Fields That Affect Scopes earlier in this chapter for details about what can override the values for DSN.

Inf(*entry,...,entry*)

Restricts Infostorage database access so that it includes access to only the specified infostorage records included within the mask such as resource rule sets, entry records, scope records, shift records, zone records, control records, and so forth.

Identify the records with a one to 44-character storage key. This storage key consists of the following:

- A letter code indicating the storage class, as follows:
  - **C**—Control records
  - **D**—DB2 resource rule sets
  - **E**—Entry records
  - **F**—Field records
  - **I**—Identity records
  - **P**—Profile records
  - **R**—Resource rule sets
  - **S**—Scope records
  - **T**—Shift and zone records.
  - **X**—Cross-reference records
- The three-character type code that identifies the type of record or resource rule set. (This type code corresponds with the one used to establish the CONTROL, DB2, ENTRY, FIELD, IDENTITY, PROFILE, RESOURCE, SCOPE, SHIFT, or XREF setting.)

- A system or division identifier that corresponds to the APPLDIV field of the GSO APPLDEF record. The length of the division is specified in the APPLDLEN field. The size of the identifier is normally 8 bytes, but can be any value from 1 to 8 bytes in length. The division identifier defines the system ID (SYSID), network node, site routine, and so forth to which the record applies. Record classes E, F, R, S, and T do not have division names.
- The key of the resource rule set or the name of the Infostorage database record.

For example, if the resource rule storage key consists of the following:

- **R**—the storage class for a resource rule set
- **CKC**—the type code for CICS transactions
- **ACFM**—the name of the transaction

Then, the entry in the INF parameter is INF(RCKCACFM-). Or, if the control record storage key consists of the following:

- **C**—the storage class for control records
- **GSO**—the type code for global system options
- **CPU1bbbb**—the eight character division or system identifier for the record name of the transaction, where bbbb are blank spaces
- **OPTS**—the name of the infostorage record

Then, the entry in the INF parameter is INF(CGSOCPU1bbbbOPTS-).

Separate multiple storage key masks with commas. If you do not specify this field and you specify the DSN, LID, or UID parameter, access to resources and to the Infostorage database is completely restricted.

However, access can be permitted by resource rules, special privileges, or pseudo data set names. Scoped users with similar authority, as specified in the INFOLIST field of the GSO OPTS record, are still able to list infostorage records regardless of the limits imposed on their scopes. See Parameters and Fields That Affect Scopes earlier in this chapter for details about what can override the values for INF.

Lid(*entry,...,entry*)

Restricts Logonid database access so that it includes access to only the specified logonid records. Identify the specified records by logonids or logonid masks of one to eight characters. Separate multiple logonids or masks with commas. If you do not specify this or the UID field, Logonid database access is completely restricted. However, access can be granted through special privileges.

Uid(*entry,...,entry*)
Restricts Logonid database access so that it includes access to only the specified logonid records. Identify these specified records by UIDs or UID masks of 1 to 24 characters. Masking for this UID field works like masking for other eTrust CA-ACF2 UID fields. You must use trailing blanks if you do not want to use masking. Separate multiple UID strings or masks with commas. If you do not specify this or the LID field, Logonid database access is completely restricted. However, access can be granted through special privileges.

Nextkey(entry)
Specifies the key of an alternate scope record that eTrust CA-ACF2 should check if access is denied based on the current scope record. A maximum of ten nextkey records are processed. Masking is not allowed in this field.

TARGET(null|=|?|*nodemask1,...,nodemask100*)
Specifies that you want to process scope records for a target node or group of nodes.

- **null**—Indicates that you want to process ACF subcommands at the home node only.
- **=** (equal sign)—Indicates that you want to process ACF subcommands at the home node only.
- **?** (question mark)—Indicates that you want to process ACF subcommands at the default target nodes.
- ***nodemask,...,nodemask***—Indicates the target nodes where you want the ACF subcommands to be processed. You can specify up to 100 node names or masks. Separate entries with a comma.

LIST Subcommand

The LIST subcommand lets you display existing scope records. The LIST subcommand has the following syntax:

```
List     {*|recid|Like(recidmask)}
         [ALL|DSN,INF,LID,UID,NEXTKEY]
         [TARGET(null|=|?|nodemask,...,nodemask)]
```

In its simplest form, this subcommand requires a scope record name.

```
l scope1
ACF60062 SCOPE SCOPE1 STORED BY PAYSDH ON 07/15/04-11:43
DSN(PAYROLL) LID(ACT-) UID(1H*PR-)
```

This example lists the contents of a scope record named SCOPE1. By default, all existing fields are listed.

Parameter Descriptions

The LIST subcommand takes the following parameters:

*** (asterisk)**
Specifies the listing of the last scope record processed since the current SCOPE setting was established.

recid
Specifies the one to eight-character name of an individual scope record to be listed.

Like(*recidmask*)
Specifies a one to eight-character mask for the names of scope records to be listed.

ALL
Indicates that all fields of the specified scope record are listed. ALL is the default.

DSN
Indicates that the DSN field of the specified scope record is listed.

INF
Indicates that the INF field of the specified scope record is listed.

LID
Indicates that the LID field of the specified scope record is listed. UID is required.

UID
Indicates that the UID field of the specified scope record is listed. LID is required.

NEXTKEY
Indicates the next scope record to be processed if access is denied from this scope record.

TARGET(null|=|?|*nodemask1,...,nodemask100*)
Specifies that you want to process scope records for a target node or group of nodes.

- **null**—Indicates that you want to process ACF subcommands at the home node only.
- **=** (equal sign)—Indicates that you want to process ACF subcommands at the home node only.
- **?** (question mark)—Indicates that you want to process ACF subcommands at the default target nodes.
- ***nodemask1,...,nodemask100***—Indicates the target nodes where you want the ACF subcommands to be processed. You can specify up to 100 node names or masks. Separate entries with a comma.

DELETE Subcommand

The DELETE subcommand lets you delete existing scope records. The DELETE subcommand has the following syntax:

```
DELete   {*|recid|LIke(recidmask)}
         [TARGET(null|=|?|nodemask,...,nodemask)]
```

In its simplest form, this subcommand requires a scope record name.

```
DELETE scope1
```

This example deletes a scope record named SCOPE1. All fields are also deleted.

Parameter Descriptions

The DELETE subcommand takes the following parameters:

*** (asterisk)**
Specifies deletion of the last scope record processed since the current SCOPE setting was established.

recid
Specifies the one to eight-character name of an individual scope record to be deleted.

LIke(*recidmask*)
Specifies a one to eight-character mask for the names of scope records to be deleted.

TARGET(null|=|?|*nodemask1,...,nodemask100*)
Specifies that you want to process scope records for a target node or group of nodes.

- **null**—Indicates that you want to process ACF subcommands at the home node only.
- **=** (equal sign)—Indicates that you want to process ACF subcommands at the home node only.
- **?** (question mark)—Indicates that you want to process ACF subcommands at the default target nodes.
- ***nodemask1,...,nodemask100***—Indicates the target nodes where you want the ACF subcommands to be processed. You can specify up to 100 node names or masks. Separate entries with a comma.

When you issue an online command that deletes multiple records from the eTrust CA-ACF2 databases, eTrust CA-ACF2 prompts you to confirm the DELETE LIKE request. If you respond Y, the command executes. If you respond N, or any other response, eTrust CA-ACF2 ignores the command and prompts you for the next command.

Activating Changes to Scope Records

eTrust CA-ACF2 loads scope records into storage at initialization time. Any changes you make to these records once eTrust CA-ACF2 is running (like adding a new record, changing an existing record, or deleting a record) do not take effect until you rebuild the records and the users address space is cycled.

To rebuild scope records dynamically, issue the following operator command. After issuing the command have the user logoff and log back on.

```
F ACF2,REBUILD(SCP),CLASS(S)
```

This causes eTrust CA-ACF2 to recognize any additions, changes, or deletions to those records so validation takes place according to the new definitions you have specified.

Note: In order for the changes to take effect you must signoff and sign back on to TSO.

For more information about the REBUILD command, see Appendix C, "Console Operator Commands" in the *Systems Programmer Guide*.

Chapter

5

Understanding SAF

The IBM System Authorization Facility (SAF) provides an interface that can direct control to all external security products, including eTrust CA-ACF2. The key component that SAF uses is the z/OS router. eTrust CA-ACF2 relies on the Computer Associates SAF interface, which makes extensive use of the z/OS router. These developments make eTrust CA-ACF2 compliant with IBM SAF. eTrust CA-ACF2 processes all SAF calls by default. In fact, in many instances, the information you can gain using the Computer Associates SAF interface is far more accurate and helpful for securing your resources.

This chapter discusses the following topics:

- The z/OS router and SAF
- How eTrust CA-ACF2 uses SAF and the z/OS router
- Components of the eTrust CA-ACF2 SAF interface
- Translating resource classes (CLASMAP)
- Defining environments for SAF calls (SAFDEF)
- Solving problems with SAF calls
- Data set loggings or violations
- Resource loggings or violations
- How eTrust CA-ACF2 processes SAF calls

The z/OS Router and SAF

The z/OS router is a system service that SAF uses to process security calls. It is not part of RACF or SAF. The z/OS router acts as a central point of control for all system products that provide resource control. The components of system products that provide resource management call the z/OS router for information that they need to make processing decisions. For example, a system product may call the z/OS router for decisions on access control. IBM refers to these functions as control points. The z/OS router determines that these calls should be handled by SAF. SAF is the single interface that all system products must invoke to communicate with any external security product for processing that takes place at one of these control points.

How eTrust CA-ACF2 Uses SAF and the z/OS Router

eTrust CA-ACF2 uses the Computer Associates SAF interface to interface with SAF and the z/OS router. The interpreter takes into consideration NEXTKEY and XREF processing. The Computer Associates SAF interface gets control **before** the z/OS router to provide support for mandatory access control (MAC). This placement also lets the Computer Associates SAF interface catch all SAF calls and determines how to process them. This means that eTrust CA-ACF2 can provide for their own unique processing issues **and** make use of the z/OS router as any other system product would. Therefore, when a system product invokes the z/OS router to request the services of an external security product, eTrust CA-ACF2 gets control. When a security event occurs, eTrust CA-ACF2 intercepts the call or processes a SAF call.

When another system product makes a request for security information, it uses the RACROUTE macro. The Computer Associates SAF interface intercepts these requests and processes them in terms that eTrust CA-ACF2 can understand. Some common requests with their eTrust CA-ACF2 translations are:

RACROUTE REQUEST	eTrust CA-ACF2 Translation
AUDIT	eTrust CA-ACF2 journals a Type=V SMF record for the specified audit event. This results in type TRC entries in the ACFRPTRV report. No SVC calls can be issued from this environment. For example, VTAM, APPC, and PSF make use of these types of calls.
AUTH, CLASS=DATASET AUTH, CLASS=DASDVOL AUTH, CLASS=TAPEVOL	eTrust CA-ACF2 performs data set validation for the request. You may need to define a SAFDEF record for the call. Note that DASDVOL and TAPEVOL are in the VOLUME.@volser or @volser.VOLUME format, depending on how your GSO RULEOPTS record is specified. All DASDVOL calls are processed as pseudo data set validation and are controlled by the VOLRULE setting of GSO RULEOPTS. For more information, see eTrust CA-ACF2 Rule Option Specifications (RULEOPTS) in the "Maintaining Global System Options Records" chapter and also see the "Maintaining Access Rules" chapter.
AUTH, CLASS=TSOAUTH	eTrust CA-ACF2 allows access, without rule checking, if the user's logonid record contains the equivalent TSO privilege. If the privilege is not turned on in the LID, then eTrust CA-ACF2 performs a resource validation.
AUTH, CLASS=others	eTrust CA-ACF2 performs a resource validation. The default CLASS is SAF. If another CLASS is specified in the RACROUTE macro, you must create a CLASMAP record to define the three-character resource type of the resource that you want to validate.
DEFINE, CLASS=DATASET DEFINE, CLASS=DASDVOL DEFINE, CLASS=TAPEVOL	eTrust CA-ACF2 performs data set validation for SAF calls with classes of DATASET, DASDVOL, or TAPEVOL. The DASDVOL and TAPEVOL entity names are in the VOLUME.@volser or @volser.VOLUME format, depending on the VOLRULE|NOVOLRULE parameter specification in the GSO RULEOPTS record. For more information, see eTrust CA-ACF2 Rule Option Specifications (RULEOPTS) in "Maintaining Global System Options Records" and also see the "Maintaining Access Rules" chapter.
DIRAUTH	eTrust CA-ACF2 processes requests to compare two security labels.

RACROUTE REQUEST	eTrust CA-ACF2 Translation
EXTRACT	eTrust CA-ACF2 executes the SAF call to extract the requested information from the eTrust CA-ACF2 databases, where applicable to eTrust CA-ACF2. See the "RACF to eTrust CA-ACF2 Translation" appendix for additional information on extract processing against fields within the BASE and TSO segments of the user profile. Standard SAF and SAF product return and reason codes are returned in all cases. Prior to r6.4, a SAF request that failed because of a non-zero return code from an accompanying eTrust CA-ACF2 TYPE=A SVC call would return a portion of the eTrust CA-ACF2 message identifier as the SAF product reason code. An example is a logonid update that fails because the logonid does not exist on the eTrust CA-ACF2 Logonid Database. This particular error would normally return the "ACF02010 LOGONIDS NOT FOUND" message. When this error occurs as a result of a SAF RACROUTE REQUEST=EXTRACT,TYPE=REPLACE call, eTrust CA-ACF2 would return the following SAF/SAF product return and reason codes: SAF return code=4 SAF product return code=16 (decimal) SAF product reason code=010 (decimal) where the SAF product reason code corresponds to the last three digits of the ACF00*nnn* message ID associated with the error. With eTrust CA-ACF2 r6.4, the standard documented SAF/SAF product return and reason codes will always be returned. This may result in some different return/reason code combinations, depending on the specific error that was encountered. For example, the previously noted failure will return the following SAF/SAF product return/reason codes under eTrust CA-ACF2 r6.4: SAF return code=4 SAF product return code=8 (decimal) SAF product reason code=8 (SAF product reason: Segment not found) Therefore, a number of sequences of SAF/SAF product return/reason codes can be expected for general error situations. For a complete list of the SAF/SAF product return/reason codes, see the SAF/SAF Product Return/Reason Codes section in Appendix B of this guide.

RACROUTE REQUEST	eTrust CA-ACF2 Translation
FASTAUTH	eTrust CA-ACF2 FASTAUTH processing retrieves the rule in storage if one exists and performs a resource validation. The validation takes into consideration both NEXTKEY and XREF processing. If access is allowed, eTrust CA-ACF2 sets an allow return code. If access is denied or no rule exists, eTrust CA-ACF2 checks for unscoped SECURITY or NON-CNCL. If these privileges are on, eTrust CA-ACF2 sets an "allow but log" return code. The caller is responsible for redriving the validation as a regular AUTH call. FASTAUTH processing does not call the eTrust CA-ACF2 SVC or z/OS SVC instructions so the following do not occur: ■ No SMF logging occurs for FASTAUTH calls unless parameters LOG=ASIS and RELEASE=2.1 or higher are specified in the RACROUTE macro. ■ No SVC exits are called. eTrust CA-ACF2 performs a FASTAUTH call only if resident rules exist. If the rules are not resident, the call gets a RC=8. See the "Maintaining Global System Options Records" chapter for information about the RESRULE, RESDIR, and INFODIR records and how to activate an infostorage record. If you have not made the rules resident, the FASTAUTH call creates a violation.
LIST	eTrust CA-ACF2 builds a resource rule directory for the specified resource class. The default CLASS is SAF. This type code must be specified in a RESDIR or INFODIR record. If the SAF call is for another CLASS, you must specify a CLASMAP record to translate the resource class into a three-character resource type. eTrust CA-ACF2 stores the resource type in the resource rule directory. See Steps 6 and 7 under Resource Loggings or Violations later in this chapter for more information.
STAT	eTrust CA-ACF2 verifies that security is active, that the class exists, and that the class is active. eTrust CA-ACF2 processes all SAF classes as ACTIVE.
TOKENBLD TOKENMAP TOKENXTR	eTrust CA-ACF2 TOKEN processing routines build, map, or extract TOKENs.

RACROUTE REQUEST	eTrust CA-ACF2 Translation
VERIFY	eTrust CA-ACF2 performs all VERIFY requests to build the user control block (ACEE and ACUCB) for the address space. eTrust CA-ACF2 performs system entry validation on the logonid associated with the address space. VERIFY requests are also supported for MUSASS environments. eTrust CA-ACF2 also builds the user control block for the address space (ACEE and ACMCB). The ACEE group table (ACEECGRP) is filled in by using the OMVS Group Facility. OMVS does not need to be started, but the system must be OMVS compatible and an OMVS Group Profile record must be defined. See the z/OS Unix System Services MVS Support chapter for more information about OMVS groups.
VERIFYX	eTrust CA-ACF2 performs all VERIFYX requests to validate a user, build a TOKEN, and return it to the caller.

For a complete list of the SAF resource classes defined to z/OS, see the "IBM-Supplied Resource Classes" appendix.

Components of the eTrust CA-ACF2 SAF Interface

To understand how eTrust CA-ACF2 processes SAF calls, you must be familiar with these components of the eTrust CA-ACF2 SAF interface:

- CLASMAP record
- SAFDEF record
- SECTRACE command
- eTrust CA-ACF2 access and resource rules

Since eTrust CA-ACF2 is SAF compliant, the eTrust CA-ACF2 SAF interface is always active. Even though eTrust CA-ACF2 processes all SAF calls by default, you can decide whether you want eTrust CA-ACF2 to validate particular SAF calls.

This section explains the function of each component. Individual components are explained in detail later in this chapter with examples that you can tailor to the unique calls you must define and validate.

CLASMAP Record

The GSO CLASMAP record translates an eight-character SAF resource class into a three-byte eTrust CA-ACF2 resource type code. eTrust CA-ACF2 checks the CLASMAP record for this type code for all SAF, #SECUR, CAISSF, HLI, and SVCA calls that set ACG8RTYP and ACF8CRTF.

To see the general CLASMAP record of RESOURCE(********) RSRCTYPE(SAF) defined by eTrust CA-ACF2, issue the following command:

```
SHOW CLASMAP
```

eTrust CA-ACF2 searches CLASMAP records in the following manner to determine the type code for a resource call:

- When a SAF general CLASMAP record exists, eTrust CA-ACF2 searches the external (site-defined) CLASMAP records. If the general CLASMAP record is the only match, eTrust CA-ACF2 searches the internal CLASMAP records provided with eTrust CA-ACF2. eTrust CA-ACF2 matches on the most specific CLASMAP regardless of it being an internal or external CLASMAP record. If no match is found, eTrust CA-ACF2 uses the general type code of SAF.
- When no SAF general CLASMAP exists, eTrust CA-ACF2 searches the external (site-defined) CLASMAP records. If no match is found, eTrust CA-ACF2 searches the internal CLASMAP records provided with eTrust CA-ACF2. If no match is found, eTrust CA-ACF2 uses the first three characters of the resource name as the type code.

SAFDEF Record

The GSO SAFDEF record identifies a SAF request and its environment to eTrust CA-ACF2. eTrust CA-ACF2 provides its own SAFDEF records for internal functions. Your site may have to provide SAFDEF records for IBM program products and any other system or application products that make SAF calls. eTrust CA-ACF2 defines general SAFDEF records to process SAF calls. These internal SAFDEF records provide eTrust CA-ACF2 protection by default. eTrust CA-ACF2 searches for a SAFDEF record that defines the specific request and the action you want eTrust CA-ACF2 to take when it validates the request.

Note: There are SAFDEF restrictions with FASTAUTH processing. FASTAUTH does not allow the use of ENTITY, JOBNAME, PROGRAM and RB on the RACROUTE field of the SAFDEF. Use of these on the SAFDEF for FASTAUTH can cause problems with FASTAUTH validations.

SECTRACE Command

The SECTRACE command is a diagnostic tool that lets you capture, format, and display the RACROUTE parameter list passed by requests for SAF services. For complete details and command syntax, see the Tracing SAF Requests section in the chapter, "Special Usage Considerations," in the *Systems Programmer Guide*.

eTrust CA-ACF2 Access and Resource Rules

Access and resource rules let you control how eTrust CA-ACF2 validates access to data sets and resources based on SAF requests. These requests can be made by other program products or system services.

Translating Resource Classes (CLASMAP)

The CLASMAP record translates eight-character resource classes into three-byte eTrust CA-ACF2 resource type codes. CLASMAP records are not required, but eTrust CA-ACF2 checks the CLASMAP record for the type code for all SAF, #SECUR, CAISSF, HLI, and SVCA calls that use the ACGRSRC parameter list and set the ACFG8RTYP and ACG8CRTF fields in ACGRSRC. When no matching CLASMAP record is found during validation, eTrust CA-ACF2 uses the first three characters of the resource class as the resource type. The three-character resource type code can let you write specific resource rules to validate security calls for a specified class.

A description of the CLASMAP record format and fields follows:

Record ID	Fields
CLASMAP*qual*	RESOURCE(*class*) RSRCTYPE(*typecode*) MIXED | NOMIXED MUSID(*musassid* | ********) ENTITYLN(0 | *entitylength*) PROFINT | NOPROFINT LOG | NOLOG

Field Descriptions

RESOURCE(*class*)
Specifies the explicit eight-character resource class from the CLASS keyword on the RACROUTE macro. RESOURCE can also define the resource class defined to eTrust CA-ACF2 by CAISSF. Standard eTrust CA-ACF2 resource name masking conventions apply.

RSRCTYPE(*typecode*)
Specifies the explicit three-character resource type code associated with the class. If you define a RESOURCE but do not define a RSRCTYPE, eTrust CA-ACF2 uses the first three characters of the RESOURCE as the RSRCTYPE. Use this type code to write resource rules to perform validation. This value cannot be masked. If you want to mask the name of the resource in your resource rule key, add this type code to the GSO RESDIR or INFODIR record and perform a REBUILD. For more information, see the "Maintaining Resource Rules" chapter.

MIXED | NOMIXED
Indicates whether eTrust CA-ACF2 accepts all input of resource names as mixed case. When MIXED is chosen, the inserted CLASMAP must be made active via the ACF2 REFRESH command before any subsequent administration commands can be issued for this resource class.

MUSID(*musassid* | ******)**
Identifies the MUSASS to which the CLASMAP record applies. This lets several MUSASSes that share the same resource class use different type codes. Standard eTrust CA-ACF2 resource name masking conventions apply.

ENTITYLN(0 | *entitylength*)
Specifies the entity length of the specified SAF class. This eliminates the need to use the class descriptor table (CDT). The default is 0. Zero causes eTrust CA-ACF2 to search its internal CLASMAP definitions; non-zero causes the GSO CLASMAP to be used. The resultant CLASMAP record, GSO or internal, is used for RSRCTYPE and ENTITYLN. If the resultant ENTITYLN is zero, eTrust CA-ACF2 assigns a length of 39, the IBM default.

PROFINT | NOPROFINT
Specifies whether the profile interpreter should be invoked for the profile record associated with this class. NOPROFINT is the default.

LOG | NOLOG
Specifies whether ACF2 will override the LOG parameter on a matching RACROUTE AUTH call and treat it as LOG=ASIS. This is a way of logging a violation to SMF that is not normally logged because the RACROUTE AUTH call specified LOG=NONE or LOG=NOFAIL or LOG=NOSTAT. NOLOG is the default. Note that LOG | NOLOG in the CLASMAP does not affect RACROUTE AUTH calls that are logged. NOLOG will not prevent loggings.

Creating Multiple CLASMAP Records

To create multiple CLASMAP records, append a qualifier to the record name in the format CLASMAP*qual* to generate a unique record ID (for example, CLASMAPVMAN or CLASMAP.DATASET). The total recid length is 16 bytes. The optional qualifier can be up to nine characters and must immediately follow the characters CLASMAP. If you use a period (.) as part of the qualifier string for the record name, eTrust CA-ACF2 counts it as one of the nine characters.

Viewing Internal and External CLASMAP Records

You can view the internal (eTrust CA-ACF2-defined) and external (site-defined) CLASMAP records by issuing the SHOW CLASMAP subcommand.

```
 CONTROL
show clasmap

    -- MERGED CLASMAP DEFINITIONS --

     MUSASS    RESOURCE   TYPE   ENTITY
       ID        CLASS    CODE   LENGTH    PROFINT    LOG  MIXED   EXTERNAL
     =======   ========    ===   ======    =======    ===  =====   ========
     ********  AC#CMD      SAF      8
     ********  ACAPPL      ACA     39
     ********  ACCBPROC    ACC     39
     ********  ACCTNUM     SAF     39
     ********  ACDIALOG    ACD     39
     ********  ACICSPCT    SAF     13
     ********  ACLIST      ACL     39
     ********  ACMSG       ACM     39
     ********  ACPANEL     ACP     39
     ********  ACREPORT    ACR     39
     ********  ACSQL       ACS     39
     ********  AIMS        SAF      8
     ********  AMARY       MAR     39                LOG  MIX     EXT
     ********  APPCLU      ALU     35      PROF
     ********  APPCPORT    SAF      8
     ********  APPCSERV    SAF     73
     ********  APPCSI      SAF     26
     ***
```

Validating SAF RACROUTE Calls

You must specify a CLASMAP record for the following types of SAF RACROUTE calls that you want to validate. For more information, see eTrust CA-ACF2 Rule Option Specifications (RULEOPTS) in the "Maintaining Global System Options Records" chapter and also see the "Maintaining Access Rules" chapter.

- REQUEST=AUTH,CLASS=DATASET|TAPEVOL|DASDVOL|others

  AUTH calls with a CLASS specification of DATASET, TAPEVOL, or DASDVOL result in data set validation. The DASDVOL and TAPEVOL entity names are in the VOLUME.@volser or @volser.VOLUME format, depending on the VOLRULE|NOVOLRULE parameter specification in the GSO RULEOPTS record.

  AUTH calls with any other CLASS specified result in resource validation.

- REQUEST=FASTAUTH,CLASS=DATASET|TAPEVOL|DASDVOL|others

  FASTAUTH calls with a CLASS specification of DATASET, TAPEVOL, or DASDVOL are ignored.

  FASTAUTH calls with any other CLASS specified result in resource validation without the issuance of an SVC call. Type codes must be placed in a resource rule directory (using the GSO RESDIR or INFODIR record) and the rules must be made resident for FASTAUTH calls to process correctly.

- REQUEST=LIST,CLASS=DATASET|TAPEVOL|DASDVOL|others

  LIST calls with a CLASS specification of DATASET, TAPEVOL, or DASDVOL are ignored.

  LIST calls with any other CLASS specified result in a directory build. SAF resource classes that issue these types of requests must be defined in a GSO RESDIR or INFODIR record.

In order to provide the recent performance benefits in the eTrust CA-ACF2 SAF component, a restriction was introduced in the manner that certain GSO SAFDEF records can be used.

When processing a SAF RACROUTE REQUEST=FASTAUTH request, eTrust CA-ACF2 will recognize only the following fields of SAFDEF records in determining whether to process or ignore the request:

```
MODE()
RACROUTE(REQUEST=FASTAUTH,SUBSYS=,REQSTOR=,CLASS=)
RACROUTE(REQUEST=FASTAUTH,SUBSYS=,REQSTOR=,CLASS=)
```

Other fields such as JOBNAME, PROGRAM, RB and RACROUTE(ENTITY=) will be ignored.

In effect, FASTAUTH resource validation can be globally enabled or disabled, but cannot be enabled for one set of users or entities and disabled for others.

The most common FASTAUTH resource is for PROGRAM validation. The following SAFDEF to enable program validation only for DFSMS programs is now invalid:

```
SAFDEF.sms MODE(global) PROGRAM(dgt-)
SAFDEF.sms MODE(global) PROGRAM(dgt-)
RACROUTE(REQUEST=FASTAUTH,REQSTOR=PROGMCHK,SUBSYS=CONTENTS)
```

To accomplish the same thing, you must create rules for the 'DGT' resources and use $KEY(********) to allow all other programs. Then insert a SAFDEF to globally enable SAF program validation:

```
SAFDEF.pgm MODE(global)
RACROUTE(REQUEST=FASTAUTH,REQSTOR=PROGMCHK,SUBSYS= CONTENTS)
```

Defining Environments for SAF Calls (SAFDEF)

The SAFDEF record defines the SAF environment and how you want eTrust CA-ACF2 to process the SAF call. eTrust CA-ACF2 provides internal SAFDEFs for SAF default protection. To see a list of the SAFDEF records active on your system, see Viewing SAFDEF Records later in this chapter. For a list of SAF resource classes, see “IBM-Supplied Resource Classes” appendix.

eTrust CA-ACF2 processes all SAF calls by default. To override the default SAF processing for a specific security event, you can specify a SAFDEF record. We provide examples of how to override SAFDEF records in Solving Problems with SAF Calls later in this chapter.

eTrust CA-ACF2 performs validation based on the environment you define in this record. SAFDEF records are processed in a specific sorted order based on fields in the record, whether it is an external or internal SAFDEF, and whether the SUBSYS is defined as ACF2 or GSO. The sort criteria is as follows:

1. Record key
2. JOBNAME
3. USERID
4. PROGRAM
5. RB
6. REQSTOR (RACROUTE parameter)
7. SUBSYS (RACROUTE parameter)
8. REQUEST (RACROUTE parameter)
9. SAFDEF (INTERNAL or EXTERNAL)
10. SUBSYS (ACF2 or GSO)

See the following pages for specific information on each of these fields.

A description of the record format and fields of the SAFDEF record follows:

Record ID	Fields
SAFDEF*qual*	ID(*name*) FUNCRET(4 | *retcode*) FUNCRSN(0 | *rsncode*) JOBNAME(mask | ********) MODE(IGNORE | GLOBAL | LOG | QUIET) NOAPFCHK | NONOAPFCHK PROGRAM(mask | ********) RACROUTE(*Keyword=value,…,Keyword=value*) RB(mask | ********) RETCODE(0 | 4 | 8) USERID(mask | ********)

Field Descriptions

ID(*name*)

Specifies an ID name associated with the record. You can specify up to eight characters. This field is optional. We recommend you specify an ID because this name will appear as the first field displayed in the SHOW SAFDEF output. The ID is also the key used for the SHOW SAFDEF subcommand.

Select a name that is unique and that conveys meaning about the type of SAF call you are defining. For example, VERSMS would be an appropriate ID for a SAFDEF record that defines the environment for a REQUEST=VERIFY call from DFSMS.

FUNCRET(4 | *retcode*)

Specifies the SAF function-dependent return code returned to the caller making the RACROUTE request when MODE is specified as IGNORE. For detailed descriptions of possible return codes, see the IBM document entitled *z/OS Security Server External Security Interface (RACROUTE) Macro Reference,* No. GC28-1922-04. The default is four.

FUNCRSN(0 | *retcode*)

Specifies the SAF function-dependent reason code to be returned to the caller making the RACROUTE request when MODE is specified as IGNORE. For detailed descriptions of possible reason codes, see the IBM document entitled *z/OS Security Server External Security Interface (RACROUTE) Macro Reference,* No. GC28-1922-04. The default is zero.

JOBNAME(*mask* | ******)**

Specifies the job names of the address spaces that apply to this SAFDEF record. You can specify an eight-character job name or a mask. The default is all job names.

MODE(IGNORE | GLOBAL | LOG | QUIET)

Specifies the mode you want eTrust CA-ACF2 to use to process this SAF request. This field defaults to GLOBAL; a value is required. You can specify any one of the following values:

- **IGNORE**—bypass processing this SAF request
- **GLOBAL**—process this SAF request using the mode specified in the GSO OPTS record. For generalized resource validations, use the eTrust CA-ACF2 SVCA recommendation to allow or deny the SAF request.
- **LOG**—process this REQUEST=AUTH call in LOG mode.
- **QUIET**—process this REQUEST=AUTH call in QUIET mode.

NOAPFCHK | NONOAPFCHK

STATUS=ACCESS is a keyword used in the RACROUTE REQUEST=AUTH security macro. It permits a user to interrogate security definitions (access and resource rules) to determine the access level for a user. No auditing is performed.

To maintain system integrity, eTrust CA-ACF2 requires that a user be authorized to access security definitions; however, some products that use STATUS=ACCESS are not authorized when they issue the request. The result is that eTrust CA-ACF2 abends the task with a S047 from ACF9C000.

To accommodate these products, eTrust CA-ACF2 lets the security administrator define the specific calls for which the authorization check for STATUS=ACCESS will be bypassed. This is done by using the NOAPFCHK keyword on a SAFDEF record that describes the specific environment from which this call is made. For example:

```
INSERT SAFDEF.apf PROGRAM(pgmname) RB(pgmname) NOAPFCHK
       RACROUTE(REQUEST=AUTH,CLASS=DATASET,STATUS=ACCESS)
```

Users who do not want to use this method should contact the vendor of the product and request that the STATUS=ACCESS call be made in an authorized environment.

PROGRAM(*mask* | ******)**

Specifies the program name of the current program request block (PRB) making the SAF request. If no PRB exists, on the active RB chain when the event occurs, the name for PROGRAM is the same as the name for RB. You can specify an eight-character program name or a mask. The default is all programs.

RACROUTE(Keyword=value,...,Keyword=value)

Identifies the SAF request being made. Use this field to specify any valid RACROUTE parameters and values. This is a multi-value field. The maximum length that you can specify for the parameter keyword, operator, and value is 64 characters. Separate the entries with commas or blanks. The REQUEST= keyword is required.

For example, an actual entity name of JES2.CANCEL.STC would be specified as ENTITY=JES2.CANCEL.STC **not** as ENTITY='JES2.CANCEL.STC'. Follow all RACROUTE macro coding conventions as described in the IBM document entitled z/OS Security Server External Security Interface (RACROUTE) Macro Reference, No. GC28-1922-04.

You can specify the following relational operators to indicate the presence of a particular value (for example, ENVIR=CREATE) or the presence of a pointer address (ACEE=>). You can use the following operators depending on your type of keyboard:

= Equal to

¬= Not equal to

<> Not equal to

!= Not equal to

=> Pointer value

¬=> No pointer value

!=> No pointer value

Pointer values are valid only if the keyword operand is specified as a pointer to a data area or data structure (for example, ACEE). When you specify a pointer value, do not also specify a value for the operand. For example, the following request defines a VERIFY request for all user IDs except JOHN, where an ACEE is supplied:

```
RACROUTE(REQUEST=VERIFY,ACEE=>,USERID¬=JOHN)
```

You can mask character data types using the standard eTrust CA-ACF2 masking characters (asterisk and dash). You can mask other types of data only if the mask is complete. A complete mask indicates that the parameter matches all values. For example, you can specify the following to indicate that this parameter matches all values of USERID:

```
USERID=-
```

Whereas, the following indicates that the USERID option does not apply to this RACROUTE request:

```
USERID¬=
```

Note: There are SAFDEF restrictions with FASTAUTH processing. FASTAUTH does not allow the use of ENTITY on the RACROUTE field of the SAFDEF.

RB(*mask*|******)**
Specifies the name of the request block (RB) where the security event occurs. When an event occurs directly under a PRB, you should specify the value for PROGRAM. When an event occurs under a supervisor call request block, specify the RB name as SVC*nnn*, where *nnn* is the decimal SVC number.

You can specify an eight-character RB name or a mask. The default is all request blocks.

RETCODE(0|4|8)
Specifies the SAF return code to be returned to the caller making the RACROUTE request when MODE is specified as IGNORE.

- **0**—Allow the request.
- **4**—Let the caller decide how to process the request. This is the default.
- **8**—Deny the request.

USERID(*useridmask*|******)**
Specifies the user ID of the address spaces that apply to this SAFDEF record. The default is all address spaces.

Creating Multiple SAFDEF Records

To create multiple SAFDEF records, append a qualifier to the record name in the format SAFDEF*qual* so that you can define a unique record ID for that SAFDEF record for a particular SYSID. The recid can be a maximum of 16 bytes. Therefore, you can specify a qualifier of up to ten characters. It must immediately follow the characters SAFDEF. If you use a period (.) as part of the qualifier string for the record name, eTrust CA-ACF2 counts it as one of the ten available characters.

For example, if you want to create a SAFDEF record for a VERIFY call from HSM and an AUTH call from HSM, you must use a qualifier to distinguish these two records. You could name the SAFDEF record for the VERIFY call SAFDEF.VERHSM and the SAFDEF record for the AUTH call SAFDEF.AUTHHSM. Naming records using qualifiers lets you describe multiple unique environments and lets eTrust CA-ACF2 add the records to the database with unique identifiers.

Viewing SAFDEF Records

The SHOW SAFDEF and SHOW ALL subcommands display the values in the SAFDEF record. The SHOW SAFDEF output displays the following information:

```
JESPOOLA JOBNAME=********   USERID=********   PROGRAM=********   RB=********
         RETCODE=0          SAFDEF=INTERNAL   MODE=IGNORE         SUBSYS=ACF2
```

You can interpret the output of the SHOW SAFDEF command as follows:

JESPOOLA
The ID field of the SAFDEF record is defined as ID(JESPOOLA).

JOBNAME=******
The JOBNAME field of the SAFDEF record was not specified so eTrust CA-ACF2 used the default, all jobs.

USERID=******
The USERID field of the SAFDEF record was not specified so eTrust CA-ACF2 used the default, all user IDs.

PROGRAM=******
The PROGRAM field of the SAFDEF record was not specified.

RETCODE=0
The RETCODE field of the SAFDEF record is defined as RETCODE(0) so that eTrust CA-ACF2 allows the request when MODE=IGNORE is specified.

SAFDEF=INTERNAL
This field indicates that this SAFDEF record was created by Computer Associates. If this record was created by your site in a SAFDEF record, the display would read SAFDEF=GSO.

MODE=IGNORE
The MODE field of the SAFDEF record is defined as MODE(IGNORE) so that eTrust CA-ACF2 bypasses processing of this request.

SUBSYS=ACF2
This field indicates that this record is for the eTrust CA-ACF2 subsystem. Another value that can be displayed here is MACS for the eTrust CA-ACF2 MAC subsystem. User-defined SAFDEF records are always targeted to all SAF processing subsystems (****).

SAFDEF Masks

A SAFDEF mask represents more than one SAFDEF record. This is illustrated by the following:

```
SHOW SAFDEF(GEN)
```

In this example, the SAFDEF mask can represent any SAFDEF that begins with the letters GEN and ends with up to eight characters. (A valid SAFDEF ID can contain up to eight characters.)

A SAFDEF mask can be defined by omitting ending characters, by using an asterisk (*), or by using a dash (-).

Omitting Ending Characters

A SAFDEF ID is automatically treated as a mask. For instance, the SAFDEF ID GENAUTH not only matches itself, it matches any string that begins with the characters GENAUTH and contains no more than eight characters.

By omitting characters, you can form a more general SAFDEF mask. For example, characters can be omitted from the SAFDEF ID to form a mask that represents all SAFDEFs that begin with GEN:

```
SHOW SAFDEF(GEN)
```

The mask matches any SAFDEF that begins with the characters GEN and contains up to eight total characters.

Using the Dash

A SAFDEF mask containing a dash must fit one of the following cases:

- If the dash falls at the end of a SAFDEF mask, it has the same effect as no dash. For example, the following two SAFDEF masks are equivalent:

  ```
  SHOW SAFDEF(GEN-)
  SHOW SAFDEF(GEN)
  ```

- If the dash is alone, then the SAFDEF represents all valid SAFDEFs:

  ```
  SHOW SAFDEF(-)
  ```

If the dash falls before or between any characters in the SAFDEF mask, it is treated literally as a dash and cannot represent any other character.

Using the Asterisk

A SAFDEF mask that contains asterisks must fit one of the following cases:

- Asterisks that fall at the end of the SAFDEF mask have the same effect as a dash or as no asterisks. For example, the following three SAFDEF masks are equivalent:

```
SAFDEF(GEN-)
SAFDEF(GEN****)
SAFDEF(GEN)
```

- If the asterisks are alone, then the asterisks represent all valid SAFDEFs.
- If the asterisks fall between or before any characters of a SAFDEF mask, then each asterisk represents exactly one character. For example, the mask GEN**T* matches any SAFDEF beginning with the letters GEN, followed by any two characters except nulls, followed by the letter T, and then followed by any other characters to form a SAFDEF ID of up to eight characters.

Using the Asterisk and Dash

If both asterisks and dashes are contained in a SAFDEF mask, they must fall at the end of the mask. They are equivalent to no asterisks or dashes. For example, the following two SAFDEF masks are equivalent:

```
SAFDEF(GEN*-)
SAFDEF(GEN)
```

Using Imbedded Blank Characters

A SAFDEF mask can contain an imbedded blank character; however, as with the dash, the blank character is treated literally as a blank character.

The SAFDEF REQ= Parameter

Using the REQ=*xxxx* parameter on the SHOW SAFDEF command displays all the SAFDEF records that match the RACROUTE REQUEST= keyword. This parameter must match a specific RACROUTE REQUEST= keyword; it displays only those SAFDEF records that match the RACROUTE REQUEST= keyword. It is not maskable. Use of this parameter limits the amount of output shown and displays the SAFDEF records in the order that they are processed.

In the following example, all the SAFDEF records that contain a specific match on the AUTH parameter are displayed:

```
SHOW SAFDEF(REQ=AUTH)
```

The output from this example might look like this:

```
 ACF
sho safdef(req=auth)
 -- SYSTEM AUTHORIZATION FACILITY DEFINITIONS --

 --SAF DEFINITIONS FOR REQUEST=AUTH

CATAUTH  JOBNAME=********   USERID=********   PROGRAM=********   RB=SVC026
         RETCODE=4          SAFDEF=INTERNAL   MODE=IGNORE        SUBSYS=ACF2
         FUNCRET=4          FUNCRSN=0

         RACROUTE REQUEST=AUTH,CLASS='DATASET'

APPL     JOBNAME=********   USERID=********   PROGRAM=********   RB=********
         RETCODE=4          SAFDEF=INTERNAL   MODE=IGNORE        SUBSYS=ACF2
         FUNCRET=4          FUNCRSN=0

         RACROUTE REQUEST=AUTH,CLASS='ACF9CSFV'

JESJOBS  JOBNAME=********   USERID=********   PROGRAM=********   RB=********
         RETCODE=4          SAFDEF=GSO        MODE=IGNORE        SUBSYS=****
         FUNCRET=4          FUNCRSN=0

         RACROUTE REQUEST=AUTH,REQSTOR='JESJOBS',CLASS='JESJOBS'

         JOBNAME=********   USERID=********   PROGRAM=********   RB=********
         RETCODE=4          SAFDEF=GSO        MODE=GLOBAL        SUBSYS=****
         FUNCRET=4          FUNCRSN=0

         RACROUTE REQUEST=AUTH,CLASS='APPL'

         JOBNAME=********   USERID=********   PROGRAM=********   RB=********
         RETCODE=4          SAFDEF=GSO        MODE=GLOBAL        SUBSYS=****
         FUNCRET=4          FUNCRSN=0

         RACROUTE REQUEST=AUTH,CLASS='DATASET'

         JOBNAME=********   USERID=********   PROGRAM=********   RB=********
         RETCODE=4          SAFDEF=GSO        MODE=GLOBAL        SUBSYS=****
         FUNCRET=4          FUNCRSN=0

         RACROUTE REQUEST=AUTH,CLASS='OPERCMDS'
```

Ignoring SAF Calls

Here is how to create a SAFDEF record when you want eTrust CA-ACF2 to ignore the SAF request.

```
SET CONTROL(GSO)
LIST LIKE(SAFDEF-)
PRD1 / SAFDEF.XYZ LAST CHANGED BY USER01 ON 07/02/04-12:13
                    ID(AUTHXYZ) MODE(IGNORE) RETCODE(0)
                    PROGRAM(XYZ-) JOBNAME(XYZ-)
                    RACROUTE(REQUEST=AUTH,CLASS=DATASET)
```

In this example, the SAFDEF record is for the XYZ product. Program XYZ123 makes a SAF call that eTrust CA-ACF2 intercepts. In this case, suppose the SAF call is a RACROUTE REQUEST=AUTH,CLASS=DATASET. The site decided to instruct eTrust CA-ACF2 to bypass processing of this request because it did not want eTrust CA-ACF2 to validate these calls.

By specifying MODE(IGNORE) and RETCODE(0), eTrust CA-ACF2 lets program XYZ123 access the data set without creating a logging record. The site does not have to create a rule.

Validating SAF Calls

Here is how to create a SAFDEF record that instructs eTrust CA-ACF2 to validate the SAF request.

```
INSERT SAFDEF.CATWRK001|ID(CAT001) MODE(global)
           RACROUTE(REQUEST=AUTH,CLASS=DATASET,ENTITY=CATALOG.MVSICF1.VWRK001)
           RETCODE(4))
```

In this example, the SAFDEF record defines an environment to eTrust CA-ACF2. In this environment, a program issuing the RACROUTE call updates the ICF catalog for the WRK001 volume. The site wants eTrust CA-ACF2 to process the call normally. eTrust CA-ACF2 validates the data set access according to MODE specification in the GSO OPTS record. If the site wants to reduce the number of loggings, it should create an access rule like the one shown in the following example:

```
$KEY(CATALOG)
 MVSICF1.VWRK001 UID(USER-) W(A)
```

This rule lets USER- jobs update the catalog without creating a logging record. If a user JOHNQ tries to update the catalog, eTrust CA-ACF2 would create an SMF record of the event and deny the update request. Notice that the RETCODE (which in this case is four) has no meaning if MODE(GLOBAL) is specified.

Solving Problems with SAF Calls

The default SAFDEF records shipped with eTrust CA-ACF2 are designed to ensure smooth operation. However, you may experience some new loggings or violations when you upgrade to new releases of IBM program products or other vendor products. When this happens, follow the procedure described in this section to quickly solve the problem.

The procedure is different depending on whether the SAF call is requesting data set or resource validation. For data set access, the procedure is short.

1. Run the ACFRPTDS report.
2. Write access rules.

See Data Set Loggings or Violations next for detailed information.

The procedure for solving problems for SAF calls that request resource access is more involved because you may want to define a CLASMAP record and a SAFDEF record. The major steps of the procedure are listed in the following:

1. Run the ACFRPTRV report.
2. Create additional SAFDEF records.
3. Create CLASMAP records.
4. Update the GSO INFODIR record.
5. Write resource rules.
6. Refresh GSO records.
7. Rebuild resource directories.

See Resource Loggings or Violations later in this chapter for detailed information.

Data Set Loggings or Violations

This section describes the steps you can take to resolve unexpected data set loggings or violations.

Step 1: Run ACFRPTDS Report

Since eTrust CA-ACF2 is SAF-compliant, it intercepts all SAF calls from system components and other program products. Depending on the setting of the MODE field in the GSO OPTS record or any $MODE control statements in rules, eTrust CA-ACF2 may deny or log the SAF request because it violates access controls. These loggings and violations appear in the output from the ACFRPTDS report.

For example, you might receive the following output on an ACFRPTDS report:

```
ACF2 UTILITY LIBRARY - ACFRPTDS DATA SET ACCESS JOURNAL      - PAGE    1
DATE 07/30/04 (04.212) TIME 01.03 DATA SET VIOLATION RECORDS
DATA SET                                        UID
         ACCESS TYPE     RM-RC     INST     STAPE   LID              NAME                   PROGRAM  DDNAME   LVOL
         VOL    RULE     LOG TYPE  PATH             JNAME            SNAME    JOB #    CPUI SOURCE   DATE             TIME
A10AU36.AUDIT.PAPERS                            SPD99SPDJQP
        SAF    OUTPUT    NORULE                   SPDJQP           JOHN Q PROGRAMMER      DSNSPY   SPYFILE  WRK004
        WRK004 A10AU36  DSET VIO                 SPDJQP    IKJACCNT TSU 1666 SYSC LN352    00.211    07/29   13.59
-------------------------------------------------------------------------------------------------------------------------------
-
```

On this report, the DSNSPY program is trying to access the A10AU36.AUDIT.PAPERS data set. The indicators you need to pay attention to include the following:

- The data set name—A10AU36.AUDIT.PAPERS
- The UID of the user attempting the access—SPD99SPDJQP
- The nature of the violation—the DSNSPY program issued a SAF call to check whether SPDJQP could access the data set.

You need this information to write an access rule. Another value you may see in the ACFRPTDS report for a SAF call by another program product is EXTRNL. This usually indicates that the call is being issued by a product that acts as a multiple-user single address space system (MUSASS). DFHSM is one example. The solution is the same: run the ACFRPTDS report and write a rule.

If you get unexpected results for a SAF call and see no entry in the ACFRPTDS report, check the SECTRACE output for that call. It may be that the LOG= parameter is specified in the RACROUTE call to prevent loggings from taking place. If this is the case, turn the TRACE bit on in the logonid to override this setting and force an entry in the ACFRPTDS report.

Step 2: Write Access Rules

You can decide the action you want eTrust CA-ACF2 to take for this SAF call. For example, you could write the following rule to let only the manager of the systems programming group access the A10AU36 data set to view the audit records:

```
$KEY(A10AU36)
 AUDIT.PAPERS UID(SYSMGR) R(A)
```

Resource Loggings or Violations

This section describes the steps that you can take to resolve unexpected resource loggings or violations.

Step 1: Run ACFRPTRV Report

In this example, you receive the following output on the ACFRPTRV report that you view online:

```
ACF2 UTILITY LIBRARY - ACFRPTRV - GENERALIZED RESOURCE LOG - PAGE    1
DATE 08/26/04 (04.238) TIME 12.16
REQUESTED RESOURCE                                  REC  LOOKUP KEY
UID                      SOURCE   CPU  MODULE   DISP     DSP-MOD  KEY-MOD  SERV
    DATE     TIME  JNAME    LID      NAME                 PRE RMC INT PST FIN
R-SAF-MVS.DISPLAY.JOB                               *VIO  R-SAF-MVS
     USER01                      XE75          NO-REC      -        -     READ
04.238 08/26 11.57 CONSOLE  USER01   TESTACF                0   8   0   0  16
CLASS: OPERCMDS
RESOURCE NAME: RSAFMVS.DISPLAY.JOB
R-SAF-MVS.DISPLAY.JOB                               *VIO  R-SAF-MVS
      USER01                     XE75          NO-REC      -        -     READ
04.238 08/26 12.00 CONSOLE  USER01   TESTACF                0   8   0   0  16
CLASS: OPERCMDS
RESOURCE NAME: RSAFMVS.DISPLAY.JOB
```

On this report, USER01 is trying to issue an MVS CONSOLE command. The indicators you need to pay attention to include the following:

- The name of the requested resource—R-SAF-MVS.DISPLAY.JOB
- The return codes from eTrust CA-ACF2—this access attempt created a violation. In this case, the return code is 8 or no rule was found that allowed the access. eTrust CA-ACF2 allowed, but logged the access.
- The logonid of the user attempting the access—USER01.
- The SAF resource class—OPERCMDS. You can translate this into a three-character type code if you want to write a specific resource rule for this call or use the default type code, SAF.
- The full name of the resource, including class and type code—RSAFMVS.DISPLAY.JOB.

With this information, you can decide what you want to do. You could decide to bypass this type of SAF call. If you want eTrust CA-ACF2 to bypass the call and eliminate the loggings, create a basic SAFDEF record as shown in the following example:

```
set control(gso)
insert safdef.oprcmds id(opercmds) mode(ignore) retcode(4) -
racroute(request=auth,class=opercmds) rep
```

However, you may want to provide more security and validate this type of SAF call as follows:

- Write a resource rule using the type code SAF. This solution is valid, but, if you want to provide security for a number of SAF calls, the SAF type code is probably too general.
- Create a SAFDEF record to describe the environment of the call.
- Create a CLASMAP record to define the unique resource class that you want to validate.

If you get unexpected results for a SAF call and see no entry in the ACFRPTRV report, check the SECTRACE output for that call. It may be that the LOG= parameter is specified in the RACROUTE call to prevent loggings from taking place. If this is the case, turn the TRACE bit on in the logonid to override this setting and force an entry in the ACFRPTRV report.

Step 2: Create a SAFDEF Record

Recall that eTrust CA-ACF2 processes SAF calls by default. If you want to override the default processing or define a more specific SAFDEF record for this call, you can follow the procedures in this step.

Here is the ACFRPTRV report output again:

```
ACF2 UTILITY LIBRARY - ACFRPTRV - GENERALIZED RESOURCE LOG - PAGE    1
DATE 08/26/04 (04.238) TIME 12.16
REQUESTED RESOURCE                              REC  LOOKUP KEY
UID                     SOURCE   CPU  MODULE   DISP     DSP-MOD  KEY-MOD  SERV
    DATE     TIME  JNAME    LID      NAME                PRE RMC INT PST FIN
R-SAF-MVS.DISPLAY.JOB                          *VIO  R-SAF-MVS
      USER01                    XE75          NO-REC      -        -     READ
04.238 08/26 11.57 CONSOLE  H06L867  TESTACF               0   8   0   0  16
CLASS: OPERCMDS
RESOURCE NAME: RSAFMVS.DISPLAY.JOB
R-SAF-MVS.DISPLAY.JOB                          *VIO  R-SAF-MVS
      USER01                    XE75          NO-REC      -        -     READ
04.238 08/26 12.00 CONSOLE  H06L867  TESTACF               0   8   0   0  16
CLASS: OPERCMDS
RESOURCE NAME: RSAFMVS.DISPLAY.JOB
```

Here is information you need to code a SAFDEF record:

- JOBNAME. You can specify this field with the specific name of this job (CONSOLE), use a mask if other jobs with similar names make this type of SAF request (CON-), or use the default (all jobs).

  Note: Security events for the class OPERCMDS issued from other jobs (for example, subsystem consoles) are not defined by this SAFDEF record.

- RACROUTE parameters. All resource calls are AUTH calls. The classes will vary. The CLASS: field in this RV report specifies that the class is OPERCMDS. You might also want to specify the entity name if you want to create a very specific SAFDEF record. The entity name is the RESOURCE NAME: MVS.DISPLAY.JOB.

You decide the following:

- ID—the name you specify for the ID is used with the SHOW SAFDEF subcommand. You must provide an ID for every SAFDEF record. We recommend that you make the ID meaningful. For example, this request is for an AUTH call to the XYZ resource. You might give this SAFDEF an ID of OPERAUT.
- MODE—the MODE is not part of the ACFRPTRV output. You must specify the mode. For classes that you want to validate, you should specify GLOBAL so that eTrust CA-ACF2 processes the request as it does all other validations on your system.

Here is an INSERT subcommand based on the example:

```
acf
 ACF
set control(gso)
 GSO
insert safdef.opr id(opercmds) jobname(console) mode(global)
racroute(request=auth,class=opercmds) userid(-) rep
```

Step 3: Create a CLASMAP Record

Here is the ACFRPTRV report output again:

```
ACF2 UTILITY LIBRARY - ACFRPTRV - GENERALIZED RESOURCE LOG - PAGE    1
DATE 08/26/04 (04.238) TIME 12.16
REQUESTED RESOURCE                                REC  LOOKUP KEY
UID                      SOURCE   CPU  MODULE   DISP     DSP-MOD  KEY-MOD  SERV
    DATE     TIME  JNAME    LID      NAME                 PRE RMC INT PST FIN
R-SAF-MVS.DISPLAY.JOB                             *VIO  R-SAF-MVS
     USER01                      XE75          NO-REC      -        -     READ
04.238 08/26 11.57 CONSOLE  H06L867  TESTACF                0   8   0   0  16
CLASS: OPERCMDS
RESOURCE NAME: RSAFMVS.DISPLAY.JOB
R-SAF-MVS.DISPLAY.JOB                             *VIO  R-SAF-MVS
     USER01                      XE75          NO-REC      -        -     READ
04.238 08/26 12.00 CONSOLE  H06L867  TESTACF                0   8   0   0  16
CLASS: OPERCMDS
RESOURCE NAME: RSAFMVS.DISPLAY.JOB
```

Here is information you need to code a CLASMAP record:

- RESOURCE—the resource class is specified in the CLASS: field. Specify the OPERCMDS resource class in the CLASMAP record.

You decide the following:

- RSRCTYPE—the type code is the three-character type code you will use to write the resource rule. Select a meaningful type code to describe the type of SAF call. You might select OPR.
- ENTITYLN—the default is 0. If the class name is defined internally, eTrust CA-ACF2 obtains the length from its internal tables. Otherwise, eTrust CA-ACF2 uses the default value of 39. If your product is making a SAF call for a resource class that is not identified in an IBM program product guide, look for the entity length in that product's documentation set.

Here is a sample INSERT subcommand based on the example:

```
acf
 ACF
set control(gso)
 GSO
insert clasmap.opr resource(opercmds) rsrctype(opr)
```

Step 4: Update INFODIR Record

Add the resource type code you created in the CLASMAP record to the resource directory so eTrust CA-ACF2 can validate any resource rules you write and so you can mask resource names. Change your INFODIR record as follows:

```
acf
 ACF
set control(gso)
 CONTROL
change infodir types(r-ropr)
```

Step 5: Create Resource Rules

After you have decided you want eTrust CA-ACF2 to validate the SAF call and you have decided how you want eTrust CA-ACF2 to validate the SAF call, you can write a resource rule. Here is the ACFRPTRV report output:

```
ACF2 UTILITY LIBRARY - ACFRPTRV - GENERALIZED RESOURCE LOG - PAGE    1
DATE 08/26/04 (04.238) TIME 12.16
REQUESTED RESOURCE                                 REC  LOOKUP KEY
UID                    SOURCE   CPU  MODULE   DISP     DSP-MOD  KEY-MOD  SERV
    DATE     TIME  JNAME    LID      NAME               PRE RMC INT PST FIN
R-SAF-MVS.DISPLAY.JOB                          *VIO  R-SAF-MVS
     USER01                       XE75         NO-REC      -        -     READ
04.238 08/26 11.57 CONSOLE  H06L867  TESTACF              0   8   0   0  16
CLASS: OPERCMDS
RESOURCE NAME: RSAFMVS.DISPLAY.JOB
R-SAF-MVS.DISPLAY.JOB                          *VIO  R-SAF-MVS
     USER01                       XE75         NO-REC      -        -     READ
04.238 08/26 12.00 CONSOLE  H06L867  TESTACF              0   8   0   0  16
CLASS: OPERCMDS
RESOURCE NAME: RSAFMVS.DISPLAY.JOB
```

Here is information you need to create a resource rule:

- RESOURCE NAME—RSAFMVS.DISPLAY.JOB. The resource name that you want eTrust CA-ACF2 to validate. Specify up to 40 characters in the $KEY or the entire resource name on a rule entry line.

You need the following information from the CLASMAP record:

- RSRCTYPE—supply a three-character type code for the rule. Use the value you specified in the CLASMAP.OPR record.

Now, write a resource rule as follows to specify users who can issue operator commands:

```
ACF
 acf
set res(opr)
 RESOURCE
compile *
 .$key(mvs) type(opr)
 . display.- uid(oper-) allow
 .
store
```

This rule lets all operators issue z/OS DISPLAY commands.

At this point, you can test the rule using the TEST subcommand. Since eTrust CA-ACF2 performs a sample validation, the results that you receive from the TEST subcommand let you know the access decision eTrust CA-ACF2 makes.

Step 6: Refresh GSO Records

After you create the GSO CLASMAP and SAFDEF records for any SAF calls, you must refresh the eTrust CA-ACF2 options to make those records active. Otherwise, the records do not become active until the next time the system is IPLed.

To refresh a GSO SAFDEF or CLASMAP record, use the following eTrust CA-ACF2 operator commands:

```
F ACF2,REFRESH(safdef),SYSID(sysid),CLASS(c),TYPE(gso)
F ACF2,REFRESH(clasmap),SYSID(sysid),CLASS(c),TYPE(gso)
F ACF2,REFRESH(INFODIR),SYSID(sysid),CLASS(c),TYPE(gso)
```

Step 7: Rebuild Resource Directories

Anytime you insert or change resource rules, you must rebuild that resource directory; otherwise, the pointers to the new or changed rule are not built and eTrust CA-ACF2 does not recognize that the new rule exists.

To rebuild the resource directories, use the following eTrust CA-ACF2 operator command:

```
F ACF2,REBUILD(opr),CLASS(r)
```

How eTrust CA-ACF2 Processes SAF Calls

For most SAF calls, eTrust CA-ACF2 provides SAFDEF records that specify MODE(GLOBAL) to indicate that eTrust CA-ACF2 processes the SAF request according to the specification of the MODE field of the GSO OPTS record. However, there are other SAF calls that eTrust CA-ACF2 does not process by default for these reasons:

- The number of SAF calls of this type are so numerous that your site might experience performance problems.
- The SAF call is issued for a resource that is not within the eTrust CA-ACF2 protection-by-default philosophy.
- The SAF call does not apply to a eTrust CA-ACF2 system.
- The SAF call duplicates eTrust CA-ACF2 protection, as with data set OPEN calls.
- Certain SAF calls associated with dataset OPEN, EOV, and ALLOCATION are internally enforced or ignored. These SAF calls are identified in the SECTRACE output with SAFEDEF=+IGNORE or SAFDEF=+ENFORCE.

eTrust CA-ACF2 does not process calls for the following SAF classes by default:

SAF Class	SAFDEF	Description
APPCPORT	APPL	Validates port of entry (POE) classes with an associated session during SAF VERIFY and VERIFYX processing at system entry.
APPL	APPL	Validates a user's authority to access an application during SAF VERIFY and VERIFYX processing at system entry.
CONSOLE	APPL	Validates a user's authority to access an MCS console. Checks the ability of commands issued from the MCS console to access other resources.
DEVICES	DEVAUTH	Validates a user's authority to allocate devices, such as unit record devices, graphic devices, and teleprocessing and communication devices.

SAF Class	SAFDEF	Description
JESINPUT	APPL	Validates a user's authority to enter commands or jobs through a JES input device.
JESJOBS	JESJOBS	Validates a user's authority to submit and cancel jobs by jobname.
JESSPOOL	JESPOOLA	Validates a user's ability to access a job's spooled output.
OPERCMDS	OPRCAUTH	Validates a user's authority to issue operator commands.
PROGRAM	PROGMCHK	Validates a user's ability to load and execute a program.
PSFMPL	PSFMSTAT	Provides a validation by PSF for page labeling.
SDSF	SDSFAUTH	Validates a user's authority to issue SDSF commands.
SMESSAGE	SMESSAGE	Controls the ability to receive TSO messages. To validate this resource, insert the following SAFDEF override: `safdef.smsg id(smsg) mode(global) racroute(request=auth,class=smessage)`
TERMINAL	APPL	Validates a user's authority to access the system based on the terminal he uses.
WRITER	WRITER	Validates a user's ability to use an external writer, including printers and outbound NJE.

Your site can decide to process some or all of these SAF calls. For an example of how to create a SAFDEF record for a specific resource class, see Resource Loggings or Violations earlier or Appendix C, "Protecting Operator Commands."

The following example shows the default SDSFAUTH class that is displayed when you issue the SHOW SAF command:

```
SDSFAUTH JOBNAME=********  USERID=********  PROGRAM=******  RB=********
         RETCODE=4         SAFDEF=INTERNAL  MODE=IGNORE     SUBSYS=ACF2
         FUNCRET=0         FUNCRSN=0

         RACROUTE REQUEST=AUTH,CLASS='SDSF'
```

To override, insert the following SAFDEF record:

```
set control(gso)
 CONTROL
insert safdef.sdsf id(mysdsf) mode(ignore)
racroute(request=auth,class=sdsf) rep


XE / SAFDEF.SDSF LAST CHANGED BY USERID ON 02/16/04-12:22
                  FUNCRET(4) FUNCRSN(0) ID(MYSDSF) MODE(IGNORE)
                  RACROUTE(REQUEST=AUTH CLASS=SDSF) RETCODE(4)
```

SHOW SAFDEF Subcommand

To view the SAFDEF records active on your system, issue the following subcommand:

```
SHOW SAFDEF
```

Customizing Your SAF Environment

eTrust CA-ACF2 makes use of the SAF user exit ICHRTX00. This exit lets you customize your SAF environment to your unique needs. Used efficiently, it can also improve performance when suppressing calls, modifying input before eTrust CA-ACF2 validation takes place, or modifying the response from eTrust CA-ACF2.

For more information about this exit, see the IBM document entitled *z/OS Security Server External Security Interface (RACROUTE) Macro Reference,* No. GC28-1922-04.

eTrust CA-ACF2 SAF Return Codes

The return codes issued by eTrust CA-ACF2 for the SAF interface are similar to those documented in the *IBM z/OS Security Server External Security Interface (RACROUTE) Macro Reference Guide*. Two sets of return codes are returned to the caller. These are the SAF return code and the RACF return and reason codes.

SAF Return Code

The SAF return code is set to a value of 0, 4, or 8. The standard meaning of these codes is:

- 0. The requested function was processed and has completed successfully.
- 4. The requested function has not been processed.
- 8. The requested function was processed and has failed.

If an ACF01004 Logonid Not Found condition, eTrust CA-ACF2 returns a SAF return code of 8 instead of 4 as documented in the IBM guide referenced in the previous section.

RACF Return and Reason Codes

The RACF return and reason codes are more variable and are based on the type of call being processed. The RACF codes that are returned normally follow the codes documented in the IBM *Reference Guide*. There may be discrepancies where there is no code matching a specific eTrust CA-ACF2 function. This is especially true with the VERIFY and VERIFYX calls. The eTrust CA-ACF2 SAF interface sets RACF return and reason codes based on the message returned by the eTrust CA-ACF2 SVC when a failure occurs.

The following table describes the RACF return and reason codes set for a particular eTrust CA-ACF2 error message. This table is used to translate the message ids returned from the eTrust CA-ACF2 SVC. The last three bytes of the message are compared against the table to determine which RACF return and reason codes are used. The message id prefix is ACF01, except where noted. If no table entry is found, a value of 24(18)/00 (return and reason code) is used. The value in parenthesis is the equivalent hex value as shown in the IBM *RACROUTE Guide*. For example, if user-defined error messages are passed back in the eTrust CA-ACF2 NEWPXIT, an attempt is made to match the last three bytes of the passed message id against the translation table. If there is no match, the corresponding return and reason code is passed back. Otherwise, the default return and reason code is passed back.

eTrust CA-ACF2 Message ID	RACF Return Code	RACF Reason Code	Message Text
(ACF01)004	04(04)	0	Logonid not found
(ACF01)006	08(08)	0	Password not allowed for logonid
(ACF01)007	08(08)	0	Password required for logonid
(ACF01)008	48(30)	0	Unauthorized input source for logonid
(ACF01)010	28(1C)	0	Logonid canceled
(ACF01)011	28(1C)	0	Logonid suspended
(ACF01)012	08(08)	0	Password not matched
(ACF01)013	28(1C)	0	Logonid suspended, password violations
(ACF01)014	28(1C)	0	Logonid expired
(ACF01)015	08(08)	0	Invalid password syntax
(ACF01)017	12(0C)	0	Password expired
(ACF01)018	16(10)	0	Invalid new password syntax
(ACF01)019	16(10)	0	Password less than minimum length

eTrust CA-ACF2 Message ID	RACF Return Code	RACF Reason Code	Message Text
(ACF01)020	16(10)	0	New password less than minimum length
(ACF01)021	12(0C)	0	Password expired, cannot be altered
(ACF01)023	16(10)	0	New=old password and old one is expired
(ACF01)025	28(1C)	0	Logonid not active
(ACF01)026	24(18)	0	Access denied by EXPPXIT exit
(ACF01)030	48(30)	0	Logonid has STC attribute
(ACF01)031	48(30)	0	Logonid does not have STC attribute
(ACF01)032	48(30)	0	Logonid/Source combo not valid
(ACF01)033	16(10)	0	Invalid new password syntax for NJE
(ACF01)034	16(10)	0	New password less than minimum length NJE
(ACF01)035	24(18)	0	SEVPRE/SEVPOST exit failed request
(ACF01)036	24(18)	0	Invalid RC from SEVPRE/SEVPOSt exit
(ACF01)037	16(10)	0	New password denied by NEWPXIT exit
(ACF01)039	48(30)	0	Invalid grouping structure
(ACF01)045	52(34)	0	Not authorized for access to MUSASS
(ACF01)060	48(30)	4	Zone record not found
(ACF01)061	48(30)	4	Logonid time not within shift
(ACF01)062	48(30)	4	Error in shift processing routines
(ACF01)063	48(30)	4	Shift record not found
(ACF01)076	0(04)	0	DDB LID not found
(ACF01)097	04(04)	0	No default logonid specified
(ACF01)098	32(20)	0	eTrust CA-ACF2 not initialized
(ACF01)100	20(14)	0	Not authorized for group
(ACF01)104	56(38)	0	Not authorized for seclabel
(ACF01)115	16(10)	0	New password cannot equal logonid
(ACF01)116	16(10)	0	New password cannot be all numeric
(ACF01)117	16(10)	0	New password has reserved word prefix

eTrust CA-ACF2 Message ID	RACF Return Code	RACF Reason Code	Message Text
(ACF01)118	16(10)	0	New password matches a previous password
(ACF01)128	16(10)	0	Invalid syntax for new password
(ACF01)130	16(10)	0	New password less than minimum length
(ACF01)131	16(10)	0	New password equals old password
(ACF01)132	16(10)	0	New password not allowed
(ACF01)136	16(10)	0	Mindays violation
(ACF01)200	08(08)	0	Invalid password/authority
(ACF01)255	16(10)	0	Password rejected by NEWPXIT exit
(ACF01)778	48(30)	0	Authorization required for background jobs
(ACFFE)999	16(10)	0	New password equals previous password; denied by Note 12 (ACFEEPSW)

Note: These codes may not agree with your intent unless you carefully select the last three bytes of your message id to agree with this table.

HCD SAF Support

If you receive the following HCD message, check to see whether you are validating the SAF OPERCMDS resource:

```
IOS500I - NO SECURITY
```

- If OPERCMDS is being validated, write a rule for resource MVS.ACTIVATE to allow access.
- If OPERCMDS is not being validated, insert the following SAFDEF record to allow the request with a return code of zero:

```
INSERT SAFDEF.hcd ID(hcd) MODE(IGNORE) RETCODE(0) FUNCRET(0)
       RACROUTE(REQUEST=AUTH,CLASS=OPERCMDS,SUBSYS=IOS,REQSTOR=IOS-) REP
```

Translating SAF Access Levels

The following table shows how eTrust CA-ACF2 translates SAF access levels into eTrust CA-ACF2 service levels for resources and into data set accesses. The eTrust CA-ACF2 translated service levels are not hierarchical as described in the IBM *SDSF Release 1.4 Customization and Security Administration Guide*. This means that the eTrust CA-ACF2 resource rule you create must explicitly specify all appropriate service values.

SAF Access Level	eTrust CA-ACF2 Service Level	eTrust CA-ACF2 Data Set Access
Read	Read	Read
Update	Update	Write
Control	Delete	Write
Alter	Add	Allocate
Execute	Execute	Execute

JESSPOOL

Validation of the JESSPOOL resource class provides the ability to protect the JES2 and JES3 data sets that are generated when a job runs. This includes those created by JES, such as joblog, JCL and allocation messages data sets, and user-created data sets such as SYSIN and SYSOUT.

The security administrator can create rules to restrict access of sensitive data to authorized individuals.

The JESSPOOL resource name consists of a node name, userid, job name, job number, data set ID, and data set name. The data set name is specified in the JCL DD statement or defaults to a question mark. For example, if SYSOUT=A,DSN=&&OUTPUT is specified in a job, the JESSPOOL name may look like:

```
NODE1.USER1.USER1A.JOB00045.D0000104.OUTPUT
```

When DSN= is not specified, the JESSPOOL name may look like:

```
NODE1.USER1.USER1A.JOB00045.D0000104.?
```

See your *JES Initialization and Tuning Guide* for more details about JESSPOOL data set names.

Implementing JESSPOOL Validation

The steps required to implement JESSPOOL validation are:

- Determine a resource type code to use for JESSPOOL resources
- Specify that type code in a GSO CLASMAP record
- Write resource rules
- Activate validation through GSO SAFDEF.

Assuming you select type code **SPL** for JESSPOOL, insert the following GSO CLASMAP record:

```
SET C(GSO)

INSERT CLASMAP.spool RESOURCE(JESSPOOL) RSRCTYPE(SPL)
```

Before you write rules, consider the following:

- No validation takes place when the requesting userid matches the userid in the JESSPOOL resource name. In other words, users can access their own JES data sets; rules are not needed for every user on the system.
- When the SAF validation includes the receiver (RECVR) keyword, access is granted even though a rule does not exist. The receiver keyword specifies a userid that is allowed access when no matching rule is found.

  For example, suppose that a data set is transmitted between systems by use of the TSO XMIT command and that the JESSPOOL resource name contains the userid of the sender. When it is received, the JESSPOOL validation includes RECVR, specifying the userid of the person to which the data set was sent. During validation, eTrust CA-ACF2 searches for a rule written for the sender. If the userid attempting to access the JESSPOOL data set matches the value specified in the RECVR field, access is allowed.

  You can see that without RECVR, rules would have to be written for all remote userids that use XMIT.
- Although logging is performed when a rule specifies LOG or when an access is allowed through NON-CNCL or security privileges, logging is suppressed when a JESSPOOL access violation occurs. You can override this by specifying LOG in the JESSPOOL CLASMAP record. Because a job may be comprised of many JES data sets, and because JESSPOOL validation occurs for each of these data sets, excessive loggings and an inflated violation count would occur without suppression.
- For better performance, add the SPL resource type code to GSO INFODIR so that the rules are made resident in storage. This avoids the overhead of retrieving rules from the database.

To create a JESSPOOL resource rule, issue the following commands:

```
SET R(SPL)

COMPILE * STORE

$KEY(NODE1) TYPE(SPL)
 prodid.prodjob.- UID(prodcntl) ALLOW
 prodid.prodjob.- UID(-) PREVENT
 prodid.- UID(-) LOG
```

This rule lets production control personnel access JES data sets created by the production userid under the production job; all other users are prevented access. Everyone is allowed access to JES data sets created by the production userid when not running under the production job, but access is logged.

Note: Although not explicitly indicated in the above rule set, all users are allowed full access to their own data sets. Access is not allowed to any data set not under the production ID when the RECVR keyword is used by the SAF caller.

After rules are written, enable JESSPOOL validation by inserting a GSO SAFDEF record as follows:

```
SET C(GSO)

INSERT SAFDEF.spool ID(JESPOOLA) MODE(global)
       RACROUTE(REQUEST=AUTH,CLASS=JESSPOOL) REP
```

To activate this new record, remember to refresh your GSO records by issuing this command:

```
F ACF2,REFRESH(SAFDEF)
```

JESSPOOL Performance Hints

JES2 issues many SAF validation calls to secure the environment, with the bulk of these calls validating the JESSPOOL resource class. If you do not want to secure JESSPOOL or if you want to limit the types of JESSPOOL validations, you can implement JES2 exit 36 to suppress the SAF calls.

Documentation for JES2 exit 36 is found in the *JES2 Customization Guide*. A sample exit 36 is found in SYS1.SAMPLIB member HASX36A. Routine NOCREDEL suppresses the JESSPOOL validation calls associated with SYSIN and SYSOUT creation.

Add an additional check to the sample code to suppress all JESSPOOL validations made during the job startup. This additional check suppresses the validation calls made for creation of a job's JES system data sets, such as job log and JCL. Adding the following two lines just prior to the check for $SEADEL also improves performance in the JES2 address space:

```
CLI     XPLIND,$SEASSOC   Was the call for system data set create?
BE      BYPASS            Yes, then bypass
```

APPC/MVS

APPC/MVS is a z/OS facility that provides a callable interface that lets MVS applications communicate with other applications using *Advanced Program-to-Program Communication (APPC)*. These other applications can reside within or outside of the mainframe.

APPC/MVS uses the z/OS System Authorization Facility (SAF) to provide security. Providing security for APPC/MVS entails defining system logonids, writing resource rules for APPC/MVS security classes, and defining eTrust CA-ACF2 profile records. You can also validate source and target LUs.

Defining Required Logonids

APPC/MVS runs with two address spaces, APPC and ASCH, and a transaction initiator, ASCHINT. Define STC logonids for these; you may also want to assign the NON-CNCL privilege to these logonids. To accomplish this, issue the following subcommands:

```
SET LID

INSERT APPC STC NON_CNCL NAME(APPC/MVS)
INSERT ASCH STC NON_CNCL NAME(APPC/MVS)
INSERT ASCHINT STC NON_CNCL NAME(APPC/MVS)
```

The APPC/MVS MULTI_TRANS scheduling type runs as a multi-user single address space. The user-specified GENERIC_ID logonid should be added to the APPC lid.

Writing Resource Rules for APPC/MVS Security Classes

Resource validations are invoked for the security classes of FACILITY, APPCTP and APPCSI. eTrust CA-ACF2 resource rules are written to protect these resources. In all cases, when masking is used in the $KEY of resource rules, resource directories must be used; this is done by defining the resource type code in GSO RESDIR or INFODIR.

The FACILITY class validations are used to control access to database tokens and to protect MULTI_TRANS TP profiles. This security class is not unique to APPC/MVS and it may already be defined to eTrust CA-ACF2 SAF. If it is already defined, you need only add the new APPC/MVS FACILITY resource names to your existing FACILITY rules. These resource names are found in APPC/MVS documentation. If using the FACILITY class for the first time, you need to choose a three-character resource type code that represents the FACILITY class; by default, the type code is FAC. The GSO CLASMAP record can be used to map the FACILITY class to a different type code.

A sample resource rule follows:

```
SET R(FAC)

COMPILE *

$KEY(appcmvs) T(fac)
dbtoken UID(-) ALLOW
tp.multi UID(-) ALLOW

STORE
```

Resource classes APPCTP and APPCSI protect access to transaction profile information and side information. These classes are protected by default and the default resource type code is SAF. You can define your own resource type codes using GSO CLASMAP; the type codes for these two classes can be the same or unique depending on your implementation. In this example, we map both classes to resource type APP and compile a sample resource rule:

```
SET C(GSO)

INSERT CLASMAP.appctp RESOURCE(APPCTP) RSRCTYPE(app)
INSERT CLASMAP.appcsi RESOURCE(APPCSI) RSRCTYPE(app)

SET R(APP)

COMPILE *
$KEY(dbtoken) T(app)
userid.tpname UID(-) ALLOW

STORE
```

See APPC/MVS documentation for the format of the resource names to use for each security class.

Profile Support for APPC/MVS

Some APPC/MVS security-related data is kept in the security system database. When using eTrust CA-ACF2, this information is defined by eTrust CA-ACF2 profile infostorage records. APPC/MVS obtains this information through a SAF EXTRACT call.

User sysout and account information is kept in USER profile WORKATTR records. This information is used when the transaction profile selects the TAILOR_SYSOUT or TAILOR_ACCOUNT options.

VTAM LU security information is kept in the APPCLU profile SESSION records. This information is used by VTAM to validate the establishment of LU sessions and to provide conversation-level security options. This security call is made when the VTAM APPL statement for APPC/MVS specifies VERIFY=OPTIONAL or VERIFY=REQUIRED.

See the "Maintaining Profile Records" chapter for more information about these profile records.

LU-LU Entry Validation

APPC/MVS provides userid and password validation when a transaction program is initiated. You can also activate validation of the source and target LUs for that user. The user's ability to enter the system from a particular LU is validated with the APPCPORT resource class; the user's ability to access the target LU is done with the APPL resource class. You can activate validation of one or both of these classes. The APPL security class is not unique to APPC/MVS and may already be defined to eTrust CA-ACF2. Both resource classes use eight-character resource names that correspond to VTAM LU names. The default resource type code for these classes is SAF.

You can define a different type code through the GSO CLASMAP record as follows:

```
SET C(GSO)

INSERT CLASMAP.appl RESOURCE(APPL) RSRCTYPE(vlu)
INSERT CLASMAP.appcport RESOURCE(APPCPORT) RSRCTYPE(vlu)

SET R(vlu)

COMPILE *
$KEY(luname) T(vlu)
UID(-) ALLOW

STORE
```

Activation of each class is done by inserting a GSO SAFDEF record:

```
SET C(GSO)

INSERT SAFDEF.appl1 ID(appl1) MODE(global) -
       RACROUTE(REQUEST=AUTH,REQSTOR=ACF9CSFV,CLASS=APPL,SUBSYS=APPC/MVS) REP
INSERT SAFDEF.port1 ID(port1) MODE(global) -
       RACROUTE(REQUEST=AUTH,REQSTOR=ACF9CSFV,CLASS=APPCPORT,SUBSYS=APPC/MVS) -
       REP
```

Chapter 6

Maintaining Access Rules

An eTrust CA-ACF2 access rule specifies which system users and under what conditions those users can access an individual data set or a group of data sets. eTrust CA-ACF2 protects all data by default. It does not use standard OS password controls for controlling data set access. Instead, eTrust CA-ACF2 replaces them with its own methods. Data sets can be accessed only by:

- A user who has been granted specific access to the data set by means of an eTrust CA-ACF2 access rule.
- An owner of the data set as defined by the PREFIX field of the logonid record.
- A user with special override logonid privileges
  - NON-CNCL
  - SECURITY (unless RULEVLD is also on)
  - Scoped SECURITY (SCPLIST specified and RULEVLD is also off)
  - READALL (if access is for READ or EXECUTE only).

All data on a system running eTrust CA-ACF2 is protected, even when new data sets are created.

Note: eTrust CA-ACF2 provides you the ability to go beyond the level of data set level security, that is, security of the data set as a whole. The eTrust CA-ACF2 member level protection facility lets you secure individual members within a partitioned data set (PDS) based on the name of the member. This lets you extend eTrust CA-ACF2 security control and audit capabilities to the individual member level within a PDS. For more information about implementing and using PDS member-level protection controls, see Appendix D, "Implementing Member-Level Protection."

This chapter describes:

- An example of a eTrust CA-ACF2 access rule set
- The elements of the eTrust CA-ACF2 access rule set including control statements, access rule entries, and comment statements
- Masking in data set names and UIDs
- Use of NEXTKEY

- eTrust CA-ACF2 features that simplify access rule writing
- The rule selection algorithm
- Creating access rule sets
- Maintaining rules using the ISPF panels and the ACF command
- Sample rules
- Program pathing control
- Execution flow

Example of a eTrust CA-ACF2 Access Rule Set

When listed, a simple access rule set might look like this:

```
$KEY(payroll)
 master.data UID(tfinpaynlt) READ(A) WRITE(A) EXEC(A)
```

This access rule set pertains to all data sets with a high-level index of PAYROLL. Only one rule entry exists in this rule set, and is interpreted as follows:

- MASTER.DATA specifies that this rule entry pertains only to the data set PAYROLL.MASTER.DATA.
- UID(TFINPAYNLT) specifies the user identification string (UID) of the users to whom access to the data set is granted.
- READ(A) WRITE(A) EXEC(A) specifies that these users have read, write, and execute authority to the data set. Because ALLOC(A) is not specified, allocate authority is assumed to be prevented—that is, ALLOC(P).

An access rule set consists of access rule entries that pertain to data sets of a particular high-level index. Each access rule set is made up of:

- Control statements
- Access rule entries
- Comment statements

Control Statements

In previous sample rule set, the first line contains a control statement. The second line is an access rule entry. Control statements must all begin in column one. The two types of control statements are the dollar sign ($) and percent symbol (%). A rule set can contain only one of each type of $ control statement but as many of each type of % control statement as desired. The $KEY control statement is the only required control statement.

Where to Place Control Statements

Here are some guidelines to follow when specifying control statements:

- Specify the appropriate $ or % control statements in a rule set. The $KEY statement is required. All others are optional.
- If you specify the same $ control statement more than once, eTrust CA-ACF2 uses the last entered value only.
- Specify all $ control statements before any other statements.
- Specify multiple $ control statements on one line if the $ appears in column one:

```
$KEY(sys1) MODE(abort) OWNER(joesmith)
```

How to Continue Control Statements

To specify that you want a control statement to continue onto the next line, use a dash (–) as the last nonblank character on the line. eTrust CA-ACF2 recognizes a continuation character unconditionally. For example, if a comment statement is continued, the next line is treated as a continuation of the comment statement, even if that line has the format of a control statement.

Format Requirements for Control Statements and Rule Entries

Input can be in variable format, with the sequence field as the first eight characters of the record (as in TSO CLIST or PL/I), or as fixed-format 80-byte records with the sequence field as the last eight characters in the record (as in DATA or CNTL type data sets). You can specify multiple $ control statements on the same line with a single $ in column one.

For example, the following formats can be used:

```
$KEY(sys9)
$PREFIX(sys*)
$USERDATA(user-comments)
```

Or

```
$KEY(sys9) PREFIX(sys*) USERDATA(user-comments)
```

Control Statement Syntax

Following is the syntax of the access rule control statements:

```
$Key(high-level-index)
[$MEMBER(membername)]
[$Mode(Quiet|Log|Warn|Abort)]
[$NORULELNG]
[$NOSort]
[$Owner(owner-id)]
[$Prefix(prefix)]
[$RESOWNER(resource-owner)]
[$Userdata(userdata)]
[%Change uidmask1,...,uidmaskn]
[%Rchange uidmask1,...,uidmaskn]
```

Control Statement Descriptions

$Key(*high-level-index*)

Specifies the high-level index of the data set name for which this rule is being written or the VSAM key of the rule set. You can specify one to eight characters. For example, when you compile a rule set to permit access to the data set SYS1.PARMLIB, the $KEY control statement contains $KEY(SYS1), because SYS1 is the high-level (or first) index of the data set name. During access validation, the $KEY value is used as the prefix unless the $PREFIX control statement is specified.

When writing rules for your own data sets, the $KEY control statement usually contains your logonid, since the logonid is often specified as the prefix for your own data sets. This field cannot be masked.

$KEY is the only required control statement. All others are optional.

$MEMBER(*membername*)

Specifies the member name to be used for a decompile into a partitioned data set (PDS) if one is not provided with the decompile request. For an explanation of how the $MEMBER control statement affects processing of the DECOMP subcommand, see the section on the DECOMP subcommand later in this chapter.

When you specify a $MEMBER name that matches the $KEY value for the access rule, eTrust CA-ACF2 issues a warning message and ignores the $MEMBER control statement.

$Mode(Quiet|Log|Warn|Abort)

Specifies the system mode you want eTrust CA-ACF2 to use when it validates access for this rule. To make eTrust CA-ACF2 enforce the $MODE control statement for a rule, you must also set the MODE field in the GSO OPTS record to RULE. (See the description of the MODE field in eTrust CA-ACF2 Option Specifications (OPTS) in the "Maintaining Global System Options Records" chapter.)

If you specify the GSO OPTS MODE as RULE and specify a $MODE statement, eTrust CA-ACF2 bases its validation on the value you specify in the $MODE control statement of the rule. For example, if you specify $MODE(LOG), access violations are logged but permitted. If you specify $MODE(ABORT), access violations are logged and denied. Valid modes affect the access rule disposition as follows:

- **$MODE(Quiet)**—specifies that all accesses to any data set covered by this rule set that are not specifically permitted by a rule are permitted. Accesses to data sets that are explicitly prevented by rules are also permitted. No eTrust CA-ACF2 data set or program SMF loggings are to occur.
- **$MODE(Log)**—specifies that all accesses to any data set covered by this rule set that are not specifically permitted by a rule are permitted, but access is logged. eTrust CA-ACF2 data set or program SMF loggings are to occur.
- **$MODE(Warn)**—specifies that all accesses to any data set covered by this rule set that are not specifically permitted by a rule are permitted, but the site warning message (defined in the GSO WARN record) is issued with the eTrust CA-ACF2 violation message (ACF99913). In addition, eTrust CA-ACF2 data set and program SMF loggings occur.
- **$MODE(Abort)**—specifies that all accesses to any data set covered by this rule set that are not specifically permitted by a rule are denied. eTrust CA-ACF2 console messages are issued and eTrust CA-ACF2 data set and program SMF loggings occur.

The $MODE control statement takes effect when access to the specified data set is denied for that user or if the specific data set name accessed is not contained in the rule set.

Here is a sample rule set with the $MODE statement specified as LOG:

```
$KEY(acctpay)
$MODE(LOG)
 monthly.data UID(acctuid) R(A) W(A)
```

Only the user whose UID is ACCTUID is permitted access to ACCTPAY.MONTHLY.DATA. This user has read or write privileges to this data set. If this user attempted an access to this data set other than read or write, this rule does not apply. This rule also does not apply to a user accessing this data set whose UID is not ACCTUID. If no rule applies, eTrust CA-ACF2 checks the current GSO OPTS record MODE field. In this example, if eTrust CA-ACF2 were in RULE mode, access is permitted based on the $MODE(LOG) control statement in the access rule.

You can also use the $MODE control statement to migrate to full security by phasing in protection at the rule set level. Use of this control statement also eliminates the need for a violation or postvalidation exit for this type of checking during the transition period.

$NORULELNG

Overrides the use of the rulelong compiler when RULELONG is active. Normally, eTrust CA-ACF2 uses the rulelong compiler to compile rules if the RULELONG option is set. The rulelong format is an expanded record format. If a ruleset is small and therefore does not require the rulelong format, specifying NORULELONG on a compile lets you compile a ruleset using a compact record format. This way, you can choose to compile rules with the format that is required for the ruleset.

Note: If $NORULELNG was specified in the rule, this option is not necessary. If the dynamic compile option (COMPDYN) is set in the GSO RULEOPTS record then the $NORULELNG control statement is not needed to compile rule sets of varying size.

$NOSort

Specifies that eTrust CA-ACF2 **should not** sort access rule entries when it stores this rule set. The rules remain in the order in which they were first entered into the compiler through the terminal or a partitioned data set. eTrust CA-ACF2 issues a warning message after compilation whenever a $NOSORT statement is used because eTrust CA-ACF2 normally sorts and then stores a set of access rule entries from most specific to most general in terms of access environment. For example, PAYROLL.PROD.– is more specific than PAY–.–. A general access rule might prevent a more specific rule from being evaluated if it is entered before a more specific rule. It is recommended that you avoid the use of $NOSORT.

The $NOSORT statement is effective only when you have specified $NOSORT in the GSO RULEOPTS record.

The $NOSORT option is described in eTrust CA-ACF2 Rule Option Specifications (RULEOPTS) in the "Maintaining Global System Options Records" chapter.

$Owner(*ownerid*)

Specifies the owner of the rule. You can specify up to 24 characters in the $OWNER control statement. eTrust CA-ACF2 provides the $OWNER statement in case you want to track ownership of a rule. eTrust CA-ACF2 does not use this field for any processing. The $OWNER statement does not grant the specified user any special privileges regarding the rule set.

eTrust CA-ACF2 stores the $OWNER data with the rule set and displays it when you decompile the rule set (similar to the $USERDATA information). eTrust CA-ACF2 also stores the $OWNER data in SMF records that can be used to produce reports. You can use your own conventions for the information placed in $OWNER to facilitate reporting methods.

$Prefix(*prefix*)

Specifies a value that overrides the rule set key as a prefix to all data set names in this rule set. You can specify up to 40 characters. When a $PREFIX is specified and the rule set is not selected by NEXTKEY, data set names in the rules are prefixed with the $PREFIX value.

To indicate that you want eTrust CA-ACF2 to check an alternate rule set, you must also specify the NEXTKEY parameter in the individual rule entry. The value you specify for NEXTKEY directs eTrust CA-ACF2 to the alternate rule set. See the description of the NEXTKEY parameter and the Use of NEXTKEY section later in this chapter. Specify the $PREFIX control statement in the alternate rule set to indicate the true high-level index of the data set.

Use the NEXTKEY parameter to define a subsequent $KEY to be checked by eTrust CA-ACF2 or a site exit. When you use a local exit, the site-supplied prevalidation exit code must recognize that the search key must be set to something other than the data set name high-level index and modify the rule key to be used in the search.

If you specify $PREFIX(), the prefix is the $KEY entry. eTrust CA-ACF2 does not generate a $PREFIX statement when it decompiles the rule. eTrust CA-ACF2 issues a warning message to indicate that the $PREFIX is null and is ignored.

You can use the asterisk (*) or dash (-) masking characters to mask the $PREFIX value.

$RESOWNER(*resource-owner*)

Specifies the resource owner (RESOWNER) of the data set. You can specify up to eight characters for the logonid acting as the RESOWNER of the data set. For details on when to specify a $RESOWNER, see the "Implementing DFSMS Support" chapter.

The $RESOWNER data is stored with the rule set and is displayed when the rule set is decompiled (similar to the $USERDATA information). The $RESOWNER data is also contained in SMF records that can be used to produce reports.

$Userdata(*text*)

Specifies any text string up to 64 characters. eTrust CA-ACF2 stores the information you place in the $USERDATA field with the rule set. This information can be accessed by site postvalidation or violation exits.

Comment statements, represented by an asterisk (*) and a space in columns one and two, can also represent user information but are removed when the rule set is compiled. Therefore, you might want to use $USERDATA if you want the information to stay with the rule. You can also use the DATA parameter (described later in this chapter) to store user information in a rule entry in an access rule set.

%Change ***uidmask1,...,uidmaskn***
Specifies who, besides the high-level index owner and security administrator, can replace or delete a set of rules. Specify the UIDs or UID masks that identify the users who have this change authority. Separate multiple UIDs or masks with commas or spaces.

%CHANGE lets the designated user change the rule set to delegate change authority to other users. The designated user can change or delete any part of the rule set. To activate or deactivate the site's ability to use this control statement, see the description of the CHANGE field of the GSO RULEOPTS record in eTrust CA-ACF2 Rule Option Specifications (RULEOPTS) in the "Maintaining Global System Options Records" chapter.

To delegate rule writing authority, a security administrator can compile and store a rule set that contains only the $KEY and %CHANGE control statements (that is, no rule entries). This allows a skeleton rule set to be stored, awaiting refinement by the designated changer, without the security administrator having to write any rule entries.

%Rchange ***uidmask1,...,uidmaskn***
Specifies the UID strings or UID masks that identify the users who have restricted change authority over the rule set. A designated user can change individual rule entries, but not control statements. He cannot further delegate any change authority, nor can he delete the rule set.

If the same user matches entries in both %CHANGE **and** %RCHANGE, the %CHANGE authority prevails.

To activate or deactivate the site's ability to use the %RCHANGE control statement, see the description of the CHANGE option of the GSO RULEOPTS record in eTrust CA-ACF2 Rule Option Specifications (RULEOPTS) in the "Maintaining Global System Options Records" chapter.

Access Rule Entries

Individual access rule entries follow the control statements in a rule set. A rule entry can extend up to 72 positions and can be continued from one line to the next by use of a dash (–). If a line ends with a dash, the next line is interpreted as a continuation of that previous line. If a comment ends with a dash, the next line begins a continuation of that comment.

You should start rule entries in column two so that when you have an entry beginning with an asterisk, it is not treated as a comment line.

Rule Entry Syntax

The full syntax of an individual access rule entry is as follows:

```
dsnmask [Volume(volmask)] [UId(uidmask)] [SOurce(source-mask)]
    ft(shift)] [Library(libmask)]
    (pgmmask)|PRogram(pgmmask)]
[DDname(ddnmask)] [UNtil(date)|For(days)] [ACTIVE(date)]
    a(text)] [Nextkey(nextkey)] -
[Read(Allow|Log|Prevent)]
[Write(Allow|Log|Prevent)] -
[Allocate(Allow|Log|Prevent)]
[Execute(Allow|Log|Prevent)]
```

Components of a Rule Entry

The information in a rule entry can be classified into one of the following type.

Environment

Specifies what is to be accessed, who can access it, and under what conditions the access can take place. The parameters that specify the environment are as follows:

- ACTIVE
- DDNAME
- DSNMASK
- FOR
- LIBRARY
- PGM
- PROGRAM
- SHIFT
- SOURCE
- VOLUME
- UID
- UNTIL

See Rule Entry Parameters later in this chapter for a description of these parameters.

Access Permission

Specifies whether the access is allowed, logged, or prevented, as described later. Possible types are:

eTrust CA-ACF2 Data Set Access	SAF Access Level
Read	Read
Write	Control
Execute	Read
Allocate	Alter

For each of these types you can specify the following actions:

- Allow
- Log
- Prevent

Pointer

Specifies an alternate access rule set that eTrust CA-ACF2 checks to perform validation. The pointer involves the NEXTKEY parameter.

Miscellaneous

The DATA parameter is a miscellaneous parameter. It contains information that eTrust CA-ACF2 does not use when validating access to a data set, unless a user exit checks that data.

Rule Entry Parameters

The parameters that make up the components of an access rule entry are as follows:

dsnmask
Specifies the name of the data set or a mask to which this entry applies. This parameter is required. For example, WORK.MASTER represents the name of the data set PAYROLL.WORK.MASTER. (The $KEY or the $PREFIX control statement specifies the high-level index PAYROLL.) You must specify the dsnmask parameter before any other parameters in a rule entry.

A data set name can have from 1 to 22 levels of qualifiers. Each level must begin with an alphabetic character or the character @, $, or #. You can specify up to eight characters per level. The entire data set name, including periods, can contain up to 44 characters.

Volume(*volmask*)
Specifies a mask for the specific volume or set of volumes where the data set must reside for this rule to apply. If omitted, all volumes are considered. If you specify VOLUME(), eTrust CA-ACF2 considers no volumes in its validation. Use VOLUME() when functions such as building a generation data group cannot provide a real volume serial number. When this happens, a volume serial number of all blanks is provided. You can use the asterisk (*) or dash (-) masking characters to specify the VOLUME name.

UId(*uidmask*)
Specifies a mask to indicate the users for whom this rule entry applies. If you do not specify a value or you specify UID(-), the entry applies to all users of the system. A UID mask can be defined by omitting ending characters by using asterisks **) or by a using a dash (-).

SOurce(*sourcemask*)
Specifies a logical or physical input source or source group name to which this rule applies. If omitted, any input source is valid. Contact your security administrator for a list of valid source group names. This field can be masked.

SHift(*shift*)
Specifies the name of the shift record to which this rule applies. The shift record defines the allowable days, dates, and times for access under this rule. This field cannot be masked.

Library(*libmask*)
Specifies a name that identifies a single library or set of libraries from which a program must execute for the access rule to apply. You can use the asterisk (*) or dash (-) masking characters to specify the library name.

If you do not enclose the name or mask in single quotes, eTrust CA-ACF2 prefixes that name or mask with the high-level index specified on the $KEY or the $PREFIX control statements. For example, if you specify LIB(PROGLIB) in the rule set with control statement $KEY(PAYROLL), the compiler assumes PAYROLL.PROGLIB as the library name.

If you specify the PGM parameter but omit the LIB parameter, eTrust CA-ACF2 tries to match on the program name. Also, specifying SYS1.LINKLIB covers all libraries in the GSO LINKLST record and link pack area (LPA).

PGM(*pgmmask*)|PRogram(*pgmmask*)
Specifies a mask defining the set of programs (in the set of libraries specified by the LIB keyword) to which this rule entry applies. If omitted, eTrust CA-ACF2 considers all programs to match the environment. You can use the asterisk (*) or dash (-) masking characters to specify the PGM name.

DDname(*ddnmask*)
Specifies a mask that identifies the specific ddnames that must be used for this rule to apply. If omitted, any ddname is permitted. You can use the asterisk (*) or dash (-) masking characters to specify the DDNAME name.

UNtil(*date*)
Specifies a Gregorian date in the form mm/dd/yy, yy/mm/dd, or dd/mm/yy, depending on a site option (see the DATE field of the GSO OPTS record), that is the last date on which this rule is considered valid. Years specified as 70-99 assume a date in the 20th century (1970-1999); years specified as 00-69 assume a date in the 21st century (2000-2069).

ACTIVE(*date*)
Specifies a Gregorian date in the form mm/dd/yy, yy/mm/dd, or dd/mm/yy, depending on a site option (see the DATE field of the GSO OPTS record), that is the first date on which this rule is considered valid. Years specified as 70-99 assume a date in the 20th century (1970-1999); years specified as 00-69 assume a date in the 21st century (2000-2069).

This parameter is valid only when the GSO RULEOPTS RULELONG parameter is set.

For(*days*)
Specifies the number of days, starting from the day the access rule set was compiled, for which this rule is considered valid. The minimum number that can be specified is zero (meaning today) and the maximum number is 365.
Note: The FOR date is converted to an UNTIL date. Each time the rule is recompiled the date will be extended to a date that may be beyond the original intended number of days.

DAta(*text*)
Specifies a one to 64-character comment that you want to store with the rule. eTrust CA-ACF2 stores the comment with the rule set and displays it when you decompile the rule. You might have standards concerning the format of this string. Values in it are not used by eTrust CA-ACF2 processing, but might be meaningful in your local implementation of eTrust CA-ACF2 (through special program exit checking, and so forth).

NEXTKEY(*nextkey*)
Specifies the key of an alternate rule set that should be checked if access to this data set is **denied** based on this rule entry. See the section on the use of the NEXTKEY parameter.

Read(Allow|Log|Prevent)
Specifies read access and the action A, L, or P that you want eTrust CA-ACF2 to take when the environment matches. Before this access permission applies, the actual access attempt must match the environment defined by other parameters of the access rule entry.

The letter codes are defined as:

- **A**—Allow the access
- **L**—Permit the access but log the event
- **P**—Prevent the access

If not specified in the rule entry, this access permission defaults to READ(P). ACF99900 messages are not issued for data set read accesses.

Write(Allow | Log | Prevent)
Specifies write access and the action (A, L, or P) that you want eTrust CA-ACF2 to take when the environment matches. This parameter works similarly to how the READ parameter does. If not specified in the rule entry, this access permission defaults to WRITE(P).

Allocate(Allow | Log | Prevent)
Specifies allocate access and the action (A, L, or P) that you want eTrust CA-ACF2 to take when the environment matches. This parameter specifies that a user has create, delete, rename, and catalog authority to a data set. If not specified in the rule entry, this access permission defaults to ALLOC(P).

Execute(Allow | Log | Prevent)
Specifies execute access and the action (A, L, or P) that you want eTrust CA-ACF2 to take when the environment matches. This parameter works similarly to how the READ parameter does. However, its access permission is the specified value or the value of the READ parameter—whichever designates the most permissive access. (For example, if READ(P) and EXEC(L) are specified, then EXEC(L) applies.) If READ access specifies A or L and EXEC access is not specified, it will default to the same access as READ. Also, if READ access is less restrictive than EXEC access (for example R(A) E(L)) then EXEC access will be set to the same access as READ.

Note: If TSO options ALTLIB ACT SYSTEM(YES), ALTLIB ACT USER(YES), ALTLIB ACT APPL(YES), or EXECUTIL SEARCHDD(YES) are in effect in a TSO session or batch TMP, the user needs READ access (rather than merely EXEC access) to libraries containing REXX execs invoked directly or indirectly during the session. This limitation is caused by the fact that REXX opens its libraries in a way that is indistinguishable from the open for an EXECIO for input. The limitation does not apply in a pure TSO CLIST environment.

Comment Statements

Comment statements, denoted by an asterisk (*) in column one, let you place any text inside the uncompiled rule set. The text on a comment statement is stripped off when the rule is compiled. Comment statements do not appear when you decompile the rule set.

Use caution when editing rules with comment statements. For example, if you moved the commented rule entry to the top of the rule, you would comment out everything but the $KEY entry because the dash signals that the line is to be continued.

```
$KEY(ABC)
* UID(PAYCLK1) SHIFT(ALL) R(A) W(A) DATA(Bill can update this data set) -
 UID(PAYMGR) R(A) W(P)
```

As you can see, the rule does not make much sense this way and eTrust CA-ACF2 cannot process it.

Masking Data Set Names and UIDs

You can use masks to represent multiple data set names or UIDs. This section describes how to use these types of masks in access rules.

Data Set Name Masks

Masking data set names (other than the name used in the $KEY high-level index field) lets an access rule entry apply to more than one data set. Consider the following PAYROLL rule set:

```
$KEY(payroll)
 work.- UID(tfinpay) R(A)
```

The data set name mask WORK.– lets the listed rule entry apply to all data sets with a high-level index of PAYROLL and a second-level index of WORK. Such data sets might include the following:

```
PAYROLL.WORK.TEST
PAYROLL.WORK.MASTER
PAYROLL.WORK.BACKUP.VER1
```

You can mask data set names using the dash (–) and asterisks (*) as follows. You **cannot** mask the $KEY statement.

Using the Dash

A data set name mask that contains a dash must fit one of the following cases:

- If the dash falls at the end of an incomplete data set name index, then the dash represents any number of characters that validly complete the index. (An index can be from one to eight characters.)

```
WORK.BA-  can represent      WORK.BA
                             WORK.BACKUP
                             WORK.BAK
          cannot represent   WORK.BACKUP.FILE
```

  Note: When you use a dash as the last character in a data set name mask, be sure there are also one or more parameters on that rule line. Any time that a dash appears as the last character on a rule line, eTrust CA-ACF2 interprets the dash as a continuation character.

- If the dash appears as a separate index in the mask, then the dash can represent any zero or more indexes until the next index of the data set name mask matches an index in the data set name.

```
WORK.-  can represent  WORK.TEST
                       WORK.TEST
                       WORK.TEST.VER1

-.TEST  can represent  WORK.TEST
                       WORK.VER1.TEST
```

  Note: When the index in the data set name mask after the dash index matches an index in the data set name, direct comparison of the data set name and the data set name mask resumes.

```
WORK.-.TEST*  can represent    WORK.VER1.TEST
                               WORK.VER1.TEST2
              cannot represent WORK.TEST1.TEST29
                               WORK.TEST1.TEST2
```

  In the second example, the WORK index in the data set name matches the WORK index in the data set name mask. The next index (TEST1) in the data set name matches the TEST* index in the data set name mask, and the dash index is considered to have represented zero (no) indexes in the data set name. The TEST2 index of the data set name then has no matching representation in the data set name mask.

- If a dash falls between or before any characters in an index, then the dash is literally a dash. For example, W–RK cannot represent WORK.

Using the Asterisk

A data set name mask that contains asterisks must fit one of the following cases:

- If the asterisks fall at the end of a partial data set name index, then the asterisks represent any number of characters from zero to the number of asterisks.

```
WORK.BACK**  can represent  WORK.BACK
                            WORK.BACKUP

             but not        WORK.BAC
                            WORK.BACLUP
                            WORK.BACKUPP
                            WORK.BACK.FILE
```

- If the asterisks form a separate index, then asterisks represent any index (of at least one character) whose length is no greater than the number of asterisks.

```
WORK.****  can represent  WORK.M
                          WORK.TST
                          WORK.BACK

           but not        WORK
                          WORK.BACKUP
                          WORK.M.MM
```

- If the asterisks fall between or before any characters of an index level, then each asterisk represents exactly one character.

```
WORK.**ST  can represent  WORK.TEST
                          WORK.LIST

           but not        WORK.ST
                          WORK.MASTER
                          WORK.TEST.M
```

Using an Asterisk Followed by a Dash

A data set name mask that contains an asterisk followed by a dash must fit one of the following cases:

- If the asterisk and dash fall at the end of a data set name index, then they represent any characters that validly complete the index (as does a dash alone).
- If the asterisk and dash form a separate index, then they represent exactly one index of at least one character.

```
WORK.*-  can represent  WORK.M
                        WORK.TEST

         but not        WORK
                        WORK.BCK.VER1
```

If the dash precedes any asterisks, then the dash is treated literally as a dash while the asterisks are treated as the asterisks of a mask.

UID Masks

A UID mask represents more than one UID and lets an access rule apply to multiple users. This is illustrated by the following UID:

```
UID(tfinpay)
```

This UID mask can represent any UID that begins with the letters TFINPAY and ends with up to 17 characters. (A valid UID can contain up to 24 total characters.)

A UID mask can be defined by omitting ending characters, by using asterisks (*), or by using a dash (–).

Omitting Ending Characters

A UID is automatically treated as a mask. For instance, the UID TFINPAYNLT not only matches itself, but also matches any string that begins with the characters TFINPAYNLT and contains no more than 24 characters.

By omitting characters, you can form a more general UID mask. For example, characters can be omitted from the UID TFINPAYNLT to form a mask that represents all users in the payroll department:

```
UID(tfinpay)
```

The mask matches any UID that begins with the characters TFINPAY and contains up to 24 total characters.

Using the Dash

A UID mask containing a dash must fit one of the following cases:

- If the dash falls at the end of a UID mask, it has the same effect as no dash. For example, the following two UID masks are equivalent:

  ```
  UID(tfinpay-)
  UID(tfinpay)
  ```

- If the dash is alone, then the UID represents all valid UIDs.

  ```
  UID(-)
  ```

If the dash falls in the UID mask, it is treated literally as a dash and cannot represent any other character.

Using the Asterisk

A UID mask that contains asterisks must fit one of the following cases:

- Asterisks that fall at the end of the UID mask have the same effect as a dash or as no asterisks. For example, the following three UIDs are equivalent:

```
UID(tfinpay-)
UID(tfinpay****)
UID(tfinpay)
```

- If the UID mask is comprised of all asterisks, then the UID mask represents all valid UIDs, regardless of the number of asterisks: UID(****).
- If the asterisks fall between or before any characters of a UID mask, then each asterisk represents exactly one character.

```
UID(TFIN***NLT)
```

  The mask TFIN***NLT matches any UID beginning with the letters TFIN, followed by any three characters except nulls, followed by the letters NLT, and then followed by any other characters to form a UID of up to 24 characters.

Using Blank Characters

A UID mask can contain blank characters. Blanks, whether they appear at the beginning of the UID, embedded in the UID, or at the end of the UID, are treated literally as blank characters.

For example, suppose that USER1 needs READ access to SYS1.DATASET, but USER12 does not. Store this rule, including the trailing blanks, to ensure that USER12 does not match the UID mask:

```
$KEY(SYS1)
 DATASET UID(*****USER1 ) R(A)
```

Use of NEXTKEY

The NEXTKEY parameter directs eTrust CA-ACF2 to evaluate an alternate access rule set when a particular environment applies to the access, but the access is prevented. eTrust CA-ACF2 only checks the NEXTKEY parameter when the access matches the environment, but the rule prevents the access. Validation of the access continues with the evaluation of the alternate access rule set.

To specify the index of the alternate rule set, use the NEXTKEY parameter. The NEXTKEY parameter in a rule entry lets you split a very large rule set into several sets or merge several rule sets together. You can also use NEXTKEY to delegate rule maintenance authority with the %CHANGE and %RCHANGE control statements.

Note: You can also use the GSO RULEOPTS RULELONG parameter instead of NEXTKEY in access rules. The RULELONG parameter specifies whether you want to use rules greater than 4K in length. See eTrust CA-ACF2 Rule Option Specifications (RULEOPTS) in the "Maintaining Global System Options Records" chapter for more information about RULELONG.

Merging Rule Sets

Similar data sets, such as production files, might require similar eTrust CA-ACF2 validation. The following rule sets provide an example of the use of NEXTKEY to merge multiple rule sets. Here are four rule sets that describe an access environment for the data sets with the high-level qualifiers ACCT01, ACCT02, ACCT03, and ACCT25: (Assume there are also rules for ACCT04 through ACCT24.)

```
$KEY(acct01)
 data.file UID(tfinpaynlt) NEXTKEY(acctxx)

$KEY(acct02)
 data.file UID(tfinpaynlt) NEXTKEY(acctxx)

$KEY(acct03)
 data.file UID(tfinpaynlt) NEXTKEY(acctxx)

$KEY(acct25)
 data.file NEXTKEY(acctxx)
```

Although none of these rule sets allow access to users who match the UID mask TFINPAYNLT, these users can be given read and write access to all of the previous accounting data sets, even though each data set has a different high-level index. Use the NEXTKEY parameter to direct eTrust CA-ACF2 evaluation to one main rule set, ACCTXX. That rule set might be written as follows:

```
$KEY(acctxx)
$PREFIX(acct**)
 data.file UID(tfinpaynlt) R(A) W(A)
```

The $PREFIX control statement must be contained in the NEXTKEY rule set to ensure that all high-level qualifiers of the rule sets directed to it (for example, ACCT01, ACCT02, and so forth) match the data set name patterns specified.

In this accounting file example, special access permission (such as ALLOCATE) to one particular data set can be specified using NEXTKEY in two ways.

1. The data set name, such as ACCT04.DATA.FILE, can be specified in a rule entry in the alternate ACCTXX rule set by enclosing the data set name in single quotes as follows:

   ```
   $KEY(acctxx)
   $PREFIX(acct**)
    'acct04.data.file' UID(tfinpaynlt) R(A) W(A) A(A)
     data.file UID(tfinpay) R(A)
   ```

   In the previous rule set, only user TFINPAYNLT has allocate access to ACCT04.DATA.FILE. All other payroll department users, including TFINPAYNLT, have read access to all accounting data sets.

2. Another method for specifying special permission to a particular data set is to place a rule entry in the original ACCT04 rule set to specify read, write, and allocate access for the user TFINPAYNLT, while retaining the NEXTKEY rule entry to govern all other access attempts:

   ```
   $KEY(acct04)
    data.file UID(tfinpaynlt) R(A) W(A) A(A)
    data.file NEXTKEY(acctxx)
   ```

Dividing Rule Sets

You can use the NEXTKEY feature to divide a particular high-level index rule set. You might divide a rule set if it is very large (and you have not enabled the RULELONG parameter of the RULEOPTS GSO record), or to delegate rule maintenance (%CHANGE or %RCHANGE) authority.

For example, a site can have numerous data sets all having the high-level index of PAYROLL:

```
PAYROLL.MASTER.SHOP1   PAYROLL.BACKUP.SHOP1
PAYROLL.MASTER.SHOP2   PAYROLL.BACKUP.SHOP2
PAYROLL.MASTER.SHOP3   PAYROLL.BACKUP.SHOP3
PAYROLL.MASTER.SHOP4   PAYROLL.BACKUP.SHOP4
PAYROLL.MASTER.SHOP5   PAYROLL.BACKUP.SHOP5
```

A single rule set for all of these data sets would be very large. You can divide the rule set for the high-level index PAYROLL into smaller rule sets with the NEXTKEY feature. Divide the rule set according to the second-level indexes:

```
$KEY(payroll)
 master.shop- NEXTKEY(master)
 backup.shop- NEXTKEY(backup)
```

In the previous rule set, the NEXTKEY parameter specifies the alternate rule sets to be used in validating access to the MASTER and BACKUP files. You can write two smaller rule sets as follows:

```
$KEY(master)                          $KEY(backup)
$PREFIX(payroll.master)               $PREFIX(payroll.backup)
%CHANGE tfinpaydir                    %RCHANGE tfinopsdir
 shop1 UID(tfinpayc1) R(A) W(A)        shop- UID(tfinopr) R(A)
 shop2 UID(tfinpayc2) R(A) W(A)
 shop3 UID(tfinpayc3) R(A) W(A)
 shop- UID(tfinpaymgr) R(A)
```

The previous rule sets are smaller than the single rule set for the high-level index PAYROLL. Also, in each of the previous rule sets, the $PREFIX control statement is specified so that the true high-level index is appended to each data set name.

In the first rule set, the payroll clerks (TFINPAYC1, TFINPAYC2, and so forth) in each of the site's shops need read and write access to only their particular shop's MASTER files to update the files. The payroll manager (MGR), however, is given read access to all shop MASTER files.

In the second rule set, the site's computer operators (TFINOPR) must have read access to only the BACKUP files for all shops.

We emphasize that eTrust CA-ACF2 validation is directed to the rule set specified in the NEXTKEY option only when access based on the current rule entry is prevented. You can build a maximum chain of 25 NEXTKEY options. If you specify more than 25, eTrust CA-ACF2 denies access. An SMF logging record is written to log the event. The NEXTKEY field of the ACFRPTDS report displays all the $KEYs of all the rules that were checked. KEYEXECS error condition indicates that too many NEXTKEYs were part of the search chain.

When using the NEXTKEY parameter, you must ensure that looping is avoided. In other words, a rule set containing a NEXTKEY parameter cannot be interpreted more than once during a single access validation. eTrust CA-ACF2 issues an error message if a loop condition occurs. In addition, eTrust CA-ACF2 denies the access request and logs the event. The NKEYLOOP error condition indicates that a loop occurred.

Note: An alternative to using NEXTKEY to divide a large rule set is to increase the size of the eTrust CA-ACF2 access Rule database. The default size of a eTrust CA-ACF2 access rule record is 4K (4096 bytes), but your site can increase the size up to 32K (32000 bytes). For more information, see the discussion of the RULEOPTS record in eTrust CA-ACF2 Rule Option Specifications (RULEOPTS) in the "Maintaining Global System Options Records" chapter. Also see the *Getting Started* guide for valuable information and considerations for increasing the size of the eTrust CA-ACF2 access Rule and Infostorage databases.

Delegating Change Authority

The NEXTKEY feature also permits %CHANGE and %RCHANGE authority to change a particular rule set.

As in the two previously shown rule sets, the director of payroll (TFINPAYDIR) is given authority to change the access rule set for the PAYROLL.MASTER files. This authority is delegated through the %CHANGE control statement. The director of operations (TFINOPSDIR) is given restricted authority to change only the rule entries for the PAYROLL.BACKUP files. This restricted authority is delegated through the %RCHANGE control statement.

eTrust CA-ACF2 Features that Simplify Access Rule Writing

The writing of access rules for all data sets residing on a system might seem to be a substantial undertaking at the time of installation. But keep in mind these three important features of eTrust CA-ACF2:

- Modes
- Centralization versus decentralization option
- Masking

These features are described in the following.

Modes

Modes let eTrust CA-ACF2 be integrated into your computer operating system in stages. These stages are designed to provide time for the development and testing of rules and various local eTrust CA-ACF2 features. For information about the modes of eTrust CA-ACF2 (QUIET, LOG, WARN, ABORT, and RULE), see the description of the MODE field of the GSO OPTS record in eTrust CA-ACF2 Option Specifications (OPTS) in the chapter entitled, “Maintaining Global System Options Records.”

Centralization versus Decentralization Option

Your site can centralize or decentralize the different aspects of eTrust CA-ACF2 administration. In a centralized environment, the security administrator generally has sole responsibility for the writing of access rules. Decentralization grants each user the authority to store an access rule set for data sets that the user owns. This centralization and decentralization feature is controlled through the GSO RULEOPTS record, described in eTrust CA-ACF2 Rule Option Specifications (RULEOPTS) in the "Maintaining Global System Options Records" chapter.

You can also use the %CHANGE and %RCHANGE control statements to specify which logonids can change a rule set. Another possible way to decentralize rule administration is to give a logonid the SECURITY privilege, but create a scope record that limits its access to a specific group of data set high-level indexes. For details on creating scope records, see the "Maintaining Scope Records" chapter.

Masking

Masking lets an access rule environment apply to a group of data sets and a group of users. Masking is most effective when you create your site's UID correctly. For details, see the *Implementation Planning Guide*.

Rule Selection Algorithm

The rule compiler converts the input into a form that the eTrust CA-ACF2 rule interpreter can verify. In addition, the rule compiler orders the rules according to the following criteria. However, when $NOSORT is specified, the rules remain in the exact order entered. We recommend that you do not specify $NOSORT.

1. DSN patterns from most specific to most general
2. VOL patterns from most specific to most general
3. UID patterns from most specific to most general
4. SOURCE operands in alphabetical order, with "not specified" last
5. SHIFT operands in alphabetical order, with "not specified" last
6. LIB patterns from most specific to most general
7. PGM/PROG patterns from most specific to most general
8. DDN patterns from most specific to most general
9. UNTIL dates from earliest to latest
10. ACTIVE dates from earliest to latest

The first rule entry that matches the actual data set, volume, UID source, shift, library, program, and date being used (that is, the defined **environment**) is the rule entry eTrust CA-ACF2 uses to determine the access privileges. If the compiler finds two conflicting rules with the same environment during the sorting process, it rejects the input rule set and terminates.

Creating Access Rule Sets

Unlike some of the other types of eTrust CA-ACF2 records, you must compile access rules to store them in the Rule database and decompile them to display them at your terminal. The compile process takes the human-readable text of the rule set and converts it into a machine-readable format. The decompile process converts the machine-readable code to human-readable text that you can view on your terminal.

You can create access rule sets directly from the terminal or by first building the rule set text in a file.

Compiling at the Terminal

To compile at the terminal, use the following procedure:

1. From TSO READY mode, issue the following commands:

```
acf
 ACF
set rule
 RULE
compile
 ACF70010 ACF COMPILER ENTERED
```

   The COMPILE subcommand lets you enter the control statements and rule entries at the terminal. Some general guidelines are:

   - Type the $KEY control statement first. Press ENTER.
   - Type each control statement on a separate line beginning in column two. Press ENTER.
   - Type each rule entry on a separate line, one line at a time. Press ENTER.
   - Press ENTER when you reach the end of the entry.

- If a rule entry is too long (that is, more than 72 characters), type a dash to indicate that the entry continues on the next line before you press ENTER. If you use the dash (–) masking character at the end of the data set name and no parameters appear on that rule line, eTrust CA-ACF2 interprets the dash as a continuation character. Be sure to include at least one parameter, such as a UID specification, in this situation.

```
acf
 ACF
set rule
 RULE
compile
 ACF70010 ACF COMPILER ENTERED

.$key(payroll)
.  work.master uid(tfinpaynlt) r(A) w(A) e(A)
.  work.backup uid(tfinpayiso) r(A) w(L) e(A)
.
store
```

2. Press ENTER twice or press END and press ENTER to signal the end of the rule.
3. Enter the TEST subcommand to make sure the rule works the way you want it to work. The TEST subcommand performs a dummy validation that does not create an SMF record. The output is what will actually happen if you store the rule. Test results should be consistent with the MODE you have specified.
4. Enter STORE to save the access rule set.
5. If the rule set is to be globally resident, then you must issue the following command to dynamically reload the new rule set:

   ```
   F ACF2,RELOAD(ruleid)
   ```

   Otherwise, the rule is not made resident until the next system IPL. For more information about resident rules, see the description of the GSO RESRULE record in the “Maintaining Global System Option Records” chapter.

Compiling from a Partitioned Data Set (PDS)

To compile from a PDS, use the following procedure:

1. Use an editor to create the text of a rule in a PDS.

   ```
   $KEY(payroll)
    work.master UID(tfinpaynlt) R(A) W(A) E(A)
    work.backup UID(tfinpayiso) R(A) W(L) E(A)
   ```

   Some general guidelines are:

   - Enter the $KEY control statement first. Press ENTER.
   - Enter each control statement on a separate line. Press ENTER.
   - Enter each rule entry on a separate line, one line at a time. Press ENTER.
   - If a rule entry is too long (that is, more than 72 characters), use a dash to indicate that the entry continues on the next line before you press ENTER. If you use the dash (–) masking character at the end of the data set name and no parameters appear on that rule line, eTrust CA-ACF2 interprets the dash as a continuation character. Be sure to include at least one parameter, such as a UID specification, in this situation.

2. Save this member.

3. Enter the following ACF subcommand:

   ```
   COMPILE dsn
   ```

   Where *dsn* is the name of the data set that contains the rule set text. eTrust CA-ACF2 stores input to the compiler from a PDS. For example,

   ```
   acf
    ACF
   set rule
    RULE
   compile work.text(rule)
   ```

   Specify the PDS name according to TSO conventions. Your high-level index is assumed unless you specify the entire PDS name and enclose it in single quotes as follows:

   ```
   'PAYNLT.WORK.TEXT(RULE)'
   ```

   If your high-level index was PAYNLT, then you could specify the following:

   ```
   WORK.TEXT(RULE)
   ```

   To compile all members of a PDS, specify WORK.TEXT ALL. This causes the members to be automatically stored.

4. Enter the TEST subcommand to make sure the rule works the way you want it to work. The TEST subcommand performs a dummy validation that does not create an SMF record. The output is what will actually happen if you store the rule. Test results should be consistent with the MODE you have specified.

5. If the rule set is to be globally resident, you must issue the following command to dynamically reload the new rule set:

   ```
   F ACF2,RELOAD(ruleid)
   ```

   Otherwise, the rule is not made resident until the next system IPL. For more information about resident rules, see the description of the GSO RESRULE record in the "Maintaining Global System Option Records" chapter.

Using Ditto for Entering Access Rules

A ditto mark (") provides a convenient method of using any of the parameters and values in the previous rule entry.

```
acf
 ACF
compile
 ACF70010 ACF COMPILER ENTERED

$key(payroll)
 .  work.master uid(tfinpay***) r(A)
 .  work.backup uid(") r(")
 .  work.test uid(") r(")
 .
store
```

In this sample rule set named PAYROLL, the first rule entry applies to the data set PAYROLL.WORK.MASTER. Read access to that data set is given to any user whose UID fits the mask TFINPAY***. The second rule entry contains ditto marks for the UID and READ parameters. Thus, read access to PAYROLL.WORK.BACKUP is given to any user whose UID fits the mask TFINPAY***. The third rule entry also has ditto marks for the UID and READ parameters. Thus, read access to PAYROLL.WORK.TEST is given to any user whose UID fits the mask TFINPAY***. The UID and READ parameters and any values are copied from the second rule entry.

The ditto feature eliminates the need for you to re-enter parameter values if they are the same as in the previous rule entry. You can use this feature to repeat the entire data set name and other parameter values. However, you cannot specify ditto for the parameter name itself (such as UID, READ, LIB, and so forth). You can specify ditto only for the parameter value. You can use the ditto feature for up to two data set name levels.

Using the ISPF Panels

Select option 1 RULES from the eTrust CA-ACF2 ISPF Option Selection Menu:

```
------------------ eTrust CA-ACF2 ISPF OPTION SELECTION MENU -----------------
OPTION  ===>

   1  RULES     - PROCESS eTrust CA-ACF2 ACCESS AND GENERALIZED RESOURCE RULES
   2  LOGONIDS  - eTrust CA-ACF2 LOGONID CREATION/MAINTENANCE FACILITY
   3  SYSTEM    - eTrust CA-ACF2 SHOW COMMANDS
   4  REPORTS   - eTrust CA-ACF2 REPORT PROGRAM PROCESSOR
   5  UTILITIES - PROCESS eTrust CA-ACF2 UTILITIES
   6  GSO       - GLOBAL SYSTEM OPTIONS SERVICES
   7  NET       - NETWORKING SYSTEM OPTIONS SERVICES
   8  CAC       - MVS DATABASE CACHE RECORD SERVICES
   9  XREF      - SOURCE/RESOURCE AND GROUP RECORD SERVICES
  10  MAC       - MANDATORY ACCESS CONTROL ADMINISTRATION
  11  CPF       - COMMAND PROPAGATION FACILITY SERVICES
  12  FIELD     - RECORD LEVEL PROTECTION CONTROLS
  13  TARGETS   - SET CPF TARGET NODES, DEFAULTS IN USE
  14  PROFILE   - PROCESS PROFILE INFORMATION RECORDS
  15  SMS       - PROCESS DFSMS SUPPORT RECORDS
  16  ENTRY     - PROCESS ENTRY SOURCE RECORDS
  17  SHIFT     - PROCESS SHIFT/ZONE RECORDS
  18  RACDCERT  - PROCESS KEYRING/CERTIFICATE COMMANDS
  19  C-CIC     - PROCESS C-CIC CICS INITIALIZATION RECORDS
  20  LDS       - PROCESS LDAP DIRECTORY SERVICES
```

The eTrust CA-ACF2 Rules Processor Menu is displayed.

```
------------- eTrust CA-ACF2 SECURITY RULES PROCESSOR MENU -------------------
OPTION  ===>

    1  ACFNRULE- NEW ACCESS RULE UTILITY

    2  ACFRULES- ACCESS/GENERALIZED RESOURCE RULE PROCESSOR

    3  ACFTEST - ACCESS RULE TEST FACILITY

    4  ACFTESTR- GENERALIZED RESOURCE RULE TEST FACILITY

    5  ACFDCMP - RULE DECOMPILER

    6  ACFNRSCR- NEW GENERALIZED RESOURCE RULE UTILITY

    7  ACFRULCU- RULE CLEANUP UTILITY

    8  ACCESS  - ACCESS COMMAND
```

Panel Field Descriptions

The options perform the following functions:

1 ACFNRULE
Lets you create or delete individual access rules. You can locate multiple access rules that contain the same character string, then verify and delete them.

2 ACFRULES
Lets you create, display, and delete access and resource rules. This chapter discusses only access rules.

3 ACFTEST
Lets you test access rules. Test results should be consistent with the MODE you have specified.

4 ACFTESTR
Lets you test resource rules. Test results should be consistent with the MODE you have specified.

5 ACFDCMP
Lets you decompile or view rules online or from a partitioned data set (PDS).

6 ACFNRSCR
Lets you create or delete individual resource rules. You can locate multiple resource rules that contain the same character string, then verify and delete them.

7 ACFRULCU
Lets you delete out-of-date rules from your rules database.

8 ACCESS
Lets you display a list of users and their associated access permissions for a given resource. It also lists any users that have been completely disallowed access to the resource.

Next select the task you want to perform from the eTrust CA-ACF2 Rules Processor Menu. The next sections describe the panels for each of these options.

Creating Access Rules with the ACFNRULE Utility

Select option 1 ACFNRULE to create an access rule using the ACFNRULE utility. The ACFNRULE utility creates a new access rule or deletes an existing rule only. You cannot use ACFNRULE to decompile (view) a rule.

```
------------------- ACFNRULE - eTrust CA-ACF2 NEW RULE UTILITY ----------------
COMMAND ===>

 eTrust CA-ACF2 ACCESS RULE KEY                      SHORT RECORD FORMAT
     eTrust CA-ACF2 RULE  ===>                   NORULELONG ===> (Y OR N)

 ADD OR DELETE    ===> ADD    ENTER: ADD OR DEL, DEFAULT = ADD
 DATA SET NAME OR MASK TO BE ADDED TO ACCESS RULE SET
   DATA SET NAME  ===>
 LOGONID OR UID OF THE USER THAT THE RULE IS TO BE WRITTEN FOR:
   LOGONID        ===>             (ALL)  UID  ===>

 ACCESS PERMISSIONS: ALLOW(A)/ALLOW & LOG(L)/PREVENT(P) DEFAULT=PREVENT
   READ  ===>      WRITE  ===>     ALLOCATE  ===>     EXECUTE  ===>

 ADDITIONAL PARAMETERS TO BE ASSOCIATED WITH THE RULE RECORD: (ALL OPTIONAL)
   VOLUME SERIAL  ===>            NEXTKEY        ===>
   LIBRARY NAME   ===>
   PROGRAM NAME   ===>            DDNAME         ===>
   INPUT SOURCE   ===>            SHIFT          ===>
   UNTIL DATE     ===>            FOR A NUMBER OF DAYS  ===>
   COMMENT DATA   ===>
   ACTIVE DATE    ===>
```

Panel Field Descriptions

The values for the fields are described in the following:

eTrust CA-ACF2 RULE
Specify the high-level qualifier of the data set. The default is your TSO prefix.

NORULELONG
Specifies to override the use of the rulelong compiler when RULELONG is active.

ADD OR DELETE
Specify ADD to create a rule, DEL to delete an existing rule.

DATA SET NAME
Specify the next qualifiers of the data set, up to 44 characters.

LOGONID
Specify a logonid or mask to which you want this record to apply. The default is all logonids. Before you enter a logonid or logonid mask, make sure your installation has defined the proper UID format to ACFNRULE. Place the logonid in quotes if trailing blanks are to be included in the formation of the UID that includes this field; otherwise, trailing blanks are excluded.

UID
Specify a UID or mask to which this record applies. The default is all UIDs.

READ
Specify A to allow read access, L to log read access, or P to prevent read access. P is the default.

WRITE
Specify A to allow write access, L to log write access, or P to prevent write access. P is the default.

ALLOCATE
Specify A to allow allocate access, L to log allocate access, or P to prevent allocate access. P is the default.

EXECUTE
Specify A to allow execute access, L to log execute access, or P to prevent execute access. P is the default.

VOLUME SERIAL
Specify the six-character volser on which this data set resides.

NEXTKEY
Specify the one to eight-character name of an alternate rule that eTrust CA-ACF2 must check when access is denied.

LIBRARY NAME
Specify the name of the library in which a program must reside for this access to be valid.

PROGRAM NAME
Specify the name of the program that must be used to access this data set.

DDNAME
Specify the ddname of the data set.

INPUT SOURCE
Specify the one to eight-character name of an entry record that eTrust CA-ACF2 must check to validate access.

SHIFT
Specify the one to eight-character name of a shift record that eTrust CA-ACF2 must check to validate access.

UNTIL DATE
Specify the date when this rule no longer applies. Use the format *mm/dd/yy* or the format specified in the DATE field of the GSO OPTS records.

FOR A NUMBER OF DAYS
Specify the number of days that this rule applies. The maximum value is 365.

ACTIVE DATE
Specify the first date when this rule applies. Use the format specified in the GSO OPTS record.

COMMENT DATA
Specify up to 64 characters of comments about this rule. This is stored in the $USERDATA portion of the access rule.

Creating, Displaying, and Deleting Rules

Select option 2 ACFRULES to create, view, or delete an access rule. The eTrust CA-ACF2 ISPF Full Rule Processor panel is displayed.

```
------------ eTrust CA-ACF2 SECURITY/ISPF FULL RULE PROCESSOR ------------------
COMMAND ===>

eTrust CA-ACF2 RULE SET NAME: (FIRST EIGHT CHARACTERS BECOMES MEMBER NAME)
   NAME    ===>
eTrust CA-ACF2 COMMAND MODE:
   MODE    ===> RULE           (RULE/RESOURCE MODE - DEFAULT IS RULE MODE)
   TYPE    ===>      RESOURCE TYPE(E.G., CKC,ITR,IAG)

DECOMPILE INTO OPTION OR PDS EDIT OF EXISTING RULESET IN A PDS:
   DECOMP RULE PRIOR TO EDIT ===> YES   YES OR NO
   TEST RULE PRIOR TO STORE  ===> YES   YES OR NO
ISPF LIBRARY DATA SET (DO NOT INCLUDE MEMBER NAME):
   PROJECT ===> LAGLO01
   LIBRARY ===> ACF2
   TYPE    ===> RULES

OTHER PARTITIONED DATA SET (DO NOT INCLUDE MEMBER NAME):
   DATA SET NAME  ===>

eTrust CA-ACF2 PROCESSING OPTIONS:
   CLEAR SESSION RULES  ===> NO    YES OR NO     NORULELONG===>   YES OR NO
   PURGE ABOVE RULE SET ===> NO    YES OR NO
   FORCE RULE REPLACE   ===> YES   YES OR NO
```

Panel Field Descriptions

The values for the fields are described in the following:

NAME
Enter the one to eight-character name of the high-level qualifier of the data set.

MODE
Specify nothing for access rules. RULE is the default.

TYPE
Specify nothing for access rules; this field only applies to resource and eTrust CA-ACF2 for DB2 rules.

`DECOMP RULE PRIOR TO EDIT`

Specify
YES to decompile the rule set before you are placed in edit mode. YES is the default.

`TEST RULE PRIOR TO STORE`

Specify YES to place you in the eTrust CA-ACF2 TEST subcommand before eTrust CA-ACF2 stores the rule. Test results should be consistent with the MODE you have specified.

`ISPF LIBRARY DATA SET (DO NOT INCLUDE MEMBER NAME):`

Enter the PROJECT, LIBRARY, and TYPE fields. For example, if the rule set is stored in a PDS named ACF2.RULES.PROD(PAYROLL), enter ACF2 for PROJECT, RULES for LIBRARY, and PROD for TYPE. You do not need to enter the member name. eTrust CA-ACF2 lets you select from a list.

`OTHER PARTITIONED DATA SET (DO NOT INCLUDE MEMBER NAME):`

Enter the fully qualified data set name in single quotes, for example: 'ADMIN01.ACF2.RULES'.

CLEAR SESSION RULES
Enter YES to clear rules that you have entered during this session. NO is the default.

NORULELONG
Enter YES to compile the rule set in a short record format. NO is the default.

PURGE ABOVE RULE SET
Enter YES to delete the rule set. No is the default.

FORCE RULE REPLACE
Enter YES to force replacement of the existing rule. YES is the default.

Testing Access Rules

Select option 3 ACFTEST to test an access rule. The ACFTEST - eTrust CA-ACF2 Access Rule Test Facility panel is displayed. Test results should be consistent with the MODE you have specified.

```
--------------- ACFTEST - eTrust CA-ACF2 ACCESS RULE TEST FACILITY -----------
COMMAND ===>

 RULEID NAME TO BE TESTED:        DECOMPILE PRIOR TO TEST:
   $KEY           ===>            DECOMP       ===> NO   YES/NO
 DATA SET NAME TO BE TESTED:
   DATA SET NAME  ===>
 LOGONID OR UID STRING OF THE USER AGAINST WHICH THE RULE WILL BE TESTED
   LOGONID        ===>            UID          ===>

 ADDITIONAL PARAMETERS TO BE ASSOCIATED WITH THE TEST
   TIME           ===>            VOLUME SERIAL  ===>
   LIBRARY NAME   ===>
   PROGRAM NAME   ===>            DDNAME         ===>
   INPUT SOURCE   ===>            DATE           ===>
   ACCESS         ===> READ       READ,WRITE,EXEC,ALLOC
   NOPREFIX       ===> NO         YES/NO

        ------------ ACFTEST OUTPUT DISPLAY AREA --------------------
```

Panel Field Descriptions

The values for the fields are described in the following:

$KEY
Enter the one to eight-character $KEY control statement of the rule that you want to test.

DECOMP
Specify YES to view the rule set before testing. NO is the default.

DATA SET NAME
Enter the name of a data set that you want test. eTrust CA-ACF2 places the $KEY value before the value you specify for DATASET NAME unless the value is in single quotes. If the rule contains a $PREFIX control card and the $PREFIX value is not masked, the $PREFIX value is used instead of the $KEY value.

LIBRARY NAME
Enter a library from which you want to test access. Any program through which the access is being made must reside in this library. eTrust CA-ACF2 places the $KEY value before the value you specify for LIBRARY NAME unless the value you specify is in single quotes. If the rule contains a $PREFIX control card and the $PREFIX value is not masked, the $PREFIX value is used instead of the $KEY value.

LOGONID
Enter the logonid of the user you want to test. If LOGONID is specified, the user making the request needs READ access to the eTrust CA-ACF2 LOGONID database as well as the appropriate access to the logonid record of the user whose access is being tested. You cannot mask the logonid being tested. Asterisks are treated as actual character values. Logonid fields such as SECURITY, NON-CNCL, PREFIX, and so forth, are taken into consideration when testing access.

NOPREFIX
When NOPREFIX is specified, the $KEY will be used instead of the $PREFIX when building the fully qualified dataset name or library name to be used in the test.

UID
Enter the UID for the user you want to test. ****** is the default.

TIME
Enter the time of day you want to test. Specify the time as *hh:mm*.

VOLUME SERIAL
Enter a six-character name of a volume or a mask that you want to test.

LIBRARY NAME
Enter the name of a library or a mask that a program must execute from that you want to test.

PROGRAM NAME
Enter the name of the program or a mask that you want to test.

DDNAME
Enter the ddname that you want to test.

INPUT SOURCE
Enter the name of a source, a source group, or a mask that you want to test.

DATE
Enter the date for which you want to test access. The format must be the same as specified in the GSO OPTS record.

ACCESS
Specifies the type of access that you want to test. Possible values for this field are READ, WRITE, EXECute, or ALLOCate. READ is the default.

Displaying Rules

Select option 5 ACFDCMP to decompile (view) a rule at the terminal or to place it in a partitioned data set (PDS). The eTrust CA-ACF2 ISPF Decompile Rule Processor panel is displayed.

```
------------- eTrust CA-ACF2 SECURITY/SPF DECOMPILE RULE PROCESSOR -------------
COMMAND ===>

 eTrust CA-ACF2 SECURITY COMMAND MODE:
   MODE    ===> RULE           RULE/RESOURCE MODE  (RULE MODE)
   DISPLAY ===>                VERBOSE/TERSE DISPLAY  (VERBOSE DISPLAY)
   TYPE    ===>                RESOURCE TYPE E.G., CKC, ITR, IAG

 eTrust CA-ACF2 SECURITY RULE SET NAME:
   $KEY    ===>
   LIKEKEY ===>
   DELETE  ===> NO   YES OR NO (NO)
   ALL     ===> NO   DECOMP ALL NEXTKEYS UNDER THIS $KEY (YES OR NO)
```

Panel Field Descriptions

The values for the fields are described in the following:

MODE
Specify nothing. RULE is the default.

DISPLAY
Specify nothing to view the entire rule. Specify TERSE to view only the user who stored the rule and the date when it was stored.

TYPE
Specify nothing; this field only applies to resource and eTrust CA-ACF2 for DB2 rules.

$KEY
Specify the one to eight-character high-level qualifier of the data set that you want to view.

LIKEKEY
Specify a mask to display a group of rules.

DELETE
Specify YES to erase the rules from the database after you view them. NO is the default.

ALL
Specifies YES to decomp the rule set and all NEXTKEY rule sets found under it. ALL and LIKEKEY are mutually exclusive keywords. NO is the default.

Using the ACF Command

You can process access rules by entering the TSO ACF command or by establishing the RULE setting of the ACF command:

```
acf
 ACF
set rule
 RULE
```

After you establish the ACF command setting, you can issue any of the following ACF subcommands:

- ACCESS
- CHKCERT
- *COMPILE
- CONNECT
- *DECOMP (or LIST)
- END
- EXPORT
- GENCERT
- GENREQ
- HELP
- REKEY
- RECKEY
- REMOVE
- ROLLOVER
- SET or T

- SHOW
- SN
- *STORE
- *TEST

The common subcommands ACCESS, END, CHKCERT, CONNECT, EXPORT, HELP, GENCERT, GENREQ, REKEY, REMOVE, ROLLOVER, SET, (or T), SHOW, and SN operate under all settings. The following text describes the function, syntax, and parameters of the other subcommands under the RULE and RESOURCE settings. The subcommands marked with an asterisk (*) work in the same manner under the ACF setting; however, if the rule ID ends with a dash (–) or a plus sign (+), enclose the rule ID in single quotes if it is an access rule. See "Overview of eTrust CA-ACF2" for a summary of these subcommands under that setting.

COMPILE Subcommand

The COMPILE subcommand lets you create a rule set. The syntax of this subcommand is as follows:

```
COMpile  [*]
         [dsn]
         [List|NOList]
         [Store|NOStore]
         [Force|NOForce]
         [Maxrule(250|nnn)]
         [NORulelong]
         [ALL]
```

eTrust CA-ACF2 provides two ways to compile rule sets:

1. Directly at the terminal.
2. From a partitioned data set (PDS).

These methods are discussed in Creating Access Rule Sets earlier in this chapter.

Parameter Descriptions

You can use the following parameters with the COMPILE subcommand:

*** (asterisk)**
Indicates that the text that follows is input to the compiler. This process is referred to as an online compile. In an online environment, the system prompts you to enter the access rule text directly from the terminal. In batch, the parameters following the COMPILE subcommand are assumed to be input.

(no parameters)
Indicates that the text that follows is input to the compiler. This is the same as specifying an asterisk.

dsn
Specifies a PDS and member name that contains the access rule text to be compiled. The PDS name follows TSO conventions. Your high-level index is assumed unless you specify the entire PDS name and enclose it in single quotes. For example, 'PAYNLT.WORK.TEXT(RULE)' would be specified.

If you do not specify a member name, eTrust CA-ACF2 prompts you for one. To compile input from all PDS members, specify the ALL parameter. When you specify ALL, eTrust CA-ACF2 does an automatic store. eTrust CA-ACF2 does not compile multiple rule sets from a single PDS member.

LIST|NOLIST
Causes the input to the compiler to be displayed on your screen or printed on your listing during compilation of a rule set. NOLIST causes no such display or printed list. LIST is the default. However, the LIST parameter of the compiler is ignored when compiling online. In the case of an online compile, NOLIST is always in effect.

STORE|NOSTORE
Determines whether the rule set is stored at compilation time. You must specify STORE when you compile a rule online. STORE is the default if you are using the ALL parameter to compile all members of a PDS; otherwise, NOSTORE is the default. NOSTORE means that you must issue the STORE subcommand to store the rule set.

FORCE|NOFORCE
FORCE stores the rule set regardless of whether it currently exists. NOFORCE stores the access rule set only if it does not already exist. FORCE|NOFORCE only apply when STORE is specified. FORCE is the default unless you are compiling selected members of a partitioned data set.

When used as parameters of the COMPILE subcommand, STORE, NOSTORE, FORCE, and NOFORCE apply only to the current compilation of this particular access rule set. When used as a parameter of the SET subcommand, FORCE or NOFORCE is in effect until it is changed or until you end ACF command processing.

MAXRULE(250|*nnn*)
Specifies a number from 0 through 999 that limits the size of the rule set that you can input to the compiler. Different rule entries require different amounts of space. The default is 250 rule entries. MAXRULE is applicable only if NORULELONG is specified in the GSO RULEOPTS record. If RULELONG is specified, MAXRULE is ignored.

The compiler obtains an input buffer large enough to contain the number of rule entries. Usually, the output rule set size is exceeded before the input limit of the MAXRULE is reached. However, sometimes there is not enough space to store the rule sets. This might occur when you attempt to compile large rule sets or rule entries that contain duplication or rule entries that are similarly duplicated (that is, parameters such as the same data set qualifiers, UIDs, program names, and so forth). If these situations occur, you can still compile the rule set by setting the MAXRULE value to a size greater than the default of 250.

NORULELONG

Overrides the use of the rulelong compiler when RULELONG is active. Normally, eTrust CA-ACF2 uses the rulelong compiler to compile rules if the RULELONG option is set. The rulelong format is an expanded record format. If a ruleset is small and therefore does not require the rulelong format, specifying NORULELONG on a compile lets you compile a ruleset using a compact record format. This way, you can choose to compile rules with the format that is required for the ruleset.

Note: If $NORULELONG was specified in the rule, this option is not necessary.

ALL

Compiles and stores the rule sets from all the members of a specified partitioned data set (PDS).

```
acf
 ACF
set rule
 RULE
compile work.text all
```

If any member of a PDS does not contain an access rule set, do not specify this parameter.

TEST Subcommand

The TEST subcommand lets you interactively test a compiled rule set. Testing determines whether the rule set provides the access you want. When you specify the TEST subcommand, eTrust CA-ACF2 performs a **full** validation of the compiled rule, but does not create any loggings for violations. Test results should be consistent with the MODE you have specified. User exits such as the SVCIXIT, DSNPRE, DSNPOST, DSNGEN, and VIOEXIT can affect the outcome of the test command for data set rule testing.

The syntax of the TEST subcommand is as follows:

```
TEst [*|ruleid]
```

Suppose you compile an access rule set with the $KEY(PAYROLL):

```
ACF70010 ACF COMPILER ENTERED

$key(payroll)
 work.backup uid(tfinpayiso) r(a) w(l) e(a)

ACF70051 TOTAL RECORD LENGTH= 162 BYTES, 3 PERCENT UTILIZED
```

You can issue the following TEST subcommand:

```
test *
 .
```

When the period (.) is displayed, the TEST subcommand is active. Enter any of the TEST subcommand keywords to specify the particular environment that you want to test. An asterisk entered in place of a parameter such as DSN(*) causes the previous parameter to be repeated.

For example, the following keywords test whether the access rule set PAYROLL lets the user TFINPAYNLT access the data set:

```
  dsn(work.master) uid(tfinpaynlt)
ACF71014 THE FOLLOWING PARAMETERS ARE IN EFFECT:
 DDN=******     UID=TFINPAYNLT
 DATE=11/01/04  SOURCE=********
 VOL=           DSN=PAYROLL.WORK.MASTER
 PGM=******     LIB=***.***
 TIME=*****     ACCESS=READ

  NO RULE APPLIES

  RESULT: DENY
  REASON: NORULE  =NO RULE IN RULE SET THAT MATCHES THE ENVIRONMENT

 end
```

After you enter the TEST subcommand keywords, the system displays all of the current values that describe the environment being tested. At the bottom of the display is an indication of whether the access is permitted, logged, or prevented.

In this example, the result is that this user is denied access.

After a result is displayed, you can make another entry of keywords and values to specify another environment for testing. The END subcommand terminates the TEST subcommand.

How the Values Describing the Environment Work

After you enter values that describe the test environment, these values remain in effect until they are specifically changed. Any values that you **do not** specify are assumed to be completely masked—the default. For instance, if you specify no UID keyword, the subcommand tests whether all UIDs are permitted access. However, the DSN parameter must be specified.

The TEST subcommand takes the following parameters:

*** (asterisk)**
Indicates that the last explicitly referenced rule set should be tested.

(no parameter)
Indicates that the last explicitly referenced rule set should be tested. This means that the TEST subcommand operates the same when you specify no parameters as when you specify an asterisk.

ruleid
Identifies the key of the rule set to be tested. To specify a rule set by its rule ID, you must have the authority to update the rule set, the AUDIT privilege level, or DECOMP authority as defined in the GSO RULEOPTS record. If the rule ID ends with a dash (–), you must enclose the rule ID in single quotes.

TEST Subcommand Keywords

After you issue the TEST subcommand with any of the parameters described previously, the TEST subcommand becomes active. You can specify a test access environment by entering any of the following keywords with the appropriate values. You must separate these keywords by blanks and you can specify them on one or more input lines.

Access(READ|WRITE|EXEC|ALLOC)
Specifies the type of data set access that you want to test. READ is the default.

Dsname
Specifies a data set for which access is tested. You must specify the DSN keyword as the first keyword. eTrust CA-ACF2 places the $KEY value before the value you specify for DSN unless the value for DSN is in single quotes. If the rule to be tested contains a $PREFIX control card, and the $PREFIX value is not masked, the $PREFIX value is used as the high level qualifier of the dataset name whenever the DSN keyword is entered without single quotes. The $KEY value will be used if the $PREFIX is masked. If the $KEY is not desired, then specify the fully qualified DSN in quotes.

LID(*logonid*)
Specifies a logonid whose access you want to test. To specify this keyword, the user making the request needs READ access to the eTrust CA-ACF2 LOGONID database as well as the appropriate access to the logonid record of the user whose access is being tested. You cannot mask the logonid being tested. Asterisks are treated as actual character values. Logonid fields such as SECURITY, NON-CNCL, PREFIX, and so forth, are taken into consideration when testing access.

LIBrary
Specifies a library from which you want to test access. Any program through which the access is being made must reside in this library. eTrust CA-ACF2 places the $KEY value before the value you specify for LIB unless the value for LIB is in single quotes. If the rule to be tested contains a $PREFIX control card, and the $PREFIX value is not masked, the $PREFIX value is used as the high level qualifier of the library name whenever the LIB keyword is entered without single quotes. The $KEY value will be used if the $PREFIX is masked. If the $KEY is not desired, then specify the fully qualified library name in quotes.

NOPREFIX
When NOPREFIX is specified, the $KEY value will be used instead of the $PREFIX value when building the fully qualified dataset name and/or library name to be used in the test.

Volume(*volsermask*)
Specifies the serial numbers (volser) of the volumes where the data sets reside for which you want access tested. You can use masking. If no volume is specified, the field defaults to blanks. If no volser is specified for the test rule, validation occurs regardless of RESVOL or SECVOL specifications.

UID(*uid*)
Specifies a user or group of users whose access you want to test. Asterisks are treated as actual character values. TEST interprets unspecified fields in the UID as blanks. So UID(*) and UID(*) are considered identical, for example.

To specify this keyword, the user making the test does not need access to the logonid record of the user whose access is being tested. Logonid fields such as SECURITY, NON-CNCL, PREFIX, and so forth, are not taken into consideration when testing access. If you specify LID and UID, the last LID or UID value specified is used. For example, if you specify LID(PAYJJD) UID(TFINPAYNLT), eTrust CA-ACF2 uses UID(TFINPAYNLT). Conversely, if you specify UID(TFINPAYNLT) LID(PAYJJD), eTrust CA-ACF2 uses LID(PAYJJD).

LIBrary(*libmask*)
Specifies a library from which you want to test access. Any program through which the access is being made must reside in this library. You can use masking.

PGM(*pgmmask*)|PRogram(*pgmmask*)
Specifies the name of the program for which access is tested. The keyword PROGRAM can be used instead of PGM. You can use masking characters.

DDname(*ddname*)
Specifies the ddname associated with the data set for which access is tested.

DAte(*date*)
Specifies the date you want to test the access. This date can be in the format *mm/dd/yy*, *yy/mm/dd*, or *dd/mm/yy*, depending on a system option. The TEST subcommand uses the current date as the default.

SOurce(*sourcemask*)
Specifies the source or source group from which access is tested.

Time(*hhmm*)
Specifies the time (in hours and minutes) when you want to test the access.

ENd
Terminates the TEST subcommand and places you back in the previous setting.

Note: The SHIFT keyword is not available when testing access rules; use DATE or TIME instead.

TEST Subcommand Results

The results of the TEST subcommand show the read, write, allocate, and execute authority for the tested access rule set or the read, add, update, and delete authority for the data set. The following example shows what eTrust CA-ACF2 displays when access to a data set is denied.

```
test *
 .  dsn('t4') uid(pay01) source(mail)
ACF71014 THE FOLLOWING PARAMETERS ARE IN EFFECT:
  DDN=******     UID=pay01
  DATE=11/08/04  SOURCE=MAIL
  VOL=           DSN=T4
  PGM=******     LIB=***.***
  TIME=*****     ACCESS=READ

  NO RULE APPLIES

  RESULT: DENY
  REASON: NOMODEAB=NO $MODE CARD SO OPTS FIELD INDICATED ABORT MODE
```

If a test access attempt is prevented but matches a NEXTKEY environment defined in the rule set, eTrust CA-ACF2 displays the rule ID for the next rule set to be interpreted.

At this point, validation of the test access attempt ends. eTrust CA-ACF2 prompts for TEST subcommand keywords and values for the next test access attempt. The END subcommand terminates the TEST subcommand.

STORE Subcommand

The STORE subcommand lets you store the previously compiled rule set. The syntax of the STORE subcommand follows:

```
STore
```

This subcommand accepts only the FORCE|NOFORCE parameter.

If SET NOFORCE has been previously issued, the rule set is stored only if it does not already exist. You receive a message if an access rule is not stored. The NOFORCE parameter of the SET subcommand is described in "Overview of eTrust CA-ACF2."

The store operation can be rejected because of insufficient authority for storing access rule sets.

DECOMP Subcommand

The DECOMP subcommand displays a rule set that has been previously compiled and stored. Use this subcommand to examine, update, or change rule sets. You can decompile a rule set at the terminal or into a member of a partitioned data set (PDS). You can also use the LIST subcommand to accomplish the same function as DECOMP.

The syntax of the DECOMP subcommand is:

```
DEComp or List  {*|ruleid|Like(ruleidmask)}
                [Into(dsn)][ALL]
```

Parameter Descriptions

The DECOMP subcommand takes the following parameters. You must specify one of the following parameters with the DECOMP subcommand.

*** (asterisk)**
Indicates you want to decompile the last explicitly referenced rule set that you processed since establishing the RULE or ACF setting.

ruleid
Specifies the $KEY of the rule set you want to decompile or list. If the rule ID ends with a dash (–), enclose the rule ID in single quotes.

Like(*ruleidmask*)
Specifies a group of rule sets that you want to decompile or list.

The following parameter is optional with the DECOMP subcommand:

Into(*dsn*)

Specifies the data set where you want the rule set to be decompiled. This data set must be a PDS. No other types qualify. When you specify a fully qualified data set name (that is, including the high-level index), you must enclose that name in single quotes. For example:

```
decomp paykln into('payjsd.work.rule')
```

When you decompile rule IDs that have an imbedded blank, you must enclose the rule ID in single quotes as follows:

```
decomp 'pay 777'
```

If a data set is not allocated, eTrust CA-ACF2 allocates that data set as a PDS and decompiles the rules into the PDS. If you specify no member name, eTrust CA-ACF2 uses the $KEY and $MEMBER values of the rule set to determine the member name. If the rule set has a $MEMBER control statement, the $MEMBER value is used as the member name. If the rule set has no $MEMBER control statement, the $KEY of the rule set is used as the member name. If the $KEY value is invalid as a member name, eTrust CA-ACF2 automatically generates a member name. For more information about the automatic generation of this member name, see the description of the SET MEMBER subcommand in "Overview of eTrust CA-ACF2."

ALL

Specifies that the rule set and all NEXTKEY rule sets found under it is decompiled. When the INTO keyword is used with the ALL keyword, all of the rule sets are decompiled into a single member. You cannot specify the ALL and LIKE keywords on the same decomp statement.

The output from the DECOMP with the ALL parameter into a data set cannot be feed back into a COMPILE from the same data set if multiple $KEY statements are present.

DELETE Subcommand

The DELETE subcommand lets you remove rule sets from the Rule database. Only users with the SECURITY privilege level can delete rule sets. You can restrict this authority through the use of scopes.

The syntax of the DELETE subcommand is:

```
DELete    {*|ruleid|Rule(ruleid)}
```

For example, the following subcommand deletes the access rule set PAY7777:

```
delete pay7777
 DELETED
```

After you issue the DELETE command, the message DELETED is returned.

Parameter Descriptions

You must specify one of the following parameters with the DELETE subcommand:

*** (asterisk)**
Indicates that you want to delete the last explicitly referenced rule set.

ruleid
Specifies a rule ID that you want to delete. If the rule ID ends with a dash (–), enclose the rule ID in single quotes.

Rule(*ruleid*)
Specifies an individual rule set that you want to delete.

RECKEY Subcommand

The RECKEY subcommand assists security administrators in maintaining rule sets and compiled infostorage rule records. This subcommand lets the user decompile, add or delete a rule entry, recompile, and store the updated rule set on one command. This command can be used in any ACF mode that handles compiled records and it executes on other CPF-defined nodes.

The syntax of the RECKEY subcommand is:

```
RECKEY {ruleid {ADD(rule-entry)|DELETE(rule-entry)}
```

For example, the following sequence shows how to add one additional rule line to an access rule using the RECKEY subcommand:

```
ACF
SET RULE
RECKEY SYS1 ADD(TEST.- UID(SYSTEST*) READ(A) EXEC(A) WRITE(A))
```

The RECKEY subcommand takes the following parameters:

ruleid
Specifies the key of the rule set being modified.

ADD|DELETE
Specifies the function to be performed. ADD inserts a rule line or entry. DELETE removes a rule line or entry.

rule-entry
Indicates the actual rule line or entry to be inserted or removed.

Sample Access Rule Sets

We can look at some examples using Jane Doe's payroll department. Assume that the payroll information, including salary rates and other confidential information, is contained in the data set PAYROLL.MASTER.DATA. Jane wants to grant only herself and her lead payroll clerk (PAY1234) access to this data set. Additionally, the clerk is permitted only to read the data, not update it, and for 30 days only.

Jane's rule set might look like this:

```
$KEY(payroll)
 master.data UID(finpay7777) READ(A) WRITE(A) ALLOC(A)
 master.data UID(finpay1234) READ(A) FOR(30)
```

The $KEY indicates that the high-level index of the data set is PAYROLL. The first field in each rule entry is the remainder of the data set name (MASTER.DATA). The UID fields specify the users being given access authority and their **allowable** types of access, thus the **A** after each READ, WRITE, and ALLOC for Jane, and only READ for her clerk. The FOR parameter indicates the clerk can have access for only 30 days.

Perhaps Jane keeps a current project list online for her department in the data set PAY7777.CURRENT.PROJECTS. She wants to let everyone in the Financial department read this data set. To make her job easier, she has authorized her lead payroll clerk (PAY1234) to update the rule set that governs her own data sets (which she indexes using her logonid PAY7777). Here is a sample of how Jane's rule set appears:

```
$KEY(pay7777)
%CHANGE finpay1234
 current.projects UID(fin) READ(A)
```

In the previous examples, access **not allowed** (such as WRITE) might have been entered as WRITE(P) to prevent write access. However, P is the default, so it is not necessary to enter it in the rule.

The sections that follow present a variety of sample access and resource rules.

TSO CLIST Considerations

You can use eTrust CA-ACF2 data set access rules and the control statement in the CLISTs to protect TSO CLISTs (Command Lists). In an access rule entry, specify the CLIST library name to be accessed as the DSN parameter. This rule entry grants specific access to CLISTs residing on that library. For example, to provide execute-only access to CLISTs residing in library TEST.CMD.CLIST, a sample access rule is:

```
$KEY(test)
cmd.clist UID(abc) EXEC(A)
```

If a user specifies a CONTROL LIST instruction in the CLIST, the contents of the CLIST are listed during execution even if a user has execute-only access. To prevent this situation from occurring, you should insert the NOLIST and NOCONLIST operands in sensitive CLISTS.

Also, when the CLIST is executed explicitly, TSO dynamically allocates the ddname SYS00nn. When the CLIST is executed implicitly, the ddname SYSPROC is used.

ISPF/PDF Skeletons, Messages, and Panels

To secure these resources use standard data set rules. A user will need READ(A) access to use these libraries. The following rule will allow users to access but not change the panels, skeletons, or messages:

```
$KEY(test)
ispf.ispplib UID(abc) READ(A)
ispf.ispslib UID(abc) READ(A)
ispf.ispmlib UID(abc) READ(A)
```

TSO REXX Considerations

Use standard data set rules to control these resources. The following rule will allow users access to the REXX script library:

```
$KEY(test)
rexx.ispexec UID(abc) READ(A)EXEC(A)
```

Note: Because of how the REXX interpreter opens files users will need both READ(A) and EXEC(A) privileges to execute scripts.

Generation Data Groups

Access to generation data groups (GDGs) can be provided by use of masking in the data set name in the rule. For example, to grant access to all generations of a payroll data set called PAYROLL.HOURS, the rule set might be:

```
$KEY(payroll)
 hours.- UID(pay) READ(A)
```

If you want to define a rule for only one specific generation, specify the full data set name, such as:

```
$KEY(payroll)
 hours.g0015v00 UID(pay) READ(A)
```

Secured Volume Access Rules

You can protect data sets at the volume level using the GSO SECVOLS record. In that record, specify the volume serial numbers (volsers) for the data sets that you want to protect.

After a volume has been specified in the SECVOLS record, you can establish:

- An access rule set for each secured volume.
- One access rule set for all secured volumes. An example of each method of establishing these access rules is given in the following.

Protection at the volume level does not exist for a given volume until that volume is specified on the secured volume list.

Validation of Secured Volume Accesses

To validate a data set access request to a secured volume, eTrust CA-ACF2 generates a pseudo data set name. This name has a format of @*volser*.VOLUME or VOLUME.@*volser*. In this format, *volser* represents the volume serial number, and VOLUME is literally the word VOLUME.

Sample Access Rule Set for a Single Secured Volume

You can establish an access rule set for each secured volume. To validate an access request to a volume with the serial number TEST01, eTrust CA-ACF2 generates a pseudo data set name of @TEST01.VOLUME to evaluate a rule set such as:

```
$KEY(@test01)
 volume UID(tfinpayjsd) R(A) W(A)
 volume UID(tfinactklw) R(A)
```

The access rule set key is the volume serial number, and each access rule entry begins with the word VOLUME.

Sample Access Rule Set for All Secured Volumes

Alternatively, you can establish one access rule set for all secured volumes. For a volume with the serial number TEST01, eTrust CA-ACF2 generates a pseudo data set name of VOLUME.@TEST01 to evaluate a rule set, such as:

```
$KEY(volume)
 @test01 UID(tfinpayjsd) R(A) W(A)
 @test01 UID(tfinactklw) R(A)
 @test02 UID(tfinact) R(A)
 @test03 UID(tfinfincrh) R(A) W(A)
```

In this case, VOLUME is the key of the access rule set, and a volume serial number (or volume serial number mask) begins each access rule.

For more information about the control of secured volume accesses, see the chapter entitled, "Maintaining Global System Options Records." To specify the list of secured volumes, see the description of the GSO SECVOLS record. To specify whether you are using one or more than one access rule set for secured volumes, see the description of the VOLRULE field of the GSO RULEOPTS record.

Tape Volumes

eTrust CA-ACF2 provides the option of data set protection on tape so that protection is equivalent for tape, disk, and mass storage systems (MSS). Under the option, eTrust CA-ACF2 considers the data set name specified in the JCL as valid although z/OS later verifies only the last 17 characters. If the site has installed a tape management system that validates the full data set name specified in the JCL as matching its catalog information, then full data set protection on tape can be accomplished without the use of eTrust CA-ACF2 exits or local coding. A tape management system that falls into this category is BrightStor CA-1.

If the tape management system does not retain information describing each data set on tape, the site must use the eTrust CA-ACF2 Pseudo Data Set Name Generator exit to interface with the tape management system and determine whether access should be allowed or whether the supplied data set name should be validated.

As a word of caution, tapes can be written by using EXCP (Execute Channel Program) access even though a tape data set was opened for input only. The only true protection is to manually remove the write ring. Thus, eTrust CA-ACF2 provides the installation exits with the IBM Ring IN/OUT status indication, which could be used for additional security checking. Furthermore, once access has been gained to a tape data set, any data set residing on that tape volume can be accessed. Therefore, data sets with differing security requirements should not be stored on the same volume.

If you use BrightStor CA-1 5.1 or above, see the security section of the BrightStor CA-1 documentation for procedures involved in tape security.

VSAM Allocation Information

VSAM-generated data set names and related VSAM data spaces might require some special information for allocation processing. For this reason, we recommend that you name both the data and index portions of the VSAM cluster with additional data set name qualifiers of DATA and INDEX, respectively.

```
cluster-name:      A.B.C
data-name:         A.B.C.DATA
index-name:        A.B.C.INDEX
```

For read or write access of a VSAM cluster, authorization is required only for the cluster name. Normal data set name access rules are therefore sufficient for normal user processing. For allocate access, authorization is also required for the data and index names.

In the following example, the user X can read and write to the cluster, and user Y can define, delete, and alter the cluster:

```
$KEY(a)
 b.c UID(x) READ(A) WRITE(A)
 b.c.- UID(y) ALLOC(A)
```

Protection of VSAM Data Spaces

Names are not given to VSAM data spaces. Therefore, when a VSAM data space or page space is created or deleted, eTrust CA-ACF2 validates access at the volume level.

For data spaces, VSAM generates an internal name such as Z9999999*x*. eTrust CA-ACF2 recognizes the VSAM naming convention and generates a pseudo data set name in the format *@volser*.VOLUME or VOLUME.*@volser*. Then, eTrust CA-ACF2 searches for a volume rule that allows access.

For example, the following access rule sets enables the user PAYSDH to define or delete VSAM data spaces on the volume VSAM01:

```
$KEY(@vsam01)
 volume UID(tfinpaysdh) ALLOC(A)
```

Or

```
$KEY(volume)
 @vsam01 UID(tfinpaysdh) ALLOC(A)
```

For more information, see the previous section on secured volume access rules.

VTOC Rules

If the VTOC (volume table of contents) is used as a data set (referenced as a data set and opened for processing), eTrust CA-ACF2 generates a pseudo data set name of SYSVTOC.*volser* to validate VTOC requests where the first character of the volser is alphabetic. If the first character is numeric, the eTrust CA-ACF2-generated pseudo data set name is SYSVTOC.@*volser*. eTrust CA-ACF2 then searches the Rule database for a SYSVTOC rule. A sample VTOC rule set is as follows:

```
$KEY(sysvtoc)
 work01 UID(abcd) R(A) W(A)
```

Catalog Processing

Special considerations exist for access to z/OS catalogs.

Implicit access to a catalog does not require a rule allowing a user access to that cataloged data set. An example of an implicit access is a situation in which a user allocates a new data set under his or her high-level qualifier. The catalog is updated to hold information about that new data set, but the user is not validated for a catalog update since the user was not directly updating the catalog.

On the other hand, a user who directly updates the catalog, usually when using catalog-related system utilities, requires authority to access the catalog data set.

When using the DEFINE and DELETE ALIAS commands, rules for the alias name and the related user catalog must allow access. In most cases, a rule allowing access to the alias name is not needed after the alias is defined. For this reason, a rule for the alias name is not normally written, and the user using the ALIAS command has an overriding privilege, such as NON-CNCL.

To avoid use of overriding privileges, eTrust CA-ACF2 exploits the catalog FACILITY authorization class for alias processing. When a rule in the SAF FACILITY class allows access to the following resource, the validation of the alias rules is bypassed:

```
$KEY(STGADMIN.IGG.DEFDEL.UALIAS) TYPE(FAC)
```

Users who are authorized to access the FACILTY resource are able to define and delete alias entries from the catalog without specific rules for the alias or any special logonid privileges. Write access is required when you define an ALIAS not when you delete an ALIAS.

Program Pathing Control

Program pathing control lets you control access to data by explicitly defining the program path that is permitted access to the data. It ensures that a data set can only be accessed by a certain program residing in a certain library. The program (PGM) and library (LIB) constitute the controlled program path for the access.

To use the program pathing feature, you must use the LIB and PGM rule entry parameters in the rule entry that controls access to the data. eTrust CA-ACF2 recognizes that program pathing is being used when the LIB and PGM access rule parameters are present in the applicable access rule entry. For example, in the following rule entry, access to the data set PAYROLL.PROD.MASTER has a defined program path:

```
$KEY(payroll)
 prod.master UID(*) LIB('prod.payroll.load') PGM(prodmst) R(A) W(A)
```

This rule entry states that any user can read and write to the data set PAYROLL.PROD.MASTER if the program PRODMST fetched from the library PROD.PAYROLL.LOAD is the actual program and library combination being used. If another program and library are being used, then the rule entry does not match.

The Active Library List

When eTrust CA-ACF2 validates access to a program-pathed data set, it takes special measures to ensure that only the defined library and program combination are used to gain access to the data set. To ensure this level of integrity, eTrust CA-ACF2 maintains a list of *active libraries*. The active library list names all the libraries from which the executing program can fetch another program or subroutine.

A program name, as used in the PGM parameter of an access rule, is a load module whose name is derived from the JCL used to execute this job step for a batch job. In the program pathing situation, eTrust CA-ACF2 must take into account that a particular routine might have been entered through LINK/XCTL/LOAD or ATTACH and not through a simple external subroutine link-edited with the load module. When another routine is entered through one of these macros, the program name used for data set access validation remains the name specified in the JCL.

Whenever a program is fetched, eTrust CA-ACF2 updates the active library list to include the name of the program and the library from which the program was fetched. At the time that a data set is actually opened, the routine in control might not be the original JCL indicated CSECT. Therefore, eTrust CA-ACF2 validates the access using the program name found in the JCL with all the libraries found on the active library list that have actually been fetched. Exceptions are made for the system linklist and the GSO LINKLIST. Details on these exceptions are documented later in The System Linklist and the GSO LINKLIST section.

To help you define a program pathing environment, eTrust CA-ACF2 provides the Data Set/Program Access Report (ACFRPTDS). This report shows the types of access being made and accurately reports the program and library eTrust CA-ACF2 believes to have made the access. Using ACFRPTDS output, sites should be able to adjust their system linklist, the GSO LINKLIST record, and rules to establish the desired program pathing environment for the desired data set.

A special program pathing trace is availabe to further help you in determining a program pathing environment. When PP-TRC (traces al accesses) or PP-TRCV (traces violations only) are on in a logonid record, batch jobs run under that logonid are traced and SMF records containing the Active Library List are cut. These trace records can be viewed using the Dataset/Program Access Report (ACFRPTDS). For further information, see the *Reports and Utilities Guide*.

The operating system provides many methods of defining libraries from which programs can be fetched. You must understand how the operating system searches program libraries because this process affects program execution. In fact, you can use these methods to alter the expected execution path of almost any program.

Special Considerations for Open of STEPLIB and JOBLIB Data Sets

A STEPLIB or JOBLIB data set is opened by the initiator program IEFSD060. However, an eTrust CA-ACF2 convention is to use the jobstep program from the PGM= on the EXEC statement as the program to be used for program pathing purposes when the OPEN of a STEPLIB or JOBLIB is being validated. It is not possible to verify that the job step program really exists in the STEPLIB or JOBLIB data set at the time of the OPEN validation because the library is not yet open.

Searching for a Program Module in z/OS

The methods z/OS uses to fetch a program are summarized in the following. Also see the IBM publication, *MVS/XA Supervisor Services and Macros*.

Search the Job Pack Area (JPA)

There is an area in each address space called the Job Pack Area (JPA). A fetched program can be loaded into the JPA for later execution. The JPA is searched before private libraries, task libraries, JOBLIBs, STEPLIBs, LPA, or the system linklist. Thus, a program could be put in the JPA and subsequent calls for it could cause the JPA copy to be executed.

Search private libraries

Private libraries can be specified in the assembler language instructions ATTACH, LINK, LOAD, and XCTL. When used, private libraries are added to the active library list.

Search task libraries

A program can attach what is called a *daughter task*. This is usually done using the ATTACH assembler language macro. By using ATTACH, programs can be fetched from other libraries that might or might not be defined in the JCL. That is, when a program fetch is done, the task libraries are searched before STEPLIBs, JOBLIBs, LPA, or the system linklist.

Search JOBLIBs or STEPLIBs

When z/OS encounters a JOBLIB or STEPLIB definition in the JCL, the named library is searched for the program to be executed. If a concatenated JOBLIB or STEPLIB definition is used, the specified libraries are searched in sequential order from first to last to locate the program. Up to 16 libraries can be concatenated in a single list.

Search Link Pack Area (LPA)

The system LPA is a range of virtual storage set aside at system IPL time. It usually contains Supervisor Calls (SVCs), access method routines, operating system programs, and user-written modules. Programs that reside in the LPA can be used by multiple users simultaneously.

Note: For LPA programs, eTrust CA-ACF2 uses SYS1.LINKLIB as the library for validation.

Search the System Linklist

The system linklist is a set of default program libraries defined in SYS1.PARMLIB through LNKLSTxx members. The linklist libraries become the default search libraries. Also, the system linklist is searched if the program being fetched is not found in the Job Pack Area (JPA), in a private library, in a task library, in a JOBLIB or STEPLIB library, or in the LPA.

Special Program Pathing Considerations When Using the VLF Extension to LLA

If a program other than the first program (which is derived from the JCL used to execute the job step for a batch job), is fetched from the VLF data space, the library that the program was originally loaded from will not be added to the Active Library list and will not be included in the program pathed validation.

The System Linklist and the GSO LINKLIST

The z/OS system linklist is a set of default program libraries defined in SYS1.PARMLIB through LNKLST*xx* members. A typical LNKLST*xx* member might look like:

```
SYS1.LINKLIB,
SYS1.CMDLIB,
SYS1.SORTLIB,
SYS1.VTAMLIB,
SYS2.PRODLIB,
OPRSYS.TSOLIB
```

In the previous example, all the libraries shown become the default search libraries. That is, a program is fetched from a linklist library if it is not found in the Job Pack Area (JPA), a private library, a task library, a JOBLIB, a STEPLIB, or LPA.

The linklist search order is from top to bottom. Using the previous sample LNKLSTxx display, SYS1.LINKLIB is checked first, then SYS1.CMDLIB, and so forth.

GSO LINKLST Record

eTrust CA-ACF2 provides a GSO record called LINKLST. This record names the libraries that the site wants eTrust CA-ACF2 to consider the same as SYS1.LINKLIB. When eTrust CA-ACF2 performs rule validation, LIB('SYS1.LINKLIB') is substituted for the true library name for all GSO LINKLST entries. For example, a GSO LINKLST record might contain the following entries:

```
SYS1.LINKLIB
SYS1.SORTLIB
SYS1.VTAMLIB
SYS2.PRODLIB
TCAM.LOAD
```

When a program fetched from SYS2.PRODLIB opens a data set, SYS1.LINKLIB is used for the library name for data set access validation.

Linklist and GSO LINKLST Program Pathing Information

eTrust CA-ACF2 performs the following special processing for programs that reside in a linklist library or are specified in the GSO LINKLST record:

- When the active library list consists only of programs fetched from a system linklist library, eTrust CA-ACF2 rule validation uses the true name of the library from which the program was fetched. For example, assume the following JCL was submitted and there were no libraries specified in the GSO LINKLST record:

  ```
  //payprod   JOB  ............
  //MSTREAD   EXEC PGM=prodmst
  //MSTFILE   DD DSN=payroll.prod.master,DISP=SHR
  ```

  The system linklist is searched for the program PRODMST. Assuming the program is found in library SYS2.PRODLIB, and when access to data set PAYROLL.PROD.MASTER is validated, the access environment is as follows:

  ```
  DSN='payroll.prod.master' LIB('SYS2.prodlib') PGM(prodmst)
  ```

- When the active library list includes programs fetched from non-linklist libraries, eTrust CA-ACF2 performs rule validation for programs that were actually fetched from a linklist library. For example, if the following JCL was submitted and there were no libraries specified in the GSO LINKLST record and the program PRODMST is actually fetched from the linklist library SYS2.PRODLIB:

  ```
  //payprod   JOB  ............
  //JOBLIB    DD DSN=payroll.prod.load,DISP=SHR
  //          DD DSN=paytest.load,DISP=SHR
  //MSTREAD   EXEC PGM=promst
  //MSTFILE   DD DSN=pay.prod.master,DISP=SHR
  ```

  When access to data set PAY.PROD.MASTER is validated, the access environment is as follows:

  ```
  DSN='pay.prod.master' LIB('payroll.prod.load') PGM(prodmst)
  DSN='pay.prod.master' LIB('paytest.load') PGM(prodmst)
  DSN='pay.prod.master' LIB('sys2.prodlib') PGM(prodmst)
  ```

- You can use the GSO LINKLST record to simplify rule writing. When you define a library in the GSO LINKLST record, eTrust CA-ACF2 substitutes LIB('SYS1.LINKLIB') for the actual library name. eTrust CA-ACF2 uses SYS1.LINKLIB for access validation. By using the GSO LINKLST record, you can include both system linklist libraries and non-linklist libraries. The following series of examples illustrates how this works.

In the examples, the system linklist libraries and the GSO LINKLST record libraries are:

System Linklist	GSO LINKLST
SYS1.LINKLIB	SYS1.LINKLIB
SYS1.CMDLIB	SYS1.SORTLIB
SYS1.SORTLIB	SYS1.VTAMLIB
SYS1.VTAMLIB	SYS2.PRODLIB
SYS2.PRODLIB	TCAM.LOAD
OPRSYS.TSOLIB	

Without a STEPLIB or JOBLIB

Suppose the following JCL is submitted and program PRODMST resides in SYS2.PRODLIB:

```
//payprod   JOB  ..........
//MSTREAD   EXEC PGM=prodmst
//MSTFILE   DD DSN=payroll.prod.master,DISP=SHR
```

By default, eTrust CA-ACF2 searches the system linklist for the program PRODMST and the access environment is:

```
DSN='payroll.prod.master'
LIB('sys1.linklib')
PGM(prodmst)
```

The library name used for validation was SYS1.LINKLIB because library SYS2.PRODLIB was defined in the GSO LINKLST record. At this point, eTrust CA-ACF2 validates the access to data set PAYROLL.PROD.MASTER using program PRODMST from library SYS1.LINKLIB.

With a STEPLIB or JOBLIB (example 1 of 2)

Suppose the following JCL is submitted and the program PRODTST resides in PAYTEST:

```
//payprod   JOB   ...........
//JOBLIB    DD DSN=pay.test,DISP=SHR
//          DD DSN=pay.load,DISP=SHR
//MSTREAD   EXEC PGM=protst
//MSTFILE   DD DSN=payroll.prod.master,DISP=SHR
```

eTrust CA-ACF2 searches the JOBLIB libraries first and the program is fetched from PAYTEST. Now the access environment is:

```
DSN='payroll.parod.master' LIB('paytest') PGM(prodtst)
DSN='payroll.prod.master' LIB('pay.load') PGM(prodtst)
```

Access to all the possible libraries from which program PRODMST might be fetched are validated. Because the program PAYTEST was not fetched from one of the system linklist libraries, no validation for linklist libraries was done.

With a STEPLIB of JOBLIB (example 2 of 2)

Suppose the following JCL is submitted and the program PRODMST resides in SYS2.PRODLIB:

```
//payprod   JOB  ...........
//JOBLIB    DD DSN=pay.test,DISP=SHR
//          DD DSN=pay.load,DISP=SHR
//MSTREAD   EXEC PGM=prodmst
//MSTFILE   DD DSN=payroll.prod.master,DISP=SHR
```

eTrust CA-ACF2 searches the JOBLIB libraries first but the program is fetched from SYS2.PRODLIB, a system linklist library that is also defined in the GSO LINKLST record. Now the access environment is:

```
DSN='payroll.prod.master' LIB('pay.test') PGM(prodmst)
DSN='payroll.prod.master' LIB('pay.load') PGM(prodmst)
```

eTrust CA-ACF2 validated only the JOBLIB libraries and bypassed the validation for library SYS2.PRODLIB, which was converted to SYS1.LINKLIB because it was defined in the GSO LINKLST record. Validation was bypassed in this case because SYS1.LINKLIB is explicitly added to the active library list only when it (or they) is the only active library. This logic applies to all libraries defined in the GSO LINKLST record whether encountered in JOBLIB, STEPLIB, or the system linklist.

Scenario of a Test Program

The following scenario illustrates how a JOBLIB or STEPLIB statement can change the path of an executing program. In the scenario, a programmer wants to test three new payroll subroutines using the production payroll master file. You do not use eTrust CA-ACF2 to control access to data sets. Instead, manual change control procedures are used to ensure that test jobs do not access production data.

First, assume two libraries exist that contain the following programs:

PAYROLL.PROD.LOAD	USER123.LOAD
ASUBRTN	ASUBRTN
BSUBRTN	BSUBRTN
CSUBRTN	CSUBRTN
PRODMST	

Next, a job is submitted to execute the PRODMST program, such as:

```
//payprod   JOB  ..........
//*LOGONID  user123
//*PASSWORD password
//JOBLIB    DD DSN=user123.load,DISP=SHR
//          DD DSN=payroll.prod.load,DISP=SHR
//MSTREAD   EXEC PGM=promst
//MSTFILE   DD DSN=payroll.prod.master,DISP=SHR
```

In the concatenation the USER123.LOAD library was first so it was searched before PAYROLL.PROD.LOAD. But because USER123.LOAD did not contain the PRODMST program, the PRODMST program was found and fetched from PAYROLL.PROD.LOAD. As you can see, the PRODMST program accesses the payroll master file, named PAYROLL.PROD.MASTER.

This programmer wants to test three new subroutines used by the PRODMST program: ASUBRTN to calculate gross pay; BSUNRTN to calculate deductions; and CSUBRTN to create pay checks and the balance report.

The USER123.LOAD library contains all three subroutines and so does PAYROLL.PROD.LOAD. Because USER123.LOAD is first in the concatenation, the subroutines are loaded from USER123.LOAD. Thus, this programmer can use a production master file for test purposes.

Use of Program Pathing to Control Test Programs

In the previous section, you learned how a JOBLIB concatenation permitted a test program to use production data. Now, learn how the eTrust CA-ACF2 program pathing feature can prevent this type of occurrence.

The objective of this scenario is to ensure that the data set PAYROLL.PROD.MASTER can be accessed through only one program path. That is only when the program PRODMST is being used from the PAYROLL.PROD.LOAD library.

The first step is to write a rule for the PAYROLL data sets. Be sure to include the LIB and PGM parameters for the PAYROLL.PROD.MASTER data set and the PRODMST program. The presence of the PGM and LIB parameters in the rule entry signal to eTrust CA-ACF2 that data set PAYROLL.PROD.MASTER is program pathed.

```
$KEY(payroll)
 prod.load UID(pjcrmcr) R(A) W(A) E(A)
 prod.load UID(ser123) R(A) E(A)
 prod.master UID(*) LIB('payroll.prod.load') PGM(prodmst) R(A) W(A)
```

If you recall, the scenario assumed that two libraries exist with each containing the following programs:

PAYROLL.PROD.LOAD	USER123.LOAD
ASUBRTN	ASUBRTN
BSUBRTN	BSUBRTN
CSUBRTN	CSUBRTN
PRODMST	

Then, a job was submitted to execute the PRODMST program, such as:

```
//payprod   JOB  ............
//*LOGONID  user123
//*PASSWORD password
//JOBLIB    DD DSN=user123.load,DISP=SHR
//          DD DSN=payroll.prod.load,DISP=SHr
//MSTREAD   EXEC PGM=prodmst
//MSTFILE   DD DSN=payroll.prod.master,DISP=SHR
```

eTrust CA-ACF2 validates access to the data set PAYROLL.PROD.MASTER. Remember, access to PAYROLL.PROD.MASTER has a program path. It can be accessed only when the PRODMST program is fetched from the PAYROLL.PROD.LOAD library. Because there are two entries in the active library list (that is, USER123.LOAD and PAYROLL.PROD.LOAD), each representing a possible program path to PAYROLL.PROD.MASTER, eTrust CA-ACF2 validates both paths.

This means there are two validations:

1. To determine whether USER123 can access PAYROLL.PROD.MASTER using program PRODMST from the PAYROLL.PROD.LOAD library. Again, the rule set for PAYROLL data sets is:

   ```
   $KEY(payroll)
    prod.load UID(pjcrmgr) R(A) W(A) E(A)
    prod.load UID(user123( R(A) E(A)
    prod.master UID(*) LIB('payroll.prod.load') PGM(prodmst) R(A) W(A)
   ```

   Therefore, the access environment matches the rule entry.

2. To determine whether USER123 can access PAYROLL.PROD.MASTER using program PRODMST from USER123.LOAD. This access is denied because there is no rule entry that permits this access path.

USER123 is not permitted to use the PAYROLL.PROD.MASTER file to test his new subroutines.

Protecting Production Data

eTrust CA-ACF2 provides several means of controlling production data. You can use access rules to determine valid types of access, and you can use logonid record fields to define production logonid records.

You can use eTrust CA-ACF2 access rules to protect JCL libraries, which are often partitioned data sets. Programmers can be permitted to only read these libraries. You can permit individuals to write to these libraries, but should log these instances by means of a rule. Logging these updates provides a complete record of all changes and updates to production libraries. Changes and updates can be monitored to ensure a controlled environment. Procedure libraries can also be protected through the use of access rules.

You can limit production data access to production logonids by using access rules that provide read, write, and allocate authority only to the production logonids. Support programmers can be given the same privileges (temporarily or permanently), but with logging specified. All production logonids should be restricted. Restricted logonids require no password and are logged upon entry to the system. They cannot be used for online processing.

A logonid can contain the name of the program that must submit any JCL with this logonid. Also, the submitting program can be required to come from an APF-authorized library and from a permitted submission source (such as a particular reader).

eTrust CA-ACF2 provides two programs, ACFSUB and JOBCOPY, to control the submission of production jobs. The source code for these programs is provided with eTrust CA-ACF2. You can fit these programs in your site's present job scheduling package. For additional information regarding these two programs, see the *Reports and Utilities Guide*.

Protecting Job Submission

Consider the ways a job can be submitted and the source of that submission. Jobs can enter the system through local or remote statement readers, started tasks initiated by the operator, or from TSO, CICS, and IMS. Once inside the system, jobs or programs can submit other jobs to the internal reader. Every submitted job has a source identifiable to eTrust CA-ACF2. To ensure control of production processing, consider the following:

- Access to JCL and cataloged procedure libraries should be controlled to prevent unauthorized additions, changes, or deletions.
- A small, specific group of people located at specific terminals should be permitted to set up and submit production jobs.

- A specific program, stored in a highly-controlled library, should be used to submit production jobs. The library should be accessible to authorized users to submit production jobs only.
- Access to production data should be restricted to production logonids and, on an exception basis, possibly to specific support programmers (with their accesses logged).

Do not forget physical access control. No matter how thoroughly you control your libraries and sources, permitting unauthorized access to printouts of production data defeats your control efforts.

Execution Flow

The flow of control during execution of a job making a data set access request is as follows:

1. If the access is for the volume table of contents (VTOC) on a resident volume, then the data set name is changed to SYSVTOC.*volser* if the first character of the volser is an alphabetic or to SYSVTOC.@*volser* if it is numeric.
2. The installation data set pre-validation exit is taken. This exit can modify the data set name or force a disposition for the request.
3. Non-Tape Data Sets. If the data set resides on the resident set of volumes (RESVOLS GSO record), the actual data set name is used in the search. If the data set resides on the secured set of volumes (SECVOLS SO record), the data set name @volser.VOLUME or VOLUME.@volser is used in the search. In this name, @volser is the actual volser, and VOLUME is literally the word VOLUME. (The exact name format is determined by the VOLRULE option of the GSO OPTS record.) If the data set's volume resides on neither RESVOLS nor SECVOLS, the data set is unprotected. In this case, the Pseudo Data Set Name Generator exit (DSNGEN) is taken.

   Tape data sets. Validation can follow one of these cases:

   - The access attempt might request tape bypass label processing (tape BLP). eTrust CA-ACF2 then gets the volser from the JCL. If the JCL does not specify a volser, eTrust CA-ACF2 attempts to get the volser from the catalog.
   - The access attempt might request tape BLP while the user logonid has the TAPE-BLP attribute. In this case, the label on the tape is ignored. eTrust CA-ACF2 gets the volser from the JCL or, alternatively, the catalog.

- The access attempt might request tape BLP while the user logonid has the TAPE-LBL attribute. If the tape has a standard label, then eTrust CA-ACF2 attempts to read the actual volser from the tape. If the tape has a nonstandard label, eTrust CA-ACF2 gets the volser from the JCL or, alternatively, the catalog.
- The access attempt might request tape BLP while the program has been authorized for tape bypass label processing (in the GSO BLPPGM record). eTrust CA-ACF2 gets the volser from the JCL or, alternatively, the catalog.
- The access attempt might request tape BLP while no bypass label authority exists. The access request is aborted unless a post-validation exit changes this access recommendation. No further validation takes place.

  After the previous validation takes place, eTrust CA-ACF2 determines whether the data set resides on a secured volume, as follows:

- If the data set resides on a secured volume, then the volser is put in the format @*volser*.VOLUME or VOLUME.@*volser*, depending on the installation option. Validation continues with the next step.
- If the data set does not reside on a secured volume and a DSNGEN exit exists, then the exit is taken and validation continues with the next step.
- If the data set does not reside on a secured volume and a DSNGEN exit does not exist, then eTrust CA-ACF2 checks whether the system-wide option for tape data set validation is in effect (through the GSO TAPEDSN record). If so, then validation continues with the next step. Otherwise, the access is allowed (assuming that an unauthorized dump is not being taken).

4. If the data set is temporary or begins with the owned prefix of the user, access is allowed.
5. If the logonid is marked NON-CNCL, then the list of allowable maintenance logonids, programs, and libraries is checked. (This list is in the GSO MAINT record.) If a match is found, access is allowed. If a match is not found, then normal validation continues. (However, if normal validation does not allow the access, then the access is allowed but logged.)
6. If the logonid is marked MAINT, then the list of allowable maintenance logonids, programs, and libraries is checked. If a match is found, then access is allowed. If a match is not found then normal validation continues.
7. If not allowed under one of the previous conditions, the list of system-wide resident rules is searched for the correct rule set. If found, this set of rules is interpreted.
8. If the applicable rule is not in the system-wide resident set, the address space list is searched. If found, this set of rules is interpreted.

9. If the rule is not in the address space list, a VSAM GET is performed out of the globally shared resource pool to obtain the record. If the record exists, it is chained to the address space list of resident rules and then interpreted. If it is not found, the request is aborted.
10. The appropriate access rule set is interpreted. If a matching access rule says allow or allow and log, then the appropriate action is taken.
11. If the request is to be aborted but the logonid is marked NON-CNCL or is a security administrator with the proper scope, the access is allowed but logged. Otherwise, the request is aborted unless the site-supplied security violation exit requests that the access be allowed.

    Note: Depending on the eTrust CA-ACF2 mode, accesses might be allowed but logged instead of being aborted where indicated in the previous execution flow. (For example, the system-wide MODE option can be set to LOG versus ABORT during a transition period.)

    Optionally, a site can specify in a logonid record that all data set accesses require authorization in an access rule. This requirement can override other attributes specified in the user's logonid record. However, the NON-CNCL attribute can be used to override rule entries. The READALL attribute allows read-only access not allowed by rule entries.

For every job, a Rule database access is performed only once for each unique high-level index referenced. Furthermore, no access is performed for resident data sets owned by the user or for temporary data sets.

Chapter

7

Maintaining Resource Rules

In addition to controlling access to the computer system and data, eTrust CA-ACF2 provides protection for other system resources such as:

- CICS resources (transactions, files, programs)
- IMS resources (transactions, commands)
- SAF-protected resources (programs, console commands).

Resource rules, similar to access rules, provide this protection. Resource rules must include a resource name and resource type. This is done using the $KEY and TYPE control statements, respectively. Extended resource keys can be used to protect resources with qualified resource names. For a complete description of these control statements, see Control Statement Descriptions later in this chapter.

You can control the use of programs with resource rules. If your site uses DFSMS, see the "Implementing DFSMS Support" chapter for information. If you are not using DFSMS, you can control program use with Note 6. See Appendix B, "eTrust CA-ACF2 Notes," in the *Systems Programmer Guide* for information.

This chapter describes:

- An example of a eTrust CA-ACF2 resource rule set
- The elements of the eTrust CA-ACF2 resource rule set including control statements, comment statements, and resource rule entries
- Masking in resource rule sets
- Use of NEXTKEY
- eTrust CA-ACF2 resource rule validation
- Creating resource rule sets
- Maintaining rules using the ISPF panels and the ACF command

Example of a Resource Rule Set

When listed, a simple resource rule set might look like this:

```
$KEY(acfm) TYPE(ckc)
 UID(tfintec) ALLOW
```

You can interpret this rule set as follows:

$KEY(ACFM)
Indicates that the rule set pertains to the eTrust CA-ACF2 CICS transaction named ACFM. (The ACFM transaction provides eTrust CA-ACF2 security administration through CICS.)

TYPE(CKC)
Defines the type code associated with this rule set. Type CKC tells us that this sample rule set pertains to CICS transactions.

UID(TFINTEC)
Identifies all users whose UIDs begin with TFINTEC.

ALLOW
Specifies that the previously identified users can access the ACFM transaction.

Resource rule sets are generally larger than this sample rule set. Even the more complicated rule set that follows, which shows customized access for various users, is brief compared to the larger rule sets you might create.

```
$KEY(acfm) TYPE(CKC)
 UID(tfinpayiso) ALLOW
 UID(tfinfiniso) LOG
 UID(tfinadmiso) UNTIL(11/24/04) ALLOW
 UID(tfinpaynlt) SHIFT(regular) ALLOW
```

A resource rule set consists of resource rule entries that pertain to resources of a particular type. Each resource rule set is made up of:

- Control statements
- Resource rule entries
- Comment statements

Control Statements

```
$KEY(acfm) TYPE(ckc)
 UID(tfintec) ALLOW
```

In the previous sample rule set, the first line is called a control statement. The second line is called a resource rule entry. Control statements must begin in column one. Two types of control statements are the dollar sign ($) control statements and the percent symbol (%) control statements. A rule set can contain only one of each type of $ control statements but as many of each type of % control statements as desired. A $KEY with a TYPE statement or a $KEY and a $TYPE control statement is required.

Where to Place Control Statements

Follow these guidelines when specifying control statements:

- Specify appropriate $ or % control statements in a rule set. You can specify the same $ control statement more than once, but eTrust CA-ACF2 only uses the last entered value.
- A $KEY control statement with a TYPE or a $KEY and a $TYPE control statement is required.
- Specify all $ control statements before any other statements.
- Specify multiple $ control statements on one line if the $ appears in column one.

  ```
  $KEY(trana) PREFIX(tran*) TYPE(ckc)
  ```

How to Continue Control Statements

To specify that you want a control statement to continue onto the next line, use a dash (-) as the last nonblank character on the line. eTrust CA-ACF2 recognizes a continuation character unconditionally. For example, when you continue a comment statement, eTrust CA-ACF2 treats the next line as a continuation of the comment statement, even if that line has the format of a control statement.

Format Requirements for Control Statements and Rule Entries

Input from a partitioned data set can be variable format, with the sequence field as the first eight characters of the record (as in TSO CLIST or PL/I), or fixed-format 80-byte records with the sequence field as the last eight characters in the record (as in DATA- or CNTL-type resources). You can specify multiple $ control statements on the same line with a single $ in column one.

For example, you can use the following formats:

```
$KEY(trana)
$PREFIX(tran*)
$TYPE(ckc)
```

Or

```
$KEY(trana) PREFIX(tran*) TYPE(ckc)
```

Control Statement Syntax

The syntax of the resource rule control statements is:

```
$Key(resourcename) [Type(typecode)]
[$Type(typecode)]
[$Member(membername)]
[$NORulelng]
[$NOSort]
[$Owner(owner-id)]
[$Prefix(prefix)]
[$Recname(recname)]
[$Userdata(userdata)]
[%Change uidmask1,...,uidmaskn]
[%RChange uidmask1,...,uidmaskn]
```

Control Statement Descriptions

This section describes each of the control statements available for resource rules.

$Key(*resourcename*)
Specifies the name of the resource (*resourcename*) to which this rule applies. eTrust CA-ACF2 uses the $KEY value to select the appropriate resource rule to use in validating a particular resource name. The $KEY control statement is required.

A protected resource must have a resource name that is unique from the names of all other resources of the same type. Other program products might see resource names as *entities* or *entity names*.

A resource name is from one to 256 characters long, depending on whether it is *unqualified* or *qualified*.

- An unqualified resource name is a one to 40-character string with no periods delimiting sections or qualifiers of the resource name.

- A qualified resource name is in the format *qualifier1.qualifier2...*, where the first qualifier is one to forty characters long, followed by a period and one or more subsequent qualifiers. All the qualifiers together make up the complete resource name. The maximum length for qualified resource names is 256 characters.

  Extended resource keys can also be used to protect resources with qualified resource names. See Extended Resource Keys later in this chapter for detailed information.

Resource names can contain embedded blanks or spaces, and can also be masked. When you mask the $KEY value for a resource record, a directory for the resource type must be locally or globally resident for any validation processing. See $KEY Masks and Globally and Locally Resident Directories in Using Masking in Resource Rules later in this chapter for more information about masking the $KEY and resident directories.

Note: If the MIXED indicator in the CLASMAP record is specified, then the resource name is case sensitive.

The following are valid $KEY values:

```
$KEY(TESTNAME)
$KEY(TEST NAME)
$KEY(TEST.NAME)
$KEY(TEST.NAME.THAT.IS.SUPPORTED.EVEN.THOUGH.IT.IS.VERY.LONG)
$KEY(TEST*NAME)
$KEY(TEST*****)
```

The following are invalid $KEY values:

```
$KEY(TESTRESOURCENAMETHATISTOOLONGFORAVALIDRESOURCEKEY)
$KEY(TEST RESOURCE NAME THAT IS TOO LONG FOR A VALID RESOURCE KEY)
$KEY(TESTQUALIFIERTHATISTOOLONGFORAVALIDRESOURCEKEY.QUALIFIER2)
```

Follow these guidelines in specifying the $KEY value for a resource rule:

- When the resource name is a single unqualified resource name, it must be forty characters or less. You must specify the entire resource name or its masked equivalent in the $KEY. For example:

```
Resource name:  TESTNAME

Sample $KEY values:  TESTNAME
                     ****NAME
                     TEST****

Sample resource rule:

$KEY(TESTNAME) TYPE(ttt)
 UID(-) ALLOW
```

- When the resource name is longer than forty characters, qualify the resource name in the format *qualifier1.qualifier2....* The high-level qualifier must be forty characters or less followed by a period; all subsequent qualifiers make up the complete resource name. You must specify the high-level qualifier or its masked equivalent in the $KEY value for the rule. Specify the remainder of the resource name (or its mask) in the extended resource rule key field of the individual rule lines, as explained in Extended Resource Key Masks later in this chapter. For example:

```
Resource name:  TEST.NAME.THAT.IS.SUPPORTED.EVEN.THOUGH.IT.IS.VERY.LONG

Sample $KEY values:  TEST
                     T**T
                     ****

Sample resource rule:

$KEY(TEST) TYPE(ttt)
 NAME.-  UID(-) ALLOW
 OTHER.- UID(-) ALLOW
 -       UID(-) LOG
```

- When the resource name is a one to 40-character qualified resource name, you can specify the entire resource name (or its mask) as the $KEY value, or the high-level qualifier (or its mask) in the $KEY value for the rule, with the remainder of the resource name specified in the extended resource rule key field of the individual rule lines. For example:

```
Resource name:  TEST.QUALIFIED.NAME

Sample $KEY values:  TEST.QUALIFIED.NAME
                     TEST.*********.NAME
                     ******************
                     TEST
                     T**T
                     ****

Sample resource rules:

$KEY(TEST.*********.NAME) TYPE(ttt)
 UID(-) ALLOW

$KEY(TEST) TYPE(ttt)
 QUALIFIED.NAME UID(-) ALLOW
 OTHER.-        UID(-) ALLOW
 -              UID(-) LOG
```

- When a qualified resource name is specified in the $KEY value, extended resource keys cannot be specified in any of the resource rule lines.

  For example:

```
$KEY(TEST.QUALIFIED) TYPE(ttt)
NAME UID(-) ALLOW
```

  Will generate the following compiler error:

```
ACF70041 SECOND LEVEL QUALIFIER NOT ALLOWED WITHIN THIS RULESET
```

  Note: For consistency, we strongly recommend the $KEY always be the high level qualifier on a multi-level resource, regardless of the actual size of the resource.

$TYPE(*typecode*)

Specifies the one to three-character resource type for this rule. Resource types denote a logical class of resources, such as transactions, account or billing numbers, and so forth. Each resource type is identified by a type code that consists of one to three characters, preferably three alphanumeric characters (for example, CKC for CICS transactions). The $TYPE control statement is required.

Note: Other program products might see resource types as resource classes. For a list of resource classes protected by SAF, see Appendix B, "IBM-Supplied Resource Classes."

eTrust CA-ACF2 provides the following resource type codes that you cannot change:

- SAF—System Authorization Facility (SAF) resource validatin other than data sets, disk, or tape volumes
- TAC—Time Sharing Option (TSO) account resource rule sets
- TPR—Time Sharing Option (TSO) procedure resource rule sets
- TGR—Group or project name resource rule sets
- VTA—VTAM ACB OPEN resource rule sets

Note: The TGR and VTA resource types have unique requirements. See Requirements for Predefined Resources later in this chapter.

The following additional resource type codes are provided by eTrust CA-ACF2, but you can rename them:

- CFC—CICS file control rule sets
- CKC—CICS transaction control rule sets
- PSB—CICS DL/I request rule sets
- CPC—CICS program control rule sets
- CTD—CICS transient data rule sets
- CTS—CICS temporary storage rule sets
- CMR—CICS MRO SYSID resource rule sets
- IAG—IMS application group name rule sets
- ICM—IMS command rule sets
- IPS—IMS PSB rule sets
- ITR—IMS transaction control rule sets
- SMS—DFSMS resource class rule sets. Do not use an asterisk (*) as a character in the $TYPE field (for example, $TYPE(**C) or $KEY(aaa) TYPE(**C)). eTrust CA-ACF2 reads asterisks in the TYPE field as the asterisk character instead of a mask when interpreting resource rules.

$Member(*membername*)

Specifies the member name to be used for a decompile into a partitioned data set (PDS) if one is not provided with the decompile request. For an explanation of how the $MEMBER control statement affects processing of the DECOMP subcommand, see DECOMP Subcommand later in this chapter.

When you specify a $MEMBER name that matches the $KEY value for the resource rule, eTrust CA-ACF2 issues a warning and ignores the $MEMBER control statement.

$NORULELNG

Overrides the use of the *rulelong* compiler when GSO RULEOPTS RULELONG is active. Normally, CA-ACF2 uses the *rulelong* compiler to compile rules if the RULELONG option is set. The rulelong format is an expanded record format. If a ruleset is small and therefore does not require the *rulelong* format and you do not want the features of *rulelong*,, specifying $NORULELNG in the rule lets you compile a ruleset using a compact record format. This way, you can choose to compile rules with the format that is best suited for the ruleset.

Note: If $NORULELNG was specified in the rule, this option is not necessary. If the dynamic compile option (COMPDYN) is set in the GSO RULEOPTS record, then the $NORULELNG control statement is not needed to compile rule sets of varying size.

$NOSort

Specifies that eTrust CA-ACF2 **should not** sort resource rule entries when it stores this rule set. The rules remain in the order in which they were first entered into the compiler through the terminal or a partitioned data set. See Creating Resource Rule Sets later in this chapter for more information.

We recommend that you **do not** use $NOSORT because eTrust CA-ACF2 normally sorts and then stores a set of resource rule entries from most specific access environment to most general access environment. When you use $NOSORT, a general resource rule could prevent a more specific rule from being evaluated.

When you specify a $NOSORT control statement, eTrust CA-ACF2 issues a warning message during compilation.

The $NOSORT statement is effective only when you have specified $NOSORT in the GSO RULEOPTS record. See eTrust CA-ACF2 Rule Option Specifications (RULEOPTS) in the "Maintaining Global System Options Records" chapter for a description of the $NOSORT option.

$Owner(*ownerid*)

A 24-byte field that contains the contents for the $OWNER control statement. If the $OWNER control statement is not present, this field contains binary zeros.

$Prefix(*prefix*)

The $PREFIX control statement is optional. You can specify it in a resource rule to provide additional or alternate matching criteria for the resource name being validated. Resource names can be masked as described in $PREFIX Masks in Using Masking in Resource Rules later in this chapter. The $PREFIX statement is most useful when specified with the NEXTKEY parameter (see Use of NEXTKEY in Resource Rules later in this chapter), but it can be specified in any resource rule.

Use the $PREFIX control statement to specify a one to 40-character alternate $KEYkey value with the format *qualifier1.qualifier2....*

Note: If the MIXED indicator in the CLASMAP record is specified, then the resource name is case sensitive.

After eTrust CA-ACF2 selects a resource rule for validation based on the $KEY value, it processes the $PREFIX statement if one exists. **Regardless of how the rule was selected for validation, the full resource name is matched against the $PREFIX value.** After it performs this match, eTrust CA-ACF2 uses any unmatched portion of the resource name to match against the extended resource keys specified in the resource rule lines. For example, with a resource name of TEST.TESTNAME2, you could create the following resource rule:

```
$KEY(TEST) TYPE(ttt)              qualifier match
$PREFIX(TEST.*********)           matches full resource name
UID(...) ...                      no extended resource key
```

When the $PREFIX value does not match the equivalent part of the resource name, eTrust CA-ACF2 denies access to the resource. For example, with the resource name of TEST.TESTNAME2, you could create the following resource rule:

```
$KEY(TEST.TESTNAME*) TYPE(ttt)      full key match
$PREFIX(TEST.SOMETHING)             doesn't match resource name
UID(...) ...

Access is denied.
```

When you specify $PREFIX(), eTrust CA-ACF2 issues a warning message to indicate that the $PREFIX is null and is ignored. eTrust CA-ACF2 does not generate a $PREFIX statement when it decompiles the rule.

$Recname(*recname*)

Specifies the name of a RECORD definition record eTrust CA-ACF2 is to validate. The $RECNAME control statement points to a RECORD definition record just as the SHIFT parameter points to a shift record. For more information about RECORD definition records and record-level protection, see the "Maintaining Field Records" chapter.

$Userdata(*text*)
Specifies any one to 64-character text string. eTrust CA-ACF2 stores the information in the $USERDATA field with the rule set. Site postvalidation or violation exits can access this information.

Comment statements, represented by an asterisk (*) in column one and a space in column two, also designate user information but are removed during compilation.

%Change uidmask1,...,*uidmaskn*
Specifies the UIDs or UID masks of those individuals, other than the security administrator, who can replace or delete a set of rules. Separate multiple UIDs or masks with commas or blank spaces.

%CHANGE lets the designated user modify the rule set to further delegate change authority to other users. The designated user can change or delete any part of the rule set. To activate or deactivate the site's ability to use this control statement, see the description of the CHANGE field of the GSO RULEOPTS record in eTrust CA-ACF2 Rule Option Specifications (RULEOPTS) the "Maintaining Global System Options Records" chapter.

To delegate rule writing authority, a security administrator compiles and stores a rule set that contains only $KEY, $TYPE, and %CHANGE control statements. The designated changer then refines this stored rule set without the security administrator having to write any rule entries.

%RChange *uidmask1,...,uidmaskn*
Specifies the UID strings or UID masks of users who have restricted change authority over the rule set. Designated users can change individual entries but not control statements. They cannot delegate any change authority or delete the rule set. If the same user matches entries in both %CHANGE and %RCHANGE, the %CHANGE authority prevails.

To activate or deactivate the site's ability to use the %RCHANGE control statement, see the description of the CHANGE option of the GSO RULEOPTS record in eTrust CA-ACF2 Rule Option Specifications (RULEOPTS) in the "Maintaining Global System Options Records" chapter.

Extended Resource Keys

When a resource name is longer than forty characters, it must be a qualified resource name and you must use an extended resource key in the generalized resource rule for the resource.

You can use extended resource keys for any resource name with fewer than forty characters if the resource name is qualified.

Most resource rules with extended resource keys use the first qualifier of the resource name (or a resource name mask) as the $KEY value and specify the remainder of the resource name (or a resource name mask) in individual rule lines. For example, if the resource name is TEST.TESTNAME2, you could create the following resource rule:

```
$KEY(TEST) TYPE(ttt)
TES TNAME2 UID(...) ALLOW
-          UI D(*)   PREVENT
```

See Extended Resource Key Masks in Using Masking in Resource Rules later in this chapter for more information about masking extended resource keys.

Requirements for Predefined Resources

The TGR and VTA resources have unique requirements as described in the following.

TGR (Group or Project Name)

At system entry, eTrust CA-ACF2 validates a group name against resource rules to determine whether the user can associate with that group. If the user enters an invalid group name, eTrust CA-ACF2 denies access.

Use the type code TGR to identify resource rules that validate the group name at system entry. The $KEY control statement contains the one to eight-character group name. This field is maskable.

See "Controlling System Entry" for more information about writing these resource rules.

VTA (VTAM ACB OPENs)

You can write resource rules to protect the opening of VTAM application definition statements (APPLIDs). OPENs to APPLIDs are performed using VTAM access method control blocks (ACBs). eTrust CA-ACF2 protects access to APPLIDs by validating ACB OPENs. Controlling access to VTAM APPLIDs lets you prevent unauthorized use of VTAM resources, since APPLIDs control access to VTAM resources.

ACBs are known to eTrust CA-ACF2 by their associated APPLIDs rather than by the ACBNAMEs defined to VTAM, so specify APPLIDs as the resource names in your rules. Ask your site's systems programmer for the required APPLIDs.

VTAM OPEN validation can be activated either via the GSO OPTS records or via the SAF interface for classes APPL and VTAMAPPL. Though either option will work, we recommend using the SAF interface to protect these classes.

For more information about VTAM ACB OPEN validations, see the "Special Usage Considerations" chapter in the *Systems Programmer Guide*. For information about the GSO OPTS record, see eTrust CA-ACF2 Option Specifications (OPTS) in the "Maintaining Global System Options Records" chapter.

Resource Rule Entries

Individual resource rule entries follow the control statements in a rule set. Each rule entry specifies an environment and access permission under which access to a particular resource can take place.

When an access attempt is made, eTrust CA-ACF2 matches the environment of the access attempt against the environment specified in the resource rule. Once this match is satisfied, eTrust CA-ACF2 determines access on the basis of the access permissions.

Rule Entry Syntax

The syntax of an individual resource rule entry is:

```
[rsrcmask]
[UId(uidmask)] [SOurce(sourcemask)] [SHift(shiftname)] -
[UNtil(date)|FOR(days)] [ACTIVE(date)] -
[SErvice(Read,Add,Update,Delete,Execute)] -
[Data(text)] [Verify] -
[Nextkey(nextkey)] [Reccheck(recid)]-
[Allow|Log|Prevent]
[*comment]
```

Components of a Resource Rule Entry

The information in a resource rule entry is classified into one of the following types.

Environment

Specifies what is to be accessed, who can access it, and under what conditions the access can take place. The parameters that describe the environment are:

- ACTIVE
- FOR
- RECCHECK
- rsrcmask
- SERVICE
- SHIFT
- SOURCE
- UID
- UNTIL
- VERIFY

For more information about how these parameters affect validation, see How Resource Rule Lines Are Selected for Validation later in this chapter.

Access Permission

Specifies whether the access is allowed, logged, or prevented.

Pointer

Specifies an alternate resource rule set for eTrust CA-ACF2 to check to perform validation. The NEXTKEY parameter specifies this alternate rule set.

Comment Statements

Specifies a comment that provides more detail about the rule set. Comment statements begin with an asterisk (*) in column one. The DATA parameter also lets you enter comments.

Rule Entry Parameters

The parameters that make up a resource rule entry are:

rsrcmask
Completes the name of the resource to be protected. Do not enclose the values for *rsrcmask* in single quotes. You can use extended resource keys that contain one or more qualifiers delimited with periods for the *rsrcmask* value. However, extended resource keys cannot be used if a qualified resource name was specified in the $KEY value of a resource rule. See Extended Resource Keys earlier in this chapter for more information. For guidelines to use when masking the resource rule line extended key, see Extended Resource Key Masks later in this chapter.

Note: If the MIXED indicator in the CLASMAP record is specified, then the resource name is case sensitive.

UId(*uidmask*)
Specifies the UID or UID mask of the user making the access attempt. If omitted, this parameter includes all users. Masking UIDs is described in the chapter entitled, "Maintaining Access Rules." If you do not specify a value for *rsrcmask*, the UID parameter must be the first parameter you specify.

SOurce(*sourcemask*)
Specifies a single name of an input source or group of input sources (logical or physical) through which access is being attempted. If omitted, this parameter includes all input sources. You can mask this field but masking, if used, will not include a lookup of source group records. For this reason, and because this field cannot contain multiple entries, we recommend you use source groups to group multiple sources specifying the source group name instead of using a mask. See Maintaining Structured Infostorage Records chapter for more information about setting up multiple sources.

SHift(*shiftname*)
Specifies the one to eight-character name of the shift record that defines the time, day, or date this rule line will be used when determining the access attempt. If omitted, this parameter includes any time or day.

Do not enter more than one SHIFT parameter in a rule entry. To create new day and time combinations, insert a new shift record. See the "Maintaining Shift and Zone Records" chapter for a description of this procedure.

UNtil(*date*)
Specifies the last date on which an access can take place under this rule. Acceptable Gregorian date formats include *mm/dd/yy*, *yy/mm/dd*, or *dd/mm/yy*. (The DATE field of the GSO OPTS record determines this date format. See the "Maintaining Global System Options Records" chapter for more information.) Years specified as 70-99 assume a date in the 20th century (1970-1999); years specified as 00-69 assume a date in the 21st century (2000-2069). Once the rule line has expired, it will automatically be discarded by the compiler the next time the rule is compiled.

ACTIVE(*date*)
Specifies the first date on which an access can take place under this rule. Acceptable Gregorian date formats include *mm/dd/yy*, *yy/mm/dd*, or *dd/mm/yy*. (The DATE field of the GSO OPTS record determines this date format. The ACTIVE field is only valid with RULELONG. See eTrust CA-ACF2 Option Specifications (OPTS) in the "Maintaining Global System Options Records" chapter for more information. Years specified as 70-99 assume a date in the 20th century (1970-1999); years specified as 00-69 assume a date in the 21st century (2000-2069).

This parameter is valid only when the GSO RULEOPTS RULELONG parameter is set.

For(*days*)
Specifies the number of days, from the compilation date of the resource rule set, during which an access attempt can be made under this rule. The minimum number you can specify is zero (meaning today only); the maximum is 365. Each time a rule with the FOR parameter is compiled, CA-ACF2 converts the FOR parameter into an UNTIL date parameter. Caution is necessary in using the FOR parameter so that subsequent compiles of the rule do not extend the life of the rule entry beyond the intended data. For this reason, it is recommended that the UNTIL parameter be used instead of the FOR parameter.

SErvice(Read,Add,Update,Delete,Execute)
Specifies the type of resource access associated with the access attempt. Specify one or more resource access keywords, using blank characters or commas to separate them. These keywords include:

- Read—indicates read-only access.
- Add—indicates access to add new records to an existing file.
- Update—indicates access for the purpose of modifying existing records.
- Delete—indicates access for the purpose of deleting records.
- Execute—indicates execute access. Execute does not imply READ access; that is controlled by the caller of the resource validation call.

Not all of the SERVICE keyword options apply to all of the eTrust CA-ACF2 product interfaces. For specific information about the options available for the eTrust CA-ACF2 product interface you are using, see the appropriate eTrust CA-ACF2 support guide. When you do not specify a SERVICE keyword, **all** services is the default.

Data(*text*)
Specifies up to 64 characters of data for your own use. This data is passed to the user exits and back to the calling application subsystem with the global USERDATA field. This data could indicate further checking or contain some control information.

Verify
Indicates that the caller should verify the identity of the user by doing a password reverification. This is normally used when a rule indicates an allow condition. The application subsystem requesting access to a resource is informed of the request for password verification. It is up to that application to actually issue the password prompt or decide to simply allow the request.

Reccheck(*recid*)
Specifies the name of the EXPRESSN record eTrust CA-ACF2 uses for this validation. The EXPRESSN record defines a Boolean expression that eTrust CA-ACF2 evaluates to determine whether a user can access a record based on the contents of a field. For more information about EXPRESSN records and record-level protection, see the “Maintaining Field Records” chapter.

Allow
Specifies that access to the resource is allowed.

Log
Specifies that access to the resource is permitted but logged. eTrust CA-ACF2 writes a System Management Facility (SMF) record to log the event.

Prevent
Specifies that access to the resource is prevented. eTrust CA-ACF2 writes an SMF record to log the event. PREVENT is the default.

NEXTKEY(*nextkey*)
Specifies the key of an alternate rule set eTrust CA-ACF2 checks if access to this resource is **denied** based on this rule entry. eTrust CA-ACF2 **only checks the NEXTKEY parameter when the access matches the environment criteria, but the rule line prevents the access**. Validation of the access continues with the evaluation of the alternate resource rule set. See Use of NEXTKEY in Resource Rules later in this chapter for more information about using this parameter.

Note: If the MIXED indicator in the CLASMAP record is specified, then the resource name is case sensitive.

Comment Statements

Comment statements, denoted by an asterisk (*) in column one, let you place any text inside the uncompiled rule set. The text on a comment statement is stripped off during the compile process. After you compile the rule set, the comments are lost. A decompile does not display the comments. Use the DATA parameter if you want comments to remain with the rule set.

Edit rules containing comment statements with care. In the following example, when you move the commented rule entry to the top of the rule, everything is commented out but the $KEY entry, because the dash signals that the line is to be continued:

```
$KEY(ABC) TYPE(CKC)
* UID(PAYCLK1) SHIFT(ALL) SERVICE(READ,UPDATE) DATA(BILL CAN UPDATE) -
 UID(PAYMGR) SERVICE(READ) ALLOW
```

As you can see, the rule does not make much sense this way, and eTrust CA-ACF2 cannot process it.

Using Masking in Resource Rules

You can use masks to represent multiple resource names. Masking is especially useful when you create your own resource names. You can standardize the names of resources of a certain type so that certain names fit a particular mask, permitting one rule set to control access to those resources.

Using masks, you can write a resource rule set for a group of resources and still write a unique rule set for a single resource in that group. For example, using the asterisk (*) masking character, you can write a resource rule set that applies to all resources of a given type code whose names are from one to eight-characters and begin with G:

```
$KEY(g*******)
```

You can then create another resource rule set that applies specifically to the transaction GETT:

```
$KEY(gett)
```

When validating access to the resource GETT, eTrust CA-ACF2 evaluates the resource rule set with the $KEY value of GETT, since that name forms a more specific rule ID.

eTrust CA-ACF2 interprets mask characters in the $KEY and $PREFIX control statements and in extended resource key masks.

$KEY Masks

You can use the asterisk (*) to mask the $KEY control statement value. However, you cannot use the dash (-) as a masking character in the $KEY value. When a dash occurs in the $KEY value, eTrust CA-ACF2 treats it as a literal character for matching purposes. For example, a resource name or qualifier of TEST.ABC- matches only TEST.ABC-.

Using the Asterisk

You can use the asterisk (*) to mask the $KEY control statement as follows:

- If one or more asterisk (*) mask characters occur at the beginning of or between the characters of the $KEY value, each asterisk matches exactly one character in the resource name or qualifier.

```
TEST***WORD     will match   TESTTHEWORD
                             TEST   WORD
                             TEST.A.WORD

                but not      TESTWORD       (missing characters)
                             TEST WORD      (missing characters)
                             TEST.WORD      (missing characters)
                             TEST.NOTHING   (unmatched characters)

*****WORD       will match   TEST WORD
                             TEST.WORD

                but not      TESTWORD       (missing characters)
                             WORD           (missing characters)
                             TEST           (unmatched characters)
```

- If one or more asterisk (*) mask characters occur at the end of the $KEY value, they match the same number of or fewer characters in the resource name or qualifier.

```
TEST.NAME***    will match   TEST.NAME
                             TEST.NAME123
                             TEST.NAME.2

                but not      TEST.NAM       (too short)
                             TEST.NAMX      (different character)
                             TEST.NAME1234  (too long)

TEST***         will match   TEST
                             TEST123
                             TEST 12

                but not      TEST1234       (too long)
```

- The asterisk (*) character can mask the period delimiter in a qualified resource name when the total length of the resource name is forty characters or less and the $KEY value matches the entire resource name.

```
TEST***WORD     will match  TEST.A.WORD
                but not     TEST.ABC.WORD (unmatched characters)

*****WORD       will match  TEST.WORD
                but not     ABC.WORD      (unmatched characters)

**********      will match  TEST.WORD
```

Globally and Locally Resident Directories

When you mask the $KEY value for a resource record, a directory for the resource type must be locally or globally resident.

Globally Resident Directories

A *globally resident directory* is a directory (or list) of the resource rules for a particular resource class (type) that exists in common system storage. eTrust CA-ACF2 uses this directory for all validations for the resource class, regardless of where in the system the resource access occurs.

Use globally resident directories if you mask the $KEY in the generalized resource rule or when more than one address space in the system makes validation calls for the same resource class. For example, ICS regions using the same set of resource rules prevents duplication. Having the rules resident reduces storage utilization in the region. Resource classes for most SAF calls must have globally resident directories to reduce I/O and improve performance. SAF FASTAUTH calls require resident directories because there can be no I/O.

Create a globally resident directory for a resource class by specifying the resource class in a GSO INFODIR record.

Locally Resident Directories

A *locally resident directory* is a directory (or list) of the resource rules for a particular resource class (type) created in the storage of a particular address space. eTrust CA-ACF2 uses this directory for all validations for the resource class from that address space.

Use locally resident directories when only one address space makes the validation calls for the resource class.

The eTrust CA-ACF2 CICS and eTrust CA-ACF2 IMS interfaces automatically create locally resident directories for the resource classes of the resources they protect, unless a globally resident directory already exists for the resource class. Other applications can use the ACGRSRC macro to request the creation of a locally resident directory. See the *Systems Programmer Guide* for more information about this macro.

$PREFIX Masks

You can use the asterisk (*) to mask the $PREFIX value. However, you cannot use the dash (-) as a masking character in the $PREFIX. When a dash occurs in the $PREFIX value, eTrust CA-ACF2 treats it as a literal character for matching purposes. For example, a $PREFIX value of TEST.ABC- matches only TEST.ABC-.

Using the Asterisk

You can use the asterisk (*) to mask the $PREFIX control statement as follows:

- The asterisk (*) cannot be used to mask the period delimiter in a qualified resource name in the $PREFIX value. Periods separating qualifiers must be specified; they designate the end of one qualifier and the start of another.

```
$PREFIX value              Portion of Resource Name

TEST.NAME***  ma tches     TES T.NAME
                           TES T.NAME123
              bu t not     TES T.NAME.2     (extra qualifier)
```

- When one or more asterisk (*) mask characters occur at the beginning of or between the characters of a qualifier in the $PREFIX value, each asterisk matches exactly one character in the corresponding qualifier of the resource name.

```
$PREFIX value           P ortion of Resource Name

TEST.***WORD  ma tches  T EST.THEWORD
                        T EST.HISWORD
              bu t not  T EST.WORD     ( missing characters)
                        T EST.AWORD    ( missing characters)
                        T EST.NOTHING  ( unmatched characters)
                        T EST.AB.WORD  ( too many qualifiers)
```

- When one or more asterisk (*) mask characters occur at the end of a qualifier in the $PREFIX value, they match the same number of or fewer characters in the corresponding qualifier of the resource name.

```
$PREFIX value           P ortion of Resource Name

TEST.NAME***  ma tches  T EST.NAME
                        T EST.NAME123
              bu t not  T EST.NAM        (to o short)
                        T EST.NAMX       (different character)
                        T EST.NAME1234   (to o long)
                        T EST.NAME.2     (ex tra qualifier)
```

- When one or more asterisk (*) mask characters form a separate qualifier in the $PREFIX value, they match a qualifier with the same number of or fewer characters in the resource name.

```
$PREFIX value           Portion of Resource Name

TEST.***.LAST  ma tches  TE ST.A.LAST
                         TE ST.ABC.LAST
               bu t not  TE ST.A       (m issing last qualifier)
                         TE ST.LAST  ( missing middle qualifier)
                         TE ST.ABCD.LAST (qualifier too long)
                         TE ST.A.B.LAST  ( too many qualifiers)
```

Extended Resource Key Masks

You can use the asterisk (*) and dash (-) to specify the rsrcmask rule entry parameter.

Using the Asterisk

Follow these guidelines when using the asterisk (*) to mask the resource rule line extended key:

- You can not use the asterisk (*) character to mask the period delimiter in a qualified resource name in the extended resource key. You must specify period delimiters; they designate the end of one qualifier and the start of another.

```
Extended resource key      Portion of Resource Name

TEST.NAME***  ma  tches     TEST.NAME
                           TEST.NAME123
              bu  t not    TEST.NAME.2   (  extra qualifier)
```

- When one or more asterisk (*) mask characters occur at the beginning of or between the characters of a qualifier in the extended key, each asterisk matches exactly one character in the corresponding qualifier of the resource name.

```
Extended resource key      Portion of Resource Name

TEST.***WORD  ma  tches    TEST.THEWORD
                           TEST.HISWORD
              bu  t not    TEST.WORD    (mi ssing characters)
                           TEST.AWORD    (missing characters)
                           TEST.NOTHING  (un matched characters)
                           TEST.AB.WORD  (to o many qualifiers)
```

- When one or more asterisk (*) mask characters occur at the end of a qualifier in the extended key, they match the same number of or fewer characters in the corresponding qualifier in the resource name.

```
Extended resource key      Portion of Resource Name

TEST.NAME***  ma tches     TEST.NAME
                           TEST.NAME123
              bu t not     TEST.NAM      (t oo short)
                           TEST.NAMX     (d ifferent character)
                           TEST.NAME1234 (t oo long)
                           TEST.NAME.2   (e xtra qualifier)
```

- When one or more asterisk (*) mask characters form a separate qualifier in the extended key, they match a qualifier with the same number of or fewer characters in the resource name.

```
Extended resource key   P ortion of Resource Name

TEST.***.LAST  ma tches  T EST.A.LAST
                         T EST.ABC.LAST
               b  ut not T EST.A          (missing last qualifier)
                         T EST.LAST       (missing middle qualifier)
                         T EST.ABCD.LAST  (qu alifier too long)
                         T EST.A.B.LAST   (to o many qualifiers)
```

Using the Dash

You can use the dash (-) masking character to specify an extended resource key mask as follows:

- When the dash (-) mask character is at the end of a qualifier in the extended key, it matches any number of characters that complete the corresponding qualifier in the resource name.

```
Extended resource key     Portion of Resource Name

TEST.NAME-    ma tches    TEST.NAME
                          TEST.NAME123
              b  ut not   TEST.NAM       (t oo short)
                          TEST.NAMX      (d ifferent character)
                          TEST.NAME.2    (e xtra qualifier)
```

- When the dash (-) mask character forms a separate qualifier in the extended key, it matches zero or more qualifiers in the resource name until the next qualifier of the extended key (if any) matches a qualifier in the resource name.

```
Extended resource key   P ortion of Resource Name

TEST.-        mat ches  T EST
                        T EST.ABC
                        T EST.ABC.DEF

TEST.-.LAST   mat ches  T EST.LAST
                        T EST.A.LAST
                        T EST.ABC.LAST
                        T EST.A.B.LAST
              bu t not  T EST.A          (missing last qualifier)
                        T EST.LAST.LAST  (ma tches first two qualifiers)
```

- When the dash (-) character occurs at the beginning of or between the characters of a qualifier in the extended key, it is treated as a literal character.

```
Extended resource key     Portion of Resource Name

TEST.A-WORD   ma tches    TEST.A-WORD
```

Using an Asterisk Followed by a Dash

An extended resource key mask that contains an asterisk followed by a dash must fit one of the following:

- When the asterisk and dash (*-) mask characters occur together at the end of a qualifier in the extended key, they function the same as the dash (-) multiple character mask; that is, they match any number of characters that complete the corresponding qualifier in the resource name.

```
Extended resource key     Portion of Resource Name

TEST.NAME*-   ma tches    TEST.NAME
                          TEST.NAME123
              b  ut not   TEST.NAM        (t oo short)
                          TEST.NAMX      (d ifferent character)
                          TEST.NAME.2     (e xtra qualifier)
```

- When the asterisk and dash (*-) mask characters form a separate qualifier in the extended key, they match exactly one qualifier of at least one character in the resource name.

```
Extended resource key     Portion of Resource Name

TEST.*-       ma tches    TEST.A
                          TEST.ABC
              b  ut not   TEST           (mi ssing last qualifier)
                          TEST.ABC.DEF   (to o many qualifiers)

TEST.*-.LAST  ma tches    TEST.A.LAST
                          TEST.ABC.LAST
                          TEST.LAST.LAST
              b  ut not   TEST.A         (mi ssing last qualifier)
                          TEST.LAST      (mi ssing middle qualifier)
                          TEST.A.B.LAST  (to o many qualifiers)
```

Use of NEXTKEY in Resource Rules

The NEXTKEY parameter directs eTrust CA-ACF2 to evaluate an alternate resource rule set when a particular environment applies to the access, but the access is prevented. Remember that eTrust CA-ACF2 validation is directed to the rule set specified in the NEXTKEY option **only** when access based on the current rule entry is prevented. Validation of the access continues with the evaluation of the alternate resource rule set.

You can build a maximum chain of 25 NEXTKEYs. When you specify more than 25, eTrust CA-ACF2 denies access and writes an SMF logging record of the event. The ACFRPTRV report displays the error condition and the $KEYs of all the rules that eTrust CA-ACF2 checked.

When using NEXTKEY, you must ensure that looping is avoided. Looping in NEXTKEY processing occurs when a NEXTKEY parameter is interpreted more than once during a single access validation. eTrust CA-ACF2 issues an error message when a loop condition occurs. It also denies the access request and logs the event. The ACFRPTRV report displays the error condition and the $KEYs of all the rules that eTrust CA-ACF2 checked.

You can use the NEXTKEY feature to divide a particular rule set. You might divide a rule set if it is very large (and you have not enabled the RULELONG parameter of the RULEOPTS GSO record), or to delegate rule maintenance (%CHANGE or %RCHANGE) authority. An alternative to using NEXTKEY to divide a large rule set is to increase the size of the eTrust CA-ACF2 access Rule database. The default size of a eTrust CA-ACF2 access rule record is 4K (4096 bytes), but your site can increase the size up to 32K (32000 bytes). For more information, see eTrust CA-ACF2 Rule Option Specifications (RULEOPTS) in the "Maintaining Global System Options Records" chapter. Also see the Getting Started guide for valuable information and considerations for increasing the size of the eTrust CA-ACF2 access Rule and Infostorage databases.

Merging Rule Sets

The following sample rule sets show how to use NEXTKEY to merge multiple rule sets. The first four rule sets describe an access environment for resources named TEST01, TEST02, TEST03, and TEST25 (assume rules for TEST04 through TEST24 also exist):

```
$KEY(test01)
 UID (tfinpaynlt) NEXTKEY(testxx)

$KEY(test02)
 UID (tfinpaynlt) NEXTKEY(testxx)

$KEY(test03)
 UID (tfinpaynlt) NEXTKEY(testxx)

$KEY(test25)
 NEX TKEY(testxx)
```

Although none of these rule sets allow access to users whose UIDs match the TFINPAYNLT mask, you can create a rule set that gives access to all of the previous resources even though the resources have different names. Use the NEXTKEY parameter to direct eTrust CA-ACF2 evaluation to a single resource rule, TESTXX. Write the rule set for this as follows:

```
$KEY(testxx) TYPE(xxx)
 UID (tfinpaynlt) ALLOW
```

Dividing Rule Sets

You can use NEXTKEY to divide a particular resource rule set if it is very large.

For example, suppose a site has the following DFSMS resources, all with the high-level qualifier of STGADMIN:

```
STGADMIN.IGD.ACTVIATE.CONFIGURATION
STGADMIN.IDC.DIAGNOSE.CATALOG
```

A single rule set for all of these resources would be very large. You can divide the rule set for the high-level qualifier STGADMIN into smaller rule sets using the second-level qualifiers and NEXTKEY as follows:

```
$KEY(STGADMIN)
 IGD .- NEXTKEY(IGD)
 IDC .- NEXTKEY(IDC)
 IGG .- NEXTKEY(IGG)
 ADR .- NEXTKEY(ADR)
```

In the previous rule set, the NEXTKEY parameter specifies the alternate rule sets to use in validating access to the specific resources. You can write smaller rule sets as follows:

```
$KEY(IGD)
$PREFIX(STGADMIN.IGD)
%CHANGE tfinpaydir
 ACT IVATE.- UID(tfinpayc1) ALLOW
 - U ID(tfinpayc2) ALLOW

$KEY(IDC)
$PREFIX(STGADMIN.IDC)
%RCHANGE tfinopsdir
 DIA GNOSE.- UID(tfinopr) ALLOW
```

The modified rule sets in the previous examples are smaller than the single rule set for the high-level qualifier STGADMIN.

Delegating Change Authority

NEXTKEY also permits you to delegate %CHANGE and %RCHANGE authority for a particular rule set.

In the previous example, you know that TFINPAYDIR has the authority to change the rule set for the STGADMIN.IGD resources because the %CHANGE control statement specifies TFINPAYDIR. TFINOPSDIR has restricted authority to change only the rule entries for the STGADMIN.IDC resources because the %RCHANGE control statement specifies TFINOPSDIR.

NEXTKEY and $PREFIX

When you use NEXTKEY in resource rules with extended resource keys, the $PREFIX control statement in the target resource rule specifies the portion of the resource name matched during NEXTKEY processing; the remainder is matched against the extended keys on the resource rule lines.

```
Resource name: TEST.TESTNAME2.TESTNAME3

Resource rule:

$KEY(TEST) TYPE(ttt)               f irst validation
TESTNAME2.- UID(-) PREVENT NEXTKEY(TESTNAME2)
 ...

$KEY(TESTNAME2) TYPE(ttt)         N EXTKEY validation
$PREFIX(TEST.TESTNAME2)            m atched two levels
TESTNAME3  U ID(-) ALLOW          t hird level validation
```

When you use NEXTKEY to select another resource rule for validation, and you do not specify a $PREFIX control statement in the alternative resource rule, eTrust CA-ACF2 processes the alternative rule as if it had been selected using a full resource key match with no portion of the resource name remaining to be matched against any rule line extended keys.

```
Resource name: TEST.TESTNAME2

Resource rule:

$KEY(TEST) TYPE(ttt)               f irst validation
TESTNAME2 UID(-) PREVENT NEXTKEY(TESTNAME3)

$KEY(TESTNAME3) TYPE(ttt)         N EXTKEY validation
UID(-) ALLOW                      n o extended key
```

eTrust CA-ACF2 Resource Rule Validation

This section describes:

- How resource rule records are selected for validation
- How resource rule lines are selected for validation
- Forcing rule validation with RSRCVLD

How Resource Rule Records Are Selected for Validation

If you mask resource rule $KEY values, remember that eTrust CA-ACF2 considers a rule less specific if a mask character appears earlier in its resource name than in another. For example, $KEY(TRANS*123) is less specific than $KEY(TRANS.***) because its masking character occurs earlier in the rule key.

When validating a particular resource name, eTrust CA-ACF2 searches for the resource rule whose $KEY value is the most specific match for the resource name. It matches the $KEY values of the resource rules to the resource name (or qualifier, as described in the following) from left to right, character by character, starting with the first character.

If the resource name being validated is 40 characters or fewer, eTrust CA-ACF2 first searches for the generalized resource rule whose $KEY value most specifically matches the full resource name of the resource being validated. When eTrust CA-ACF2 finds a rule that matches (directly or with masking) the full resource name, it uses that rule for the validation. When no generalized resource rule key matches the full resource name and the resource name is a qualified resource name, eTrust CA-ACF2 searches for the resource rule whose $KEY most specifically matches the first qualifier of the resource name. When it finds a rule that matches (directly or with masking) the first qualifier, it uses that rule for the validation. See the following example.

```
Resource name:  TEST.TESTNAME2

Sample resource rules:

$KEY(**************) TYPE(ttt)      full key match
 UID(...) ALLOW

$KEY(TEST) TYPE(ttt)                qualifier match
 TESTNAME2 UID(...) ALLOW
```

Note: If you use a fully masked resource rule $KEY value as a catch-all rule, and you also use resource rules with qualifier $KEY values, remember that eTrust CA-ACF2 searches first for the generalized resource rule whose $KEY value matches the full resource name of the resource being validated. If eTrust CA-ACF2 finds a rule that matches (directly or with masking) the full resource name, it uses that rule for the validation and does not search for a match using the first qualifier of the resource name. In the previous example, if both resource rules exist, eTrust CA-ACF2 will find and use the fully masked resource rule with $KEY(**************), and will not use the resource rule with the qualifier $KEY(TEST). In this situation, in order to use a specific rule, you need to create a rule that does not use the qualifier $KEY value, as in the following examples.

Sample resource rules:

```
$KEY(TEST.TESTNAME2) TYPE(ttt)          full key match
UID(...) ALLOW                          rule lines without extended keys

$KEY(TEST.*********) TYPE(ttt)          full key match with mask
$PREFIX(TEST)                           qualifier prefix
TESTNAME2 UID(...) ALLOW                rule lines with extended keys
```

If the resource name is longer than forty characters, it must be a qualified resource name. eTrust CA-ACF2 searches for the resource rule whose $KEY most specifically matches the first qualifier of the resource name. When it finds a rule that matches (directly or with masking) the first qualifier, it uses that rule for the validation.

```
Resource name:  TEST.NAME.THAT.IS.SUPPORTED.EVEN.THOUGH.IT.IS.VERY.LONG

Sample resource rule:

$KEY(TEST) TYPE(ttt)
 NAME.- UID(...) ALLOW
```

Resource names specified in the NEXTKEY parameter of a resource rule line are exceptions to this selection process. For NEXTKEY resource names, eTrust CA-ACF2 only searches for the resource rule whose $KEY most specifically matches the **full resource name** specified in the NEXTKEY parameter. When eTrust CA-ACF2 finds a rule that matches (directly or with masking) the full NEXTKEY resource name, it uses that rule for the validation. When it finds no rule that matches the full NEXTKEY resource name, it denies access to the resource. It performs no search for a match on the qualifier.

```
Resource name:  TEST.TESTNAME2

Sample resource rules:

$KEY(TEST) TYPE(ttt)                                    first validation
 TESTNAME2 UID(-) PREVENT NEXTKEY(GROUP.TESTNAME2)

$KEY(GROUP.TESTNAME2) TYPE(ttt)                         NEXTKEY validation
$PREFIX(****.TESTNAME2)
 UID(-) ALLOW
```

When eTrust CA-ACF2 finds an invalid resource name or finds no matching resource rule, it denies access to the resource.

How Resource Rule Lines Are Selected for Validation

When a resource rule set is compiled, the rule compiler converts the rule set input into a form usable by the rule interpreter during verification checking. Unless a $NOSORT control statement is specified, the compiler orders the rules using these criteria:

1. Extended resource key parameters from most specific to least specific with "not specified" first.
2. UID patterns from most specific to most general.
3. SOURCE parameters in alphabetical order with "not specified" last.
4. SHIFT parameters in alphabetical order with "not specified" last.
5. UNTIL dates from earliest to latest.
6. ACTIVE date.

After it selects a resource rule record to use in validating a resource name, eTrust CA-ACF2 scans the resource rule lines looking for the first rule line whose parameters or **environment criteria** match the environment of the resource access.

The first rule line with environment criteria that matches the actual access attempt determines whether access is granted. When the access decision is PREVENT, a NEXTKEY specification in the rule line might cause evaluation of another resource record for the resource access.

When no rule lines match the environment criteria for the resource access, access to the resource is denied even when the global mode is RULE. (Mode is set in the MODE field of the GSO OPTS record.)

If the resource access is denied, resource grouping for this resource type might cause further evaluation of another resource record for the access.

Extended Resource Key

The first environment check for a resource rule line is for the extended resource key. This check is affected by whether the resource record was selected because of a $KEY match on the full resource name or a single qualifier, and by whether the record contains a $PREFIX statement. These factors determine how much, if any, of the resource name remains to be matched by extended resource keys.

The simplest and most common case is when eTrust CA-ACF2 selects a resource record with a full key match with no $PREFIX statement and no extended resource keys. In this case, eTrust CA-ACF2 skips the check for the extended resource, and the environment checks continue with the UID check.

```
Resource name:  TEST

Sample resource rule:

$KEY(TEST) TYPE(ttt)
 UID(...) ...
```

When the resource record contains no $PREFIX statement and eTrust CA-ACF2 selects the record because of a $KEY match on the full resource name, no part of the resource name remains to be matched. When eTrust CA-ACF2 selects the record because of a $KEY match on the first qualifier of the resource name, the rest of the resource name after the period delimiter remains to be matched against any extended resource keys.

```
Resource name:  TEST.TESTNAME2

Possible resource rules:

$KEY(TEST.TESTNAME2) TYPE(ttt)      If full key match
 UID(...) ...

$KEY(TEST) TYPE(ttt)                If qualifier match
 TESTNAME2 UID(...) ...
```

When the resource record contains a $PREFIX statement, regardless of whether eTrust CA-ACF2 selects the record using a full key or a qualifier match, eTrust CA-ACF2 matches the full resource name against the $PREFIX value. The $PREFIX value might match the full resource name or some portion of the resource name. This determines how much of the resource name remains to be matched against any extended resource keys.

```
Resource name: TEST.TESTNAME2

Resource rule:

$KEY(TEST) TYPE(ttt)            qualifier match
$PREFIX(TEST.*********)         matches full resource name
UID(...) ...                    no extended resource key
```

After eTrust CA-ACF2 selects the resource rule record and performs any $PREFIX processing, the entire resource name has been matched or some portion of the resource name remains to be matched against the rule line extended keys.

If the full resource name has been matched, only two kinds of lines satisfy the extended key criterion:

- Rule lines with no extended key specified
- Rule lines with only a "–" full mask character (matches any or no characters) specified as the extended resource key.

Since no more of the resource name remains to be matched, any other rule line with an extended resource key specified does not satisfy the extended key environment check. Note in the following example that rule lines with no extended resource key are considered more specific than rule lines with only a dash (-) full mask character.

```
Resource name:  TEST

Resource rules:

$KEY(TEST) TYPE(ttt)
UID (...) ...                          matching rule line
TES TNAME2 UID(...) ...
-         UI D(...) ...

$KEY(TEST) TYPE(ttt)
TES TNAME2 UID(...) ...
-         UI D(...) ...                matching rule line
```

When some portion of the resource name remains to be matched against the rule line extended keys, eTrust CA-ACF2 matches that portion against the resource extended key (if any) specified on the rule line. Rule lines with no extended key or with an extended key that does not match do not satisfy the extended resource key environment check.

```
Resource name: TEST.TESTNAME2

Resource rules:

$KEY(TEST) TYPE(ttt)
UID (...) ...
TES TNAME2 UID(...) ...              matching rule line
-         UI D(...) ...
```

When the extended resource key environment check is satisfied, eTrust CA-ACF2 continues with the next environment check. Otherwise, the rule line is not considered for validation, and eTrust CA-ACF2 skips to the next rule line for evaluation.

UID

When the UID of the user attempting the resource access matches the UID mask specified in the rule line, eTrust CA-ACF2 continues with the next environment check. Otherwise, the rule line is not considered for validation, and eTrust CA-ACF2 skips to the next rule line for evaluation.

Source

If you do not specify a SOURCE value in the rule line, or if the SOURCE of the actual access matches either the source specified or one of the members of the source group specified the source environment check is satisfied, and CA-ACF2 continues with the next environment check. Otherwise, the rule line is not considered for validation, and CA-ACF2 skips to the next rule line for evaluation.

Shift

If you do not specify a SHIFT value in the rule line, or if the time of the resource access matches the time specified in the SHIFT record named in the rule line, the shift environment check is satisfied, and eTrust CA-ACF2 continues with the next environment check. Otherwise, the rule line is not considered for validation, and eTrust CA-ACF2 skips to the next rule line for evaluation.

Note: If the shift record does not match the current day, date or time the access request is being processed, the rule line is not considered for validation and CA-ACF2 skips to the next rule line for evaluation.

Service

If you do not specify a SERVICE value in the rule line, or if the type of resource access attempted matches the value of the SERVICE parameter specified in the rule line, the service environment check is satisfied, and eTrust CA-ACF2 continues with the next environment check. Otherwise, the rule line is not considered for validation, and eTrust CA-ACF2 skips to the next rule line for evaluation.

Dates

When you specify any date parameters (UNTIL, FOR, or ACTIVE) in the rule line, eTrust CA-ACF2 evaluates them to determine whether the date of the resource access satisfies the date specification. If you do not specify any date parameters or if the date of the resource access satisfies the date specifications, the date environment checks are satisfied, and eTrust CA-ACF2 continues with the next environment check. Otherwise, the rule line is not considered for validation, and eTrust CA-ACF2 skips to the next rule line for evaluation.

Record-Level Protection (RLP)

When you specify a RECCHECK parameter in the rule line, eTrust CA-ACF2 evaluates it to see whether the record level criteria of the resource access matches the criteria in the EXPRESSN record specified in the RECCHECK parameter. If you do not specify a RECCHECK parameter or if the resource access satisfies the EXPRESSN criteria, the RLP environment check is satisfied, and eTrust CA-ACF2 uses the rule line to make the resource access decision. Otherwise, the rule line is not considered for validation, and eTrust CA-ACF2 skips to the next rule line for evaluation.

RSRCVLD Forces Rule Validation

If you specify RSRCVLD in the logonid record for a user, eTrust CA-ACF2 grants access to a resource only if a rule exists that permits the user access to that resource.

When the RSRCVLD bit is set on a logonid that also has the SECURITY privilege, eTrust CA-ACF2 can deny that logonid access to the requested resource. After the check to see whether the user being denied is a security officer, eTrust CA-ACF2 checks to see whether RSRCVLD is set. When it is, even though the user is a security officer, eTrust CA-ACF2 denies access to the resource. The ACFRPTRV report shows a return code 16 from the resource rule interpreter.

Creating Resource Rule Sets

You must *compile* resource rules to store them in the Infostorage database, and *decompile* them to display them at your terminal. The compile process takes the human-readable text of the rule set and converts it into a machine-readable format. The decompile process converts the machine-readable code to human-readable text that you can view on your terminal.

You create resource rule sets directly from the terminal or by first building the rule set text in a file, most commonly as a member of a partitioned data set (PDS).

Compiling at the Terminal

To compile at the terminal, follow this procedure:

1. From TSO READY mode, issue the following commands:

```
acf
 ACF
set resource(typ)
 RESOURCE
compile
 ACF70010 ACF COMPILER ENTERED
```

   The COMPILE subcommand lets you enter the control statements and rule entries at the terminal. Follow this sequence:

   - Type the $KEY and $TYPE control statements first. Press ENTER.
   - Type each additional control statement on a separate line beginning in column two. Press ENTER.
   - Type each rule entry on a separate line, one line at a time. Press ENTER.
   - If a rule entry is longer than 72 characters, type a dash at the end of the line to indicate that the entry continues on the next line.
   - Press ENTER when you reach the end of each rule entry.

```
acf
 ACF
set resource(ckc)
 RESOURCE
compile
 ACF70010 ACF COMPILER ENTERED

 .$key(trana) type(ckc)
 .  uid(pay-) allow
 .
```

2. Type **END** and press ENTER (or hit ENTER twice) to signal the end of the rule.

3. Enter the **TEST** subcommand to verify that the rule works the way you expect. The TEST subcommand performs a dummy validation without writing an SMF record. The output shows what actually happens when you store the rule. The TEST subcommand checks the test rule against the current rules in a directory; you might have to store the rule and rebuild the directory before testing to get accurate results for the test.
4. Enter **STORE** to save the resource rule set.
5. For globally resident rule sets, enter the following command from the operator console or under the ACF command to rebuild the directory of resident resource rules:

   ```
   F ACF2,REBUILD(typecode)
   ```

   For more information about resident rules, see the description of the GSO INFODIR record in Infostorage Rule Directories (INFODIR) in the "Maintaining Global System Options Records" chapter. Also, see the *Systems Programmer Guide* for a description of the REBUILD operator command.

Compiling from a Partitioned Data Set (PDS)

To compile from a PDS, follow this procedure:

1. Use an editor to create the text of a rule in a PDS.

   ```
   $KEY(TRANA) TYPE(CKC)
    UID(PAY-) ALLOW
   ```

   Follow this sequence:

   - Enter the $KEY and $TYPE control statements first. Press ENTER.
   - Enter each control statement on a separate line. Press ENTER.
   - Enter each rule entry on a separate line, one line at a time. Press ENTER.
   - When a rule entry is longer than 72 characters, end the line with a dash to indicate that the entry continues on the next line.
2. Save this member.
3. Enter the ACF command from TSO READY, then enter:

   ```
   COMPILE dsn
   ```

   Where *dsn* is the name of the data set that contains the rule set text. eTrust CA-ACF2 stores the input from a PDS to the compiler. For example:

   ```
   acf
    ACF
   set resource(ckc)
    RESOURCE
   compile work.text(rule)
   ```

Specify the PDS name according to TSO conventions. Your high-level index is assumed unless you specify the entire PDS name and enclose it in single quotes: 'PAYTRAN.TEXT(RULE)'. Otherwise simply specify PAYTRAN.TEXT. To compile all members of a PDS, specify PAYTRAN.TEXT ALL. This automatically stores the members.

4. Enter the TEST subcommand to verify that the rule works the way you expect. The TEST subcommand performs a dummy validation without writing an SMF record. The output shows what actually happens when you store the rule. The TEST subcommand checks the test rule against the current rules in a directory; you might have to store the rule and rebuild the directory before testing to get accurate results for the test.
5. Type STORE and press ENTER to save the rule set.
6. For globally resident rule sets, enter the following command from the operator console or under the ACF command to rebuild the directory of resident resource rules:

   ```
   F ACF2,REBUILD(typecode)
   ```

   For more information about resident rules, see the description of the GSO INFODIR record in Infostorage Rule Directories (INFODIR) in the "Maintaining Global System Options Records" chapter. Also, see the *Systems Programmer Guide* for a description of the REBUILD command.

Using the ISPF Panels

Select option 1 RULES from the eTrust CA-ACF2 ISPF Option Selection Panel.

```
----------------- eTrust CA-ACF2 ISPF OPTION SELECTION MENU ------------------
OPTION  ===>

   1  RULES     - PROCESS eTrust CA-ACF2 ACCESS AND GENERALIZED RESOURCE RULES
   2  LOGONIDS  - eTrust CA-ACF2 LOGONID CREATION/MAINTENANCE FACILITY
   3  SYSTEM    - eTrust CA-ACF2 SHOW COMMANDS
   4  REPORTS   - eTrust CA-ACF2 REPORT PROGRAM PROCESSOR
   5  UTILITIES - PROCESS eTrust CA-ACF2 UTILITIES
   6  GSO       - GLOBAL SYSTEM OPTIONS SERVICES
   7  NET       - NETWORKING SYSTEM OPTIONS SERVICES
   8  CAC       - MVS DATABASE CACHE RECORD SERVICES
   9  XREF      - SOURCE/RESOURCE AND GROUP RECORD SERVICES
  10  MAC       - MANDATORY ACCESS CONTROL ADMINISTRATION
  11  CPF       - COMMAND PROPAGATION FACILITY SERVICES
  12  FIELD     - RECORD LEVEL PROTECTION CONTROLS
  13  TARGETS   - SET CPF TARGET NODES, DEFAULTS IN USE
  14  PROFILE   - PROCESS PROFILE INFORMATION RECORDS
  15  SMS       - PROCESS DFSMS SUPPORT RECORDS
  16  ENTRY     - PROCESS ENTRY SOURCE RECORDS
  17  SHIFT     - PROCESS SHIFT/ENTRY RECORDS
  18  RACDCERT  - PROCESS KEYRING/CERTIFICATE COMMANDS
  19  C-CIC     - PROCESS C-CIC INITIALIZATION RECORDS
  20  LDS       - PROCESS LDAP DIRECTORY SERVICES
```

The eTrust CA-ACF2 Rules Processor Menu is displayed:

```
---------------- eTrust CA-ACF2 SECURITY RULES PROCESSOR MENU -----------------
OPTION  ===>

    1  ACFNRULE- NEW ACCESS RULE UTILITY

    2  ACFRULES- ACCESS/GENERALIZED RESOURCE RULE PROCESSOR

    3  ACFTEST - ACCESS RULE TEST FACILITY

    4  ACFTESTR- GENERALIZED RESOURCE RULE TEST FACILITY

    5  ACFDCMP - RULE DECOMPILER

    6  ACFNRSCR- NEW GENERALIZED RESOURCE RULE UTILITY

    7  ACFRULCU- RULE CLEANUP UTILITY

    8  ACCESS  - ACCESS COMMAND
```

Panel Field Descriptions

Choose the option for the function you want to perform:

1 ACFNRULE
Lets you create or delete individual rules. You can locate multiple rules that contain the same character string, then verify and delete them. The "Maintaining Access Rules" chapter describes this panel.

2 ACFRULES
Lets you create, display, and delete access and resource rules. This chapter discusses resource rules. For information about creating, displaying, and deleting access rules, see "Maintaining Access Rules."

3 ACFTEST
Lets you test access rules. The "Maintaining Access Rules" chapter describes this panel. The TEST subcommand checks the test rule against the current rules in a directory; you might have to store the rule and rebuild the directory before testing to get accurate results for the test.

4 ACFTESTR
Lets you test resource rules. The TEST subcommand checks the test rule against the current rules in a directory; you might have to store the rule and rebuild the directory before testing to get accurate results for the test.

5 ACFDCMP
Lets you decompile rules online or from a partitioned data set (PDS).

6 ACFNRSCR
Lets you create or delete individual resource rules. You can locate multiple resource rules that contain the same character string, then verify and delete them.

7 ACFRULCU
Lets you delete outdated rules from your rules database.

8 ACCESS
Lets you display a list of users and their associated access permissions for a given resource. It also lists any users that have been completely disallowed access to the resource.

Now select the next task you want to perform from the eTrust CA-ACF2 Rules Processor Menu. The following sections describe the panels for each of these options.

Creating, Displaying, and Deleting Rules

Select option 2 ACFRULES from the eTrust CA-ACF2 Rules Processor Menu to create, view, or delete a rule. The eTrust CA-ACF2/ISPF Full Rule Processor panel is displayed. Descriptions of each field follow this sample screen:

```
------------------- eTrust CA-ACF2/ISPF FULL RULE PROCESSOR ------------------
COMMAND ===>

ACF2 RULE SET NAME: (FIRST EIGHT CHARACTERS BECOMES MEMBER NAME)
   NAME    ===>
ACF2 COMMAND MODE:
   MODE    ===>                (RULE/RESOURCE MODE - DEFAULT IS RULE MODE)
   TYPE    ===>      RESOURCE TYPE(E.G., CKC,ITR,IAG)

DECOMPILE INTO OPTION OR PDS EDIT OF EXISTING RULESET IN A PDS:
   DECOMP RULE PRIOR TO EDIT ===> YES   YES OR NO
   TEST RULE PRIOR TO STORE  ===> YES   YES OR NO
ISPF LIBRARY DATA SET (DO NOT INCLUDE MEMBER NAME):
   PROJECT ===>
   LIBRARY ===>
   TYPE    ===>
OTHER PARTITIONED DATA SET (DO NOT INCLUDE MEMBER NAME):
   DATA SET NAME  ===>

ACF2 PROCESSING OPTIONS:
   CLEAR SESSION RULES  ===>       YES OR NO    NORULELONG===>  YES OR NO
   PURGE ABOVE RULE SET ===>       YES OR NO
   FORCE RULE REPLACE   ===>       YES OR NO
```

Panel Field Descriptions

NAME
Enter the name of the high-level qualifier of the data set.

MODE
Specify RESOURCE for resource rules. RULE is the default.

TYPE
Specify the three-character type code. This field only applies to resource and eTrust CA-ACF2 for DB2 rules.

DECOMP RULE PRIOR TO EDIT
Specify YES to decompile the rule set before entering edit mode. YES is the default.

TEST RULE PRIOR TO STORE
Specify YES to TEST the rule before eTrust CA-ACF2 stores it. The TEST subcommand checks the test rule against the current rules in a directory; you might have to store the rule and rebuild the directory before testing to get accurate results for the test.

ISPF LIBRARY DATA SET (DO NOT INCLUDE MEMBER NAME)
Enter the PROJECT, LIBRARY, and TYPE fields. For example, if the rule set is stored in a PDS named ACF2.RULES.PROD(PAYROLL), enter ACF2 for PROJECT, RULES for LIBRARY, and PROD for TYPE. You do not need to enter the member name. eTrust CA-ACF2 lets you select from a list.

OTHER PARTITIONED DATA SET (DO NOT INCLUDE MEMBER NAME)
Enter the fully qualified data set name in single quotes. For example, 'ADMIN01.ACF2.RULES'.

CLEAR SESSION RULES
Enter YES to clear rules you entered during this session. NO is the default.

NORULELONG
Enter YES to compile the rule set in a short record format. NO is the default.

PURGE ABOVE RULE SET
Enter YES to purge this rule. NO is the default.

FORCE RULE REPLACE
Enter YES to force replacement of the existing rule. YES is the default.

Testing Resource Rules

Select option 4 ACFTESTR from the eTrust CA-ACF2 Rules Processor Menu to test a resource rule. The ACFTESTR - eTrust CA-ACF2 Resource Rule Test Facility panel is displayed. Descriptions of each field follow this sample screen. The TEST subcommand checks the test rule against the current rules in a directory; you might have to store the rule and rebuild the directory before testing to get accurate results for the test.

```
-------------- ACFTESTR - eTrust CA-ACF2 RESOURCE RULE TEST FACILITY ---------
COMMAND ===>

 RESOURCE RULEID NAME TO BE TESTED:
   $KEY       ===>
 RESOURCE TYPE OF RULEID:
   TYPE       ===>      (E.G., CKC, ITR, TAC, ETC.)
 EXTENDED RESOURCE NAME:
   RSRCNAME   ===>
 DECOMPILE PRIOR TO TEST:
   DECOMP     ===> NO   YES/NO

 LOGONID OR UID STRING OF THE USER THAT THE RULE IS TO BE TESTED AGAINST:
   LOGONID    ===>                UID         ===>

 ADDITIONAL PARAMETERS TO BE ASSOCIATED WITH THE TEST: OPTIONAL
   TIME           ===>             hhmm
   INPUT SOURCE   ===>             source-id
   UNTIL DATE     ===>             mm/dd/yy
   SERVICE        ===>                          read,add,update,delete
   NOPREFIX       ===>             YES/NO
        ------------- ACFTESTR OUTPUT DISPLAY AREA --------------------
```

Panel Field Descriptions

$KEY
Enter the one to 40-character $KEY control statement of the rule you want tested. This is a required field.

TYPE
Enter the three-character type code of the resource you want tested.

RSRCNAME
Enter the name of the resource you want to test (maximum 40 characters). eTrust CA-ACF2 places the $KEY value before the value you specify for RSRCNAME unless the value you specify is in single quotes. If the rule contains a $PREFIX control card, and the $PREFIX value is not masked, the $PREFIX value is used instead of the $KEY value.

NOPREFIX
When NOPREFIX is specified, the $KEY will be used instead of the $PREFIX when building the fully qualified resource name to be used in the test.

DECOMP
Specify YES to view the rule set before testing. NO is the default.

LOGONID
Enter the logonid of the user you want tested. eTrust CA-ACF2 will automatically go to the logonid database to construct the UID string for the logonid using it in the validation test.

UID
Enter the complete UID string for the user you want tested. Any masking characters used in the input of this field are taken as literal. This is a required field if no LOGONID is to be specified.

TIME
Enter a time of day you want tested. Specify the time in *hh:mm*.

INPUT SOURCE
Enter the name of a source, a source group, or a mask that you want tested.

UNTIL DATE
Enter a date you want access tested. The format must be the same as specified in the GSO OPTS record.

SERVICE
Enter the accesses you specified in this rule entry. Use commas or blanks to separate multiple entries.

Note: In using the TEST command masking is not allowed in any of the fields. Any masked character specified for input is taken as a literal. Also, resource rule testing may also give unpredictable results if there exists a resident directory for the type code being tested and the stored rule was not followed by a rebuild of that directory.

Displaying Rules

Select option 5 ACFDCMP from the eTrust CA-ACF2 Rules Processor Menu to decompile (view) a rule at the terminal or to place it in a PDS. The eTrust CA-ACF2/ISPF Decompile Rule Processor panel is displayed. Descriptions of each field follow this sample screen:

```
------------eTrust CA-ACF2 SECURITY /ISPF DECOMPILE RULE PROCESSOR -------------
COMMAND ===>

 eTrust CA-ACF2 COMMAND MODE:
   MODE    ===> RULE           RULE/RESOURCE MODE  (RULE MODE)
   DISPLAY ===>                VERBOSE/TERSE DISPLAY  (VERBOSE DISPLAY)
   TYPE    ===>                RESOURCE TYPE E.G., CKC, ITR, IAG

 eTrust CA-ACF2 RULE SET NAME:
   $KEY    ===> LAGLO01
   LIKEKEY ===>
   DELETE  ===> NO   YES OR NO (NO)
   ALL     ===> NO   DECOMP ALL NEXTKEYS UNDER THIS $KEY (YES OR NO)
```

Panel Field Descriptions

MODE
Specify RESOURCE. RULE is the default.

DISPLAY
Specify nothing to view the entire rule. Specify TERSE to view only the date and the user who stored the rule.

TYPE
Specify the three-character resource type code. TYPE only applies to resource and eTrust CA-ACF2 for DB2 rules.

$KEY
Specify the one to 40-character name of the resource you want to view. Specify only the high-level qualifier for rules with extended resource keys.

LIKEKEY
Specify a mask to display a group of rules.

DELETE
Specify YES to erase the rules from the database after you view them. NO is the default.

ALL
Specify YES to decomp the rule set and all NEXTKEY rule sets found under it. ALL and LIKEKEY are mutually exclusive keywords. NO is the default.

Displaying Access

To list users and their associated access permissions for a given resource, select Option 8 ACCESS from the eTrust CA-ACF2 Rules Processor Menu. It will also list any users that have been completely disallowed access to the resource.

```
-------------- eTrust CA-ACF2/ISPF ACCESS COMMAND PROCESSOR ----------------
COMMAND ===>

 eTrust CA-ACF2 SECURITY ACCESS COMMAND PARAMETERS:
   DATASET NAME  ===> RULE
   RESOURCE NAME ===>
   TYPE    ===>                RESOURCE TYPE E.G., CKC, ITR, TBL
   CLASS   ===>                RESOURCE CLASS E.G., D, R (DEFAULT)
   SYSID   ===>                DB2 SYSID
```

Creating Resource Rules

Select option 6 ACFNRSCR from the eTrust CA-ACF2 Rules Processor Menu to create a resource rule. The ACFNRSCR - eTrust CA-ACF2 New Generalized Resource Rule Utility panel is displayed. Descriptions of each field follow this sample screen:

```
------- ACFNRSCR - eTrust CA-ACF2 NEW GENERALIZED RESOURCE RULE UTILITY ------
COMMAND ===>

eTrust CA-ACF2 GENERALIZED RESOURCE RULE
              $KEY ===>
              TYPE ===>                              SHORT RECORD FORMAT
                                                     NORULELONG ===>  (Y OR N)
eTrust CA-ACF2 GENERALIZED RESOURCE RULE OPTIONS

     ADD OR DELETE ===> ADD (ADD OR DELETE, DEFAULT IS ADD)
           RSRCNAME ===>
   FOR: UID STRING ===>
   -or-  LOGONID   ===>

   SERVICE:  READ  ===>   UPDATE ===>       ADD ===>                  (Y OR N)
             EXEC  ===>   DELETE ===>
    ACCESS: ALLOW  ===>      LOG ===>   PREVENT ===>   VERIFY ===>   (Y OR N)

         RULE DATA ===>
             SHIFT ===>
            SOURCE ===>
   EXPIRATION DATE ===>           - OR - FOR A NUMBER OF DAYS ===>
       ACTIVE DATE ===>

ENTER END TO CANCEL OR ENTER TO PROCESS REQUEST.
```

Panel Field Descriptions

KEY
Specify the one to 40-character name of the resource to which this rule applies. You can use the asterisk (*) to mask the resource name.

TYPE
Specify the three-character type code of the resource to which this rule applies.

NORULELONG
Enter YES to compile the rule set in a short record format. NO is the default.

ADD OR DELETE
Specify ADD to create a rule or DELETE to delete an existing rule. ADD is the default.

RSRCNAME
Specify the extended name.

UID STRING
Specify the UID of the users for which this rule applies. Do not specify both the UID and LID fields.

LOGONID
Specify the LID of the users for which this rule applies. Do not specify both the UID and LID fields. Place the logonid in quotes if trailing blanks are to be included in the formation of the UID that includes this field; otherwise, trailing blanks are excluded.

READ
Specify Y for yes or N for no to indicate the type of access for this resource. You can specify more than one.

UPDATE
Specify Y for yes or N for no to indicate the type of access for this resource. You can specify more than one.

ADD
Specify Y for yes or N for no to indicate the type of access for this resource. You can specify more than one.

EXEC
Specify Y for yes or N for no to indicate the type of access for this resource. You can specify more than one.

DELETE
Specify Y for yes or N for no to indicate the type of access for this resource. You can specify more than one.

ALLOW
Specify Y to allow access to this resource.

LOG
Specify Y to log access to this resource.

PREVENT
Specify Y to prevent access to this resource.

VERIFY
Specify Y to force users to re-enter their password to access this resource.

RULE DATA
Specify up to 24 characters of comments to be stored with this rule.

SHIFT
Specify the name of a shift record that indicates the time period when access must occur. You cannot mask this field.

SOURCE
Specify the name of an entry record that defines permitted entry sources from which a user can access this resource.

EXPIRATION DATE
Specify the last date this rule applies. Use the format specified in the GSO OPTS record. Do not specify both EXPIRATION DATE and FOR A NUMBER OF DAYS fields.

FOR A NUMBER OF DAYS
Specify the number of days that this rule applies. Do not specify both EXPIRATION DATE and FOR A NUMBER OF DAYS fields.

ACTIVE DATE
Specify the first date this rule applies. Use the format specified in the GSO OPTS record.

Using the ACF Command

You process resource rules after you establish the RESOURCE setting of the ACF command with the appropriate type code.

```
ACF
SET RESOURCE(ckc)
```

After you establish the RESOURCE setting and appropriate type code, you can issue any of the following ACF subcommands:

- ACCESS
- CHKCERT
- COMPILE
- CONNECT
- DECOMP or LIST
- DELETE
- END
- EXPORT
- GENCERT
- GENREQ
- HELP
- MLSLABEL
- MLWRITE
- REKEY
- RECKEY
- REMOVE
- ROLLOVER
- SET or T
- SHOW

- SN
- STORE
- SYNCH
- TEST

The common subcommands ACCESS, CHKCERT, CONNECT, EXPORT, END, QUIT, GENCERT, GENREQ, HELP, MLSLABEL, MLWRITE, REKEY, REMOVE, ROLLOVER, SHOW, SN, and SYNCH operate under all settings. The following sections describe the function, syntax, and parameters of the other subcommands listed previously.

SET Subcommand

Use the SET RESOURCE subcommand to establish the RESOURCE setting of the ACF command. You can LIST or DECOMPILE resource rules by specifying:

```
SET RESOURCE(yyy)
```

Where yyy is the resource type. The SET RESOURCE subcommand does not control the COMPILE or STORE of resource rules.

The SET subcommand operates under the RESOURCE setting for all other uses as described in "Overview of eTrust CA-ACF2."

COMPILE Subcommand

Use the COMPILE subcommand to create a rule set. The syntax of this subcommand is:

```
COMpile  [*]
         [dsn]
         [List|NOList]
         [Store|NOStore]
         [NORulelong]
         [ALL]
```

eTrust CA-ACF2 provides two methods for compiling rule sets:

1. Directly at the terminal.
2. From a partitioned data set (PDS).

See Creating Resource Rule Sets earlier in this chapter for a discussion of these methods.

Parameter Descriptions

Use the following parameters with the COMPILE subcommand:

*** (asterisk)**
Indicates that the following text is input to the compiler. This process is called an *online compile*. In an online environment, the system prompts you to enter the access rule text directly from the terminal. In batch, the parameters following the COMPILE subcommand are assumed to be input.

(no parameters)
Indicates that the following text is input to the compiler. This is the same as specifying an asterisk.

dsn
Specifies a PDS and member name that contains the resource rule text to be compiled. The PDS name follows TSO conventions. Your high-level index is assumed unless you specify the entire PDS name and enclose it in single quotes. For example, 'PAYROLL.********.TEXT(RULE)'

If you do not specify a member name, eTrust CA-ACF2 prompts you for one. To compile input from all PDS members, specify the ALL parameter. When you specify ALL, eTrust CA-ACF2 does an automatic store. eTrust CA-ACF2 does not compile multiple rule sets from a single PDS member.

LIST|NOLIST
Causes the input to the compiler to display on your screen or print on your listing during compilation of a rule set. NOLIST causes no such display or printed list. LIST is ignored when you compile online; for online compiles, NOLIST is always in effect. LIST is the default when compiling from a PDS.

NORULELONG
Overrides the use of the rulelong compiler when RULELONG is active. Normally, eTrust CA-ACF2 uses the rulelong compiler to compile rules if the RULELONG option is set. The rulelong format is an expanded record format. If a ruleset is small and therefore does not require the rulelong format, specifying NORULELONG on a compile lets you compile a ruleset using a compact record format. This way, you can choose to compile rules with the format that is required for the ruleset.

STORE|NOSTORE
Causes the rule set to be stored at compilation time. NOSTORE means that you must issue the STORE subcommand to store the rule set. STORE is the default if you are using the ALL parameter to compile all members of a PDS. Otherwise, NOSTORE is the default.

ALL

Compiles and stores the rule sets from all the members of a specified partitioned data set (PDS).

```
acf
 ACF
set resource(cfc)
 RES OURCE
compile payt all
```

Do not specify ALL if any members of the PDS contain no rule set.

TEST Subcommand

The TEST subcommand lets you interactively test a compiled rule set. Testing determines whether the rule set allows the access you intended. When you specify the TEST subcommand, eTrust CA-ACF2 performs a validation of the compiled rule but does not create any loggings for violations. The TEST subcommand checks the test rule against the current rules in a directory; you might have to store the rule and rebuild the directory before testing to get accurate results for the test. User exits such as the RSCXIT1, RSCXIT2 and SVCIXIT can affect the outcome of the test command for resource rule testing. The syntax of the TEST subcommand is:

```
TEst [*|ruleid]
```

Suppose you compile a resource rule set with $KEY(STGADMIN) and TYPE(FAC):

```
 ACF 70010 ACF COMPILER ENTERED □

 .$k ey(stgadmin) type(fac) □
  .  igd .- uid(tfinpaynlt) service(read) allow □
  .  idc .- uid(tfinpayiso) log □
 . □
store □
 ACF 70051 TOTAL RECORD LENGTH= 162 BYTES, 3 PERCENT UTILIZED □
```

To test this rule, issue the following TEST subcommand:

```
test *
 .
```

When the period (.) appears, the TEST subcommand is active. Enter any of the TEST subcommand keywords to specify the particular environment you want tested. Unspecified keywords inherit the default value for that keyword (if it has not yet been specified) or they inherit the value specified in the previous test and retain that value until a new value is explicitly specified for that keyword.

For example, the following keywords test whether the resource rule set STGADMIN lets the user TFINPAYNLT access the SMS storage administration resources:

```
test *
 .   rsrcname(igd) uid(tfinpaynlt) service(read)
```

After you enter the TEST subcommand keywords, the system displays all of the current values that describe the environment being tested. The last two lines of the display indicate whether the access is permitted, logged, or prevented:

```
test *
.   rsrcname(igd) uid(tfinpaynlt) service(read)
 AC F71114 THE FOLLOWING PARAMETERS ARE IN EFFECT:
 DA TE=04/02/04 TIME=1445 SOURCE=******** UID=TFINPAYNLT
 SE RVICE=(READ)

 TA RGET RESOURCE: RFAC STGADMIN.IGD

 VA LIDATED RULE LINE FROM STGADMIN TYPE FAC
 IG D.- UID(TFINPAYNLT) SERVICE(READ) ALLOW

  RESULT: ACCESS WOULD BE ALLOWED
 RE ASON: RESOURCE RULE
```

This example shows that the user is permitted read authority only because a resource rule exists that defines that access.

After the result displays, you can specify other keywords and values to define another environment for testing. The END subcommand terminates the TEST subcommand.

How the Values Describing the Environment Work

The values describing the test environment remain in effect until they are specifically changed. Any values you **do not** specify are assumed to be completely masked, which means they assume the default values. For instance, when you do not specify a UID, the subcommand tests whether all UIDs are permitted access. **You must specify the RSRCNAME parameter**.

The TEST subcommand takes these parameters:

*** (asterisk)**
Indicates that you want the last explicitly referenced rule set tested.

(no parameter)
Indicates that you want the last explicitly referenced rule set tested. The TEST subcommand operates the same whether you specify no parameters or an asterisk.

ruleid
Identifies the key of the rule set you want tested. To specify a rule set by its rule ID, you must have the authority to update the rule set, the SECURITY or AUDIT privilege level, or DECOMP authority as specified in the GSO RULEOPTS record. If the rule ID ends with a dash (-), enclose the rule ID in single quotes.

TEST Subcommand Keywords

When you issue the TEST subcommand with any of these parameters, it becomes active. These fields are not maskable. Any masked character specified for input is taken as a literal. Resource rule testing may also give unpredictable results if there exists a resident directory for the type code being tested and the stored rule was not followed by a rebuild of that directory. Specify a test access environment by entering any of the following keywords with the appropriate values:

Rsrcname(*rsrcname*) – Specifies a resource name for which access is tested. eTrust CA-ACF2 places the $KEY value before the RSRC value unless you specify the RSRC value in single quotes. If the rule to be tested contains a $PREFIX control card, and the $PREFIX value is not masked, the $PREFIX value is used as the high level qualifier of the resource name whenever the RSRC value is entered without single quotes. The $KEY value will be used if the $PREFIX is masked. If the $KEY is not desired, then specify the fully qualified resource name in quotes.

NOPREFIX – When NOPREFIX is specified, the $KEY value will be used instead of the $PREFIX value when building the fully qualified resource name to be used in the test.

Lid(*logonid*)
Specifies a logonid whose access you want tested. To specify this keyword, the user making the request needs authorized access to the logonid record of the user whose access is being tested. You cannot mask the logonid being tested. Asterisks are treated as actual character values. eTrust CA-ACF2 calls the database to obtain this logonid's UID string and uses that for test purposes. eTrust CA-ACF2 considers logonid fields such as SECURITY, NON-CNCL, PREFIX, and so forth, when testing.

Uid(*uid*)
Specifies a user whose access you want tested. Asterisks are treated as actual character values. TEST interprets unspecified fields in the UID as blanks (for example, UID(*) and UID(*) are considered identical).

To specify this keyword, the user making the test does not need access to the logonid record of the user whose access is being tested. eTrust CA-ACF2 does not consider logonid fields such as SECURITY, NON-CNCL, PREFIX, and so forth, when testing this keyword. When you specify LID and UID, eTrust CA-ACF2 uses the last LID or UID value specified. For example, when you specify LID(PAYJJD) UID(TFINPAYNLT), eTrust CA-ACF2 uses UID(TFINPAYNLT). Conversely, when you specify UID(TFINPAYNLT) LID(PAYJJD), eTrust CA-ACF2 uses LID(PAYJJD).

Date(*date*)
Specifies the date for which you want access tested. This date can be in the format *mm/dd/yy*, *yy/mm/dd*, or *dd/mm/yy*, depending on the DATE field of the GSO OPTS record. The TEST subcommand uses the current date as the default.

SOurce(*sourcemask*)
Specifies the source or source group from which access is tested.

Time(*hhmm*)
Specifies the time (in hours and minutes) for which you want the access tested.

ENd
Terminates the TEST subcommand and places you back in the previous setting

SErvice(Read,Add,Update,Delete,Execute)
Specifies the type of resource access tested. This access type can be READ, ADD, UPDATE, DELETE, or EXECUTE. Separate multiple access types with blank characters or commas as delimiters. This keyword applies only to resource rules for CICS file. When you omit the SERVICE keyword, eTrust CA-ACF2 assumes ALL services. Execute access type does not work with CICS resource.

Note: The SHIFT keyword is not available when testing resource rules; use DATE or TIME instead.

TEST Subcommand Results

When a test access attempt is prevented but the attempt matches a NEXTKEY environment defined in the rule set, eTrust CA-ACF2 displays the rule ID that it actually checks:

```
set resource(ckc)
 RESOURCE
decomp 'pay-'
 ACF75052 RESOURCE RULE PAY- STORED BY USER01 ON 04/02/04-15:00
 $KEY(PAY-) TYPE(CKC)
  UID(PAY01) ALLOW
  UID(*) PREVENT
 ACF75051 TOTAL RECORD LENGTH= 174 BYTES, 4 PERCENT UTILIZED
 RESOURCE
decomp paym
 ACF75052 RESOURCE RULE PAYM STORED BY USER01 on 04/02/04-14:59
 $KEY(PAYM) TYPE(CKC)
  UID(HRS) NEXTKEY(HRS-) PREVENT
  UID(PAY) NEXTKEY(PAY-) PREVENT
  UID(*) PREVENT
 ACF75051 TOTAL RECORD LENGTH= 246 BYTES, 6 PERCENT UTILIZED
 RESOURCE
test *
 .  uid(pay01)
 ACF71114 THE FOLLOWING PARAMETERS ARE IN EFFECT:
  DATE=04/02/04 TIME=1504 SOURCE******** UID=PAY01
```

```
TARGET RESOURCE: RCKC PAYM

NEXTKEY VALUES USED IN VALIDATION:
NEXTKEY ELEMENTS: PAY-

VALIDATED RULE LINE FROM PAY- TYPE CKC
UID(PAY01) ALLOW

RESULT: ACCESS WOULD BE ALLOWED
REASON: RESOURCE RULE
```

At this point, validation of the test access attempt ends. eTrust CA-ACF2 prompts for TEST subcommand keywords and values for the next test access attempt. The END subcommand terminates the TEST subcommand.

STORE Subcommand

The STORE subcommand lets you store the previously compiled rule set.

The syntax of the STORE subcommand is:

```
STore
```

This subcommand accepts no parameters.

When SET NOFORCE is in effect, eTrust CA-ACF2 stores the rule set only if it does not already exist. A message displays when a resource rule is not stored. The NOFORCE parameter of the SET subcommand is described in the "Overview of eTrust CA-ACF2" chapter.

eTrust CA-ACF2 does not store the rule when the issuer does not have the SECURITY privilege.

DECOMP Subcommand

The DECOMP subcommand displays a previously compiled and stored rule set. Use this subcommand to examine, update, or change rule sets. You can decompile a rule set at the terminal or into a member of a partitioned data set (PDS). The LIST subcommand accomplishes the same function as DECOMP.
Note: See the "Overview of eTrust CA-ACF2" chapter for an example of the SHOW RSRCTYPE command, which displays of all R and DB2 type resource types defined.

The syntax of the DECOMP subcommand is:

```
DEComp or List  {*|ruleid|Like(ruleidmask)}
                [Into(dsn)][ALL]
```

Note: Where there are identical $MEMBER or $KEY values in Rules with different TYPE codes it is possible for a PDS member to be overwritten in a DECOMP INTO operation, for example:

```
SET R(***)
DECOMP LIKE(-) INTO(hlq.rules.backup.pds)
```

The DECOMP subcommand accepts the following parameters. You must specify one of the following with the DECOMP or LIST subcommand:

*** (asterisk)**
Indicates you want to decompile the last explicitly referenced rule set processed since establishing the RESOURCE setting.

ruleid
Specifies the $KEY of the rule set you want to decompile or list. When the rule ID ends with a dash (-), enclose the rule ID in single quotes.

Like(*ruleidmask*)
Specifies a group of rule sets you want to decompile or list.

The following parameter is optional with the DECOMP or LIST subcommand:

Into(*dsn*)
Specifies the data set into which you want the rule set decompiled. This data set must be a PDS. When you specify a fully qualified data set name, including the high-level index, enclose that name in single quotes.

```
decomp paykln into('payjsd.work.rule')
```

When you decompile rule IDs that contain imbedded blanks, enclose the rule ID in single quotes as follows:

```
decomp 'pay 777'
```

When a data set is not allocated, eTrust CA-ACF2 allocates that data set as a PDS and decompiles the rules into the PDS. When you do not specify a member name, eTrust CA-ACF2 uses the $KEY and $MEMBER values of the rule set to determine the member name. If the rule set has a $MEMBER control statement, the $MEMBER value is used as the member name. If the rule set has no $MEMBER control statement, the $KEY of the rule set is used as the member name. If the $KEY value is invalid as a member name, eTrust CA-ACF2 automatically generates a member name. For more information about the automatic generation of this member name, see the description of the SET MEMBER subcommand in "Overview of eTrust CA-ACF2."

ALL
Specifies that the rule set and all NEXTKEY rule sets found under it are decompiled. When the INTO keyword is used with the ALL keyword, all of the rule sets are decompiled into a single member. You cannot specify the ALL and LIKE keywords on the same decomp statement.

The output from the DECOMP with the ALL parameter into a data set cannot be feed back into a COMPILE from the same data set if multiple $KEY statements are present.

DELETE Subcommand

The DELETE subcommand lets you delete rule sets. Only users with the SECURITY privilege level can delete rule sets. You can restrict this authority through the use of scopes.

The syntax of the DELETE subcommand is:

```
DELete    {*|ruleid}
```

For example, the following subcommand deletes the resource rule set PAYMST:

```
acf
 ACF
set resource(ckc)
 RESOURCE
delete paymst
 ACF60018 RESOURCE PAYMST DELETED
```

After you issue the DELETE command, the message DELETED displays.

You must specify one of the following parameters with the DELETE subcommand:

*** (asterisk)**
Indicates that you want to delete the last explicitly referenced rule set.

ruleid
Specifies a rule ID that you want deleted. When the rule ID ends with a dash (-), enclose the rule ID in single quotes.

RECKEY Subcommand

The RECKEY subcommand assists security administrators in maintaining rule sets and compiled infostorage rule records. This subcommand allows the user to decompile, add or delete a rule entry, recompile, and store the updated rule set with one command. This command can be used in any ACF mode that handles compiled records and it executes on other CPF-defined nodes.

The syntax of the RECKEY subcommand is:

```
RECKEY ruleid {ADD(rule-entry)|DELETE(rule-entry)}
              [CLASS(class)]
              [TYPE(type)]
              [SYSID(sysid)]
              [LIST|NOLIST]
              [NOVERIFY|VERIFY]
```

For example, the following sequence shows how to add one additional rule line to a resource rule using the RECKEY subcommand:

```
ACF
SET RESOURCE(CKC)
RECKEY PAYM ADD(HRS UID(HRS-) ALLOW)
```

Parameter Descriptions

The RECKEY subcommand takes the following parameters:

ruleid
Specifies the key of the rule set being modified.

ADD | DELETE
Specifies the function to be performed. ADD inserts a rule line. DELETE removes a rule line.

rule-entry
Indicates the actual rule entry to be inserted or removed. If you are deleting a rule entry, you must specify the complete rule entry as displayed by the decompiler.

CLASS(*class*)
Specifies the one-character class code of the infostorage rule set to be processed. The class code for eTrust CA-ACF2 for DB2 rule sets is "D." If CLASS is specified, a TYPE parameter must also be specified. If CLASS is not specified but TYPE is, CLASS(R) is assumed.

TYPE(*type*)
Specifies the type of resource processed. If not specified, the current ACF mode setting is used.

SYSID(*sysid*)
This file is used when specifying multiple input lines to the ACFNRULE utility. The SYSIN file enables you to specify a set of parameters that exceeds 100 characters and is used only in batch. ACFNRULE can modify **one** ruleset per execution.

LIST | NOLIST
Specifies whether the rule set is listed upon completion. LIST is the default.

NOVERIFY | VERIFY
Prompts to verify deletion of rule during delete processing. NOVERIFY is the default.

Note: Masking can be used in the ADD parameter but as the first character in the string will be taken as a comment card. To specify a masking character at the beginning of a rule line using RECKEY, simply put a space as the first character in the ADD parameter.

For example, the following sequence shows how to add one additional rule line that would end up as a comment card in the rule:

```
ACF
SET RESOURCE(CKC)
RECKEY PAYM ADD(*RS UID(HRS-) ALLOW)
```

To get this the qualifier '*RS' to be taken as a mask you would want to use the following format:

```
RECKEY PAYM ADD( *RS UID(HRS-) ALLOW)
```

Chapter 8

Maintaining Entry Source and Source Group Records

The ENTRY setting of the ACF command lets you define the sources or groups of sources from which users can access the system. Entry source records (E-SRC) define individual input sources, such as terminals or card readers, for the purpose of eTrust CA-ACF2 Security validation. E-SRC records translate physical input sources to logical input sources. eTrust CA-ACF2 recognizes both. Entry source group records (E-SGP) define groups of input sources for the purpose of eTrust CA-ACF2 validation. These groups can be made up of logical or physical input source names.

After you make these records active, you can specify their record names in the SOURCE field of a user's logonid record. For more information about the SOURCE field of the logonid record, see the "Maintaining Logonid Records" chapter.

This chapter describes:

- The XREF source group record
- Example of an entry source record
- Example of an entry source group record
- Fields in entry records
- Naming source or source group records
- Using the ISPF panels to maintain entry records
- Using the ACF commands to maintain entry records
- Activating changes to E(SRC) records
- Activating changes to E(SGP) records

The XREF Source Group Record

The XREF setting provides the option of using the XREF Source Group Record (X-SGP) in place of the Entry Source Group Record (E-SGP) for grouping sources. Here are considerations:

- You can use masking characters. For example, suppose your site has four departments: payroll, personnel, operations, and programming. All source names begin with the first three letters of the department, such as PAY for payroll and PER for personnel. You could reduce the number of entries in a source group to one using a mask, for example PAY-.

  However, using masking might increase the time eTrust CA-ACF2 takes to validate an access attempt. eTrust CA-ACF2 uses a binary search to locate a source name entry in an unmasked record. However, because of the ability to use source names or masks, eTrust CA-ACF2 uses a modified sequential search to locate a source name in a masked record. If the masked record contains many source names, eTrust CA-ACF2 takes longer to process the masked record.

- You can exclude sources from a source group. For example, you might want to grant write access to users in the payroll department who log on at terminals PAY01 through PAY15, but prevent them from writing if they log on from the group terminal, PAY07. You could specify INCLUDE(PAY-) and EXCLUDE(PAY07) in the X-SGP record for the PAY source group.

- You can define source groups and sets of source group records separately.

- You can use standard eTrust CA-ACF2 scope records for X-SGP records, whereas the E-SGP records require that you use a pseudo data set name to obtain a scoping function.

Entry records that define single sources (the E-SRC records) are supported in eTrust CA-ACF2. An advantage of E-SGP records is the ability to use the NEWDATA, VERDATA, and OLDDATA parameters to change all records with a particular source entry. For example, suppose an employee moves from the payroll department to the personnel department. If his old source was named PAY57, and his new source was named PER33, you could change every instance of PAY57 to PER33 using a single CHANGE subcommand:

```
CHANGE LIKE(-) OLDDATA(PAY57) NEWDATA(PER33)
```

For more information about X-SGP records, see the “Maintaining Cross-Reference Records” chapter.

Example of an Entry Source Record

When listed, an entry source record might look like the following:

```
set entry(src)
 ENTRY
list lv133
 TYPE: SRC   ENTRY: LV133    1 DATA ITEMS
  ROOM110
```

In this example, the type code SRC identifies this record as an entry source record. LV133 is the name of the entry record, and is also the physical name that identifies a terminal. The data item, ROOM110, defines a logical name for the terminal. The user has chosen this logical name because it is more meaningful than the name LV133.

To permit a user to access the system from this terminal, specify SOURCE(ROOM110) in his logonid record.

Example of an Entry Source Group Record

When listed, an entry source group record might look like the following:

```
set entry(sgp)
 ENTRY
list payroll
 TYPE: SGP   ENTRY: PAYROLL    3 DATA ITEMS
  ROOM110
  ROOM120
  LV156
```

In this example, the type code SGP identifies this record as an entry source group record. PAYROLL is the name of the entry record, and is a logical name that identifies all terminals in the payroll department. This record contains three data items that identify the individual terminals or groups that make up the source group.

In a source group record, logical names or physical names identify individual terminals. You can define an individual input source simultaneously in more than one source group. The number of entries you can put in an E-SGP record depends on the length of each entry. For additional information on exactly how many entries a record can hold, see the description of the ACNTRECB field of the ACNTRY parameter list in the "Parameter Lists and Mapping Macros" chapter in the *Systems Programmer Guide*.

Fields in Entry Records

The fields of an entry record are referred to as data items. For each type of entry record, the following table summarizes the type code and what information is permitted for the record name and data items:

Type Code	Record Name	Data Item
SRC	Physical input	Logical input source name to represent the physical source name, or input source name (one data item only).
SGP	Source group name	Physical or logical names of input sources in the group, or names of source groups contained in the group.

Naming Source or Source Group Records

Use caution when naming source and source group records. eTrust CA-ACF2 considers all E-SGP record names source group names. It also considers all entries in an E-SGP record as individual sources. The exception is an entry in an E-SGP record that is also the name of another E-SGP record. In this case, eTrust CA-ACF2 considers the entry a group name.

For example, suppose you create an E-SGP record named ACCT with the following entries: AAA, BBB, CCC, DDD, and ZZZ. eTrust CA-ACF2 considers each of the entries an individual source. Later you create another E-SGP record named ZZZ with the following entries: MMM, LLL, YYY, and XXX. eTrust CA-ACF2 considers ZZZ to be the name of a source group. This means that all the entries in ZZZ become part of the ACCT group. The ACCT group now includes sources MMM, LLL, YYY and XXX, but ZZZ is no longer considered as an input source.

Using the ISPF Panels

The E-SRC records are supported by the ISPF panels. Select option 16 ENTRY from the eTrust CA-ACF2 Security ISPF Option Selection Menu:

```
------------- eTrust CA-ACF2 Security ISPF OPTION SELECTION MENU --------------
OPTION  ===>

   1  RULES     - PROCESS eTrust CA-ACF2 ACCESS AND GENERALIZED RESOURCE RULES
   2  LOGONIDS  - eTrust CA-ACF2 LOGONID CREATION/MAINTENANCE FACILITY
   3  SYSTEM    - eTrust CA-ACF2 SHOW COMMANDS
   4  REPORTS   - eTrust CA-ACF2 REPORT PROGRAM PROCESSOR
   5  UTILITIES - PROCESS eTrust CA-ACF2 UTILITIES
   6  GSO       - GLOBAL SYSTEM OPTIONS SERVICES
   7  NET       - NETWORKING SYSTEM OPTIONS SERVICES
   8  CAC       - MVS DATABASE CACHE RECORD SERVICES
   9  XREF      - SOURCE/RESOURCE AND GROUP RECORD SERVICES
  10  MAC       - MANDATORY ACCESS CONTROL ADMINISTRATION
  11  CPF       - COMMAND PROPAGATION FACILITY SERVICES
  12  FIELD     - RECORD LEVEL PROTECTION CONTROLS
  13  TARGETS   - SET CPF TARGET NODES, DEFAULTS IN USE
  14  PROFILE   - PROCESS PROFILE INFORMATION RECORDS
  15  SMS       - PROCESS DFSMS SUPPORT RECORDS
  16  ENTRY     - PROCESS ENTRY SOURCE RECORDS
  17  SHIFT     - PROCESS SHIFT/ZONE RECORDS
  18  RACDCERT  - PROCESS KEYRING/CERTIFICATE COMMANDS
  19  C-CIC     - PROCESS C-CIC CICS Initialization Records
```

The eTrust CA-ACF2 Security Entry Source Services panel is displayed:

```
------------ eTrust CA-ACF2 Security ENTRY SOURCE SERVICES -------------------
OPTION  ===>

   1  INSERT   - INITIALLY DEFINE ENTRY SOURCE RECORDS

   2  CHANGE   - CHANGE EXISTING ENTRY SOURCE RECORDS

   3  LIST     - DISPLAY ENTRY SOURCE RECORDS

   4  DELETE   - DELETE AN ENTIRE ENTRY SOURCE RECORD

   5  TARGETS  - SET CPF TARGET NODES, DEFAULTS IN USE
```

Note: If you want to process an ACF command at a node other than the default or the current target setting, you must select the TARGET option and specify the target nodes before you select the INSERT, CHANGE, LIST or DELETE options.

The ISPF panels can only be used to administer ENTRY SRC records not ENTRY SGP records. ESGP records can only be administered using the ACF command.

Creating Entry Records

Select option 1 INSERT from the eTrust CA-ACF2 Security Entry Source Services panel to create an Entry Source record. The Insert An Entry Source Record panel is displayed:

```
------------------------ INSERT AN ENTRY SOURCE RECORD ----------------------
COMMAND ===>

  INSERT

  USING RECID   ===>                          OPTIONAL PROTOTYPE RECORD NAME
  RECID         ===>

  DSN           ===>
                                              (PSEUDO DATASET NAME)
  NEWDATA       ===>                          SOURCE

  CLEAR         ===>                          (Y OR N, DEFAULT IS N)
```

Panel Field Descriptions

The fields on the panel are described in the following:

USING RECID
Specify the name of a record that you want to use as a model to create this record.

RECID
Specify the one to eight-character name of the record ID that you want to create.

DSN
Controls which users can access the entry record for the purpose of listing, changing or deleting it.

NEWDATA
Specify the addition of a data item to an entry record.

CLEAR
Specify whether data items of the model record are ignored when the new record is created.

Changing Entry Records

Select option 2 CHANGE from the eTrust CA-ACF2 Security Entry Source Services panel to change an Entry Source record. The Change An Entry Source Record panel is displayed:

```
------------------------ Change an ENTRY SOURCE record -------------------------
COMMAND ===>

Since input sources (E-SRC) use only the first data item, you should not
  use the CHANGE subcommand for them.

  Please delete and re-add the entry source record.
```

Displaying Entry Source Records

Select option 3 LIST from the eTrust CA-ACF2 Security Entry Source Services panel to display Entry Source records. The List An Entry Source Record panel is displayed:

```
------------------------- LIST AN ENTRY SOURCE RECORD -------------------------
COMMAND ===>

  LIST

  MASK RECID    ===>                   RECID MASKED VALUE

  RECID         ===>
```

Panel Field Descriptions

The fields on the panel are described in the following:

MASK RECID
Specify a record name mask to list a group of Entry Source records.

RECID
Specify the one to eight-character name of the record ID that you want to list.

Deleting Entry Records

Select option 4 DELETE from the eTrust CA-ACF2 Security Entry Source Services panel to delete Entry Source records. The Delete An Entry Source Record panel is displayed:

```
------------------------ DELETE AN ENTRY SOURCE RECORD ------------------------
COMMAND ===>

  DELETE

  MASK RECID    ===>                   RECID MASKED VALUE

  RECID         ===>
```

Panel Field Descriptions

The fields of the panel are described in the following:

MASK RECID
Specify a record name mask to delete a group of Entry Source records

RECID
Specify the one to eight-character name of the record ID that you want to delete.

Using the ACF Command

You can process entry records after you have issued the ACF command and have established the appropriate setting: SRC or SGP.

After you establish one of the ENTRY settings, you can issue any of the following ACF subcommands:

- ACCESS
- *CHANGE
- CHKCERT
- CONNECT
- *DELETE
- END or QUIT
- EXPORT
- GENCERT
- GENREQ
- HELP
- *INSERT
- *LIST
- MLSLABEL
- MLWRITE
- REKEY
- REMOVE
- ROLLOVER
- SET or T
- SHOW

- SN
- SYNCH

The common subcommands ACCESS, CHKCERT, CONNECT, EXPORT, END, QUIT, GENCERT, GENREQ, HELP, MLSLABEL, MLWRITE, REKEY, REMOVE, ROLLOVER, SET, SHOW, SN, and SYNCH operate under all settings. The following sections describe in detail each of the other eTrust CA-ACF2 subcommands marked with asterisks (*).

INSERT Subcommand

The INSERT subcommand, under the E-SRC and E-SGP settings, lets you insert an entry record into the Infostorage database.

The INSERT subcommand has the following syntax:

```
Insert  {*|recid|Using(modelrecid) newrecid Type(SRC|SGP) [Clear]
 Newdata(newdata)]}
 Dsn(pseudodsn)]
 [TARGET(null|=|?|nodemask1,...,nodemask100)]
```

In its simplest form, the INSERT subcommand lets you insert an entry record with a record name and one data item:

```
INSERT lv433 NEWDATA(room100)
```

In this example, the physical input source name LV433, a terminal, is the record name. The logical input source name ROOM100 is the data item to be contained in the record.

Parameter Descriptions

The INSERT subcommand has the following parameters.

*** (asterisk)**
Lets you insert the last record you processed.

recid
Specifies the one to eight-character name of an entry source record that you want to create.

USing(*modelrecid*) *newrecid* Type(SRC|SGP) [Clear] [Newdata(*newdata*)]
Lets you insert an entry record by copying the data items and any pseudo data set name from an existing model entry record. (The pseudo data set name is described in the following under the description of the DSN parameter.) The following example illustrates the USING parameter:

```
INSERT USING(payroll) finance TYPE(sgp) NEWDATA(room200)
```

Using the source group record PAYROLL as a model, this sample subcommand inserts a record for the source group FINANCE. Any data items and any pseudo data set name in PAYROLL is copied in FINANCE. The logical input source name ROOM200 is also added to the source group FINANCE.

You can specify the following with the USING parameter:

- *newrecid*—specifies the one to eight-character name of an entry record that you want to create.
- TYPE(SRC|SGP)—specifies the type code of the model entry record. Valid values are: SRC or SGP.
- CLEAR—Lets you create a new record that does not contain the data items from the record name specified with the USING parameter.

  For example, if you enter the following command, eTrust CA-ACF2 creates an entry record for LVL145 with only the ROOM110 data item:

  ```
  INSERT lv145 USING(lv133) CLEAR NEWDATA(room110)
  ```

  It does not carry over any data items from the LV133 record. If you do not specify CLEAR, eTrust CA-ACF2 copies all data items from the model record, and adds NEWDATA values. Specify CLEAR with the USING parameter only.
- NEWDATA(*newdata*)—Specifies the addition of a data item to an entry record. You can use the NEWDATA parameter to identify a logical source name. eTrust CA-ACF2 considers the data you specify as a single data item, even though many logical source names can be entered here. For example, you could enter the following:

  ```
  acf
   ACF
  set entry (SGP)
   SGP
  insert floor5 newdata(room501 room502 room503)
  ```

  eTrust CA-ACF2 considers ROOM501, ROOM502, and ROOM503 to be a single data item. You cannot replace ROOM502 with ROOM506 using the OLDDATA parameter as follows:

  ```
   SGP
  change floor5 olddata(room502) newdata(room506)
   FLOOR5 UNCHANGED
  ```

  Since eTrust CA-ACF2 views the data item to be ROOM501 ROOM502 ROOM503, it does not find a matching data item for ROOM502 and cannot perform the replace.

  To perform the replace, enter the following commands:

  ```
   SGP
  change floor5 olddata(room501 room502 room503) newdata(room506)
  change floor5 newdata(room501 room503)
  ```

The first CHANGE subcommand replaces ROOM501 ROOM502 ROOM503 with ROOM506. The second CHANGE subcommand adds two logical sources to ROOM506.

If you issue a LIST subcommand to verify the results of the change, you would see the following:

```
 SGP
list floor5
TYPE: SGP   ENTRY: FLOOR5    3 DATA ITEMS
 ROOM501
 ROOM503
 ROOM506
```

DSN(*pseudodsn*)

Controls which users can access the entry record for the purpose of listing, changing, or deleting it. This parameter lets you specify a pseudo data set name that can be the name of an actual or nonexistent data set. This data set name must be fully qualified and entered without single quotes. Any users with access to the pseudo data set using access rules also have access to the entry record. Without the use of the DSN parameter, a user's access to the entry record is limited by other user attributes, such as special privileges and scopes.

TARGET(null|=|?|*nodemask1,...,nodemask100*)

Specifies that you want to process structured infostorage records for a target node or group of nodes.

- **null**—indicates that you want this subcommand to process at the home node only. SET TARGET() indicates that you want the subcommand to process at the home node.
- **=** (equal sign)—indicates that you want to process ACF subcommands at the home node only.
- **?** (question mark)—indicates that you want this subcommand to process at the default target nodes.
- ***nodemask1,...,nodemask100***—indicates the node names or masks where you want this subcommand to process. You can specify up to 100 target nodes.

If you want to process an ACF command at a node other than the default node or the current target setting, you must select the TARGET parameter **before** any other parameters.

Making SGP Records Active

After inserting or changing a source group (SGP) record, the new values are not active until the next system startup (IPL) or until you enter the following operator command:

```
F ACF2,NEWXREF
```

The NEWXREF operator command dynamically updates the system's active input source cross-reference table. See Appendix C, "Console Operator Commands" in the *Systems Programmer Guide* for more information.

CHANGE Subcommand

The CHANGE subcommand provides several methods to add, replace, and remove data items in entry records. **Note:** Since input sources (E-SRC) use only the first data item, you should not use the CHANGE subcommand for them.

The syntax of this subcommand is:

```
CHAnge {*|recid|Like(recidmask)}
       {[Newdata(newdata)] [Olddata(olddata)] [Verdata(verdata)]}
       [Clear]
       [Dsn(dsn)]
       [TARGET(null|=|?|nodemask1,...,nodemask100)]
```

Parameter Descriptions

The CHANGE subcommand takes the following parameters.

*** (asterisk)**
Specifies the last record of this type processed since you established the ENTRY setting with a specific type code (for example, SET ENTRY(SGP)).

recid
Specifies the one to eight-character name of one particular entry record.

LIKE(*recidmask*)
Lets you add a data item to more than one entry record at a time. For example, combine LIKE with VERDATA, as follows:

```
CHANGE LIKE(recidmask) NEWDATA(newdata) VERDATA(verdata)
```

eTrust CA-ACF2 adds the new data item to each entry record that matches the mask and contains the data item specified in the VERDATA parameter. You can use the OLDDATA and VERDATA parameters in a similar manner to remove a data item from more than one record at a time:

```
CHANGE LIKE(recidmask) OLDDATA(olddata) VERDATA(verdata)
```

eTrust CA-ACF2 removes an old data item from each entry record that matches the mask and contains the data item specified in the VERDATA parameter.

NEWDATA(*newdata*) OLDDATA(*olddata*) VERDATA(*verdata*)
Determines the effect of the CHANGE subcommand as follows. Specify at least one of the parameters: Newdata, Olddata, or Verdata. The following list presents examples of how to use these parameters:

- To add a new data item to an entry record, use the Newdata parameter:

  ```
  CHANGE pay03 NEWDATA(room100)
  ```

This subcommand adds another input source name, ROOM100, to the source group PAY03.

- To remove a data item from an entry record, use the Olddata parameter:

  ```
  CHANGE pay03 OLDDATA(room100)
  ```

  This subcommand removes the input source name ROOM100 from the source group PAY03.

- To replace a data item with a new one, use the Olddata and Newdata parameters:

  ```
  CHANGE pay03 OLDDATA(room100) NEWDATA(room110)
  ```

  For the source group PAY03, this subcommand replaces the input source name ROOM100 with the input source name ROOM110.

- To remove a data item from an entry record if that record contains the data item specified by the Verdata parameter, use the following:

  ```
  CHANGE pay03 VERDATA(room100) OLDDATA(room110)
  ```

  This subcommand removes the input source name ROOM110 if the input source name ROOM100 currently exists in the source group PAY03. The names specified in the Verdata and Olddata parameters can be the same.

- To add a data item to an entry record if that record contains the data item specified by the Verdata parameter, use the following:

  ```
  CHANGE pay03 VERDATA(room100) NEWDATA(room110)
  ```

  This subcommand adds the input source name ROOM110 if the input source name ROOM100 currently exists in the source group PAY03. The names specified in the Newdata and Verdata parameters can be the same.

- To replace a particular data item in an entry record if that record contains the data item specified by the Verdata parameter, use the following:

  ```
  CHANGE pay03 VERDATA(room100) NEWDATA(room112) OLDDATA(room110)
  ```

  This subcommand replaces the input source name ROOM110 with the input source name ROOM112 if the source group PAY03 currently contains the input source name ROOM100. The names specified in the Verdata parameter can be the same as the ones specified in the Olddata or Newdata parameter.

CLEAR

Lets you remove existing data items from a record and replace them with a new data item. If you do not specify Clear, eTrust CA-ACF2 copies all existing data items from the existing record and adds Newdata values.

DSN(*dsn*)
Lets you change the pseudo data set name that controls which users have access to the entry record. If this parameter specifies a pseudo data set name where none existed previously, the name is added to the entry record. If this parameter specifies a name where one currently exists, the new name replaces the old name. If this parameter is specified as DSN(), the existing name is removed. The data set name must be a fully qualified data set name. Specify the data set name without single quotes.

TARGET(null|=|?|*nodemask1,...,nodemask100*)
Specifies that you want to process entry records for a target node or group of nodes.

- **null**—indicates that you want this subcommand to process at the home node only. Specify TARGET() to indicate that you want the subcommand to process at the home node.
- **=** (equal sign)—indicates that you want to process this ACF subcommands at the home node only.
- **?** (question mark)—indicates that you want this subcommand to process at the default target nodes.
- ***nodemask1,...,nodemask100***—indicates the node names or masks where you want this subcommand to process. You can specify up to 100 target nodes.

Making a Changed Record Active

After you have changed a source group record, enter the following operator command:

```
F ACF2,NEWXREF
```

This command updates the input source cross-reference table. See Appendix C, "Console Operator Commands" of the *Systems Programmer Guide* for information about this command.

LIST Subcommand

The LIST subcommand under the ENTRY setting lists the type, record name, and number of data items for the specified entry record.

The LIST subcommand has the following syntax:

```
List     {*|recid|Like(recidmask)}
         [TARGET(null]=|?|nodemask1,...,nodemask100)]
```

Parameter Descriptions

The LIST subcommand takes the following parameters:

*** (asterisk)**
Specifies, by record name, the listing of the last entry record processed since you established the current ENTRY setting.

recid
Specifies the one to eight-character record name of one particular entry record to be listed.

LIKE(*recidmask*)
Specifies a record name mask that lets a group of entry records with similar record names be listed by one LIST subcommand.

TARGET(null|=|?|*nodemask1,...,nodemask100*)
Specifies that you want to process entry records for a target node or group of nodes.

- **null**—indicates that you want this subcommand to process at the home node only. Specify TARGET() to indicate that you want the subcommand to process at the home node.
- **=** (equal sign)—indicates that you want to process ACF subcommands at the home node only.
- **?** (question mark)—indicates that you want this subcommand to process at the default target nodes.
- ***nodemask1,...,nodemask100***—indicates the node names or masks where you want this subcommand to process. You can specify up to 100 target nodes.

DELETE Subcommand

The DELETE subcommand removes an entry record from the Infostorage database (including record name, data items, and any pseudo data set name).

The DELETE subcommand has the following syntax:

```
DELete   {*|recid|LIke(recidmask)}
         [TARGET(null|=|?|nodemask1,...,nodemask100)]
```

Parameter Descriptions

The DELETE subcommand takes the following parameters:

*** (asterisk)**
Specifies, by record name, processing of the last entry record processed since establishing the current ENTRY setting.

recid
Specifies the one to eight-character record name of one particular entry record.

LIKE(*recidmask*)
Specifies a record name mask that lets a group of entry records with similar record names be deleted by one DELETE subcommand.

When you enter any online command that deletes multiple records from the eTrust CA-ACF2 database, eTrust CA-ACF2 prompts you to confirm the DELETE LIKE request. If you respond Y, the command is executed. If you respond N or anything else, eTrust CA-ACF2 ignores the command and prompts for the next command.

TARGET(null|=|?|*nodemask1,...,nodemask100*)
Specifies that you want to process entry records for a target node or group of nodes.

- **null**—indicates that you want this subcommand to process at the home node only. Specify TARGET() to indicate that you want the subcommand to process at the home node.
- **=** (equal sign)—indicates that you want to process this ACF subcommands at the home node only.
- **?** (question mark)—indicates that you want this subcommand to process at the default target nodes.
- ***nodemask1,...,nodemask100***—indicates the node names or masks where you want this subcommand to process. You can specify up to 100 target nodes.

Activating Changes to E(SRC) Records

eTrust CA-ACF2 loads entry source records into storage at initialization time. Any changes you make to these records once eTrust CA-ACF2 is running (like adding a new record, changing an existing record, or deleting a record) do not take effect until you rebuild the records.

To rebuild entry source records dynamically, issue the following operator command:

```
F ACF2,REBUILD(SRC),CLASS(E)
```

This causes eTrust CA-ACF2 to recognize any additions, changes, or deletions to those records so validation takes place according to the new definitions you have specified.

For more information about the REBUILD command, see Appendix C, "Console Operator Commands" in the *Systems Programmer Guide*.

Activating Changes to E(SGP) Records

After adding, changing, or deleting a source group (SGP) record, the new values are not active until the next system startup (IPL) or until you enter the following operator command:

```
F ACF2,NEWXREF
```

This command dynamically updates the input source cross-reference table. See Appendix C, "Console Operator Commands" in the *Systems Programmer Guide* for more information.

Chapter 9

Maintaining Cross-Reference Records

The XREF Setting

The purpose of the cross-reference (XREF) setting is to let you define groups of sources or groups of resources for eTrust CA-ACF2 validation processing. You must define the sources or resources in a group only once. Then use the group's name whenever you must specify the same group again.

For example, if you want users to access a system only through a particular group of terminals, you could define the individual terminals in the group and assign the group a source group name in an XREF source group (X-SGP) record. You then must specify only the source group name, rather than each individual terminal, in the SOURCE field of users' logonid records.

During system entry validation, eTrust CA-ACF2 matches the group of terminals defined in the X-SGP record with the source group named in the logonid records to discover which terminals the users are permitted to use to enter the system.

This cross-reference concept applies to resource grouping as well. For example, if you want users to have access to only a particular group of transactions, you could define the individual transactions in the group and assign the group a resource group name in an XREF resource group (X-RGP) record. Then write only one resource rule for the entire group of transactions by specifying the resource group name in the resource rule record key. You save yourself the work of writing a rule for each transaction.

During resource access validation, eTrust CA-ACF2 matches the group of transactions defined in the X-RGP record with the resource group named in the resource rule to discover which transactions the users are permitted to access.

Logonids can be scoped to limit their access to cross-reference records, as with other infostorage records; however, unlike other infostorage records, cross-reference records are sysid-dependent. For more information about the syntax to use when scoping cross-reference records, see Specifying Scope Record Fields in the "Maintaining Scope Records" chapter.

In addition to using X-SGP and X-RGP records, eTrust CA-ACF2 IMS Batch Interface users can use XREF IDS (X-IDS) records for cross referencing segment codes to segment names. For more information about the X-IDS record, see the *IMS Batch Support Guide*.

Cross-Reference Source Group (X-SGP) Records

You can use the X-SGP record or the E-SGP record for grouping sources. However, we recommend that you use the X-SGP record because it provides more processing options. For more information about E-SGP records, see the "Maintaining Entry Source and Source Group Records" chapter. The following sections describe the function of the X-SGP record and how to define the record.

What Source Group Records Do

Depending on the type of access to be validated, the X-SGP record cross-references the SOURCE field in one of the following records:

- Logonid record
- Access rule record
- Resource rule record

An X-SGP record can also cross-reference another X-SGP record.

Using the SOURCE Field of the Logonid Record

For system access validations, eTrust CA-ACF2 matches the source group defined in the X-SGP record with the source group named in the SOURCE field of the user's logonid record. To ensure that a user can log on to only a particular group of terminals, you can define the terminals belonging to the group in the X-SGP record, then specify the name of the source group record in the SOURCE field of the user's logonid record. For more information about logonid records, see the "Maintaining Logonid Records" chapter.

Using the SOURCE Field of the Data Set Access Rule

For data set access validations, eTrust CA-ACF2 matches the source group defined in the X-SGP record with the source group named in the SOURCE field of the data set access rule. To ensure that users can access a data set only from a particular group of terminals, you can define the terminals belonging to the group in the X-SGP record, then specify the name of the source group record in the SOURCE field of the access rule written for the data set. For more information about data set access rules, see the "Maintaining Access Rules" chapter.

Using the SOURCE Field of the Resource Rule

For resource access validations, eTrust CA-ACF2 matches the source group defined in the X-SGP record with the source group named in the SOURCE field of the resource rule. To ensure that users can access a resource only from a specific group of terminals, you can define the terminals belonging to the group in the X-SGP record, then specify the name of the source group record in the SOURCE field of the resource rule written for the resource. For more information about resource rules, see "Maintaining Resource Rules."

Cross-Referencing X-SGP Records

Besides cross-referencing the other eTrust CA-ACF2 records just described, X-SGP records can cross-reference each other. This cross-reference function lets you define groups in larger groups.

For example, suppose you have three groups of terminals. Some users are permitted to log on to all three groups, and the other users can log on to only one of the three groups. In this case, you want to define each of the three individual groups of terminals, and one larger group that consists of all three individual groups.

To do this, you define a total of four X-SGP records, one for each individual group of terminals, and one for the larger group that consists of all three of the individual terminal groups. For each X-SGP record ID (recid), you use the name of the terminal group or the name of the set of terminal groups that the X-SGP record defines. You might have X-SGP records with record IDs such as TERMA, TERMB, TERMC, and GROUP1, where GROUP1 consists of TERMA, TERMB, and TERMC.

After you have defined all of the X-SGP records, specify one of the individual terminal groups (TERMA, TERMB, or TERMC) or the set of all three terminal groups, GROUP1 in the SOURCE field of the users' logonid records.

- Specify TERMA in the SOURCE field of the logonid records for those users who are permitted to log on to only TERMA terminals.
- Specify TERMB in the SOURCE field of the logonid records for those users allowed to log on to only the terminals in TERMB.
- Specify TERMC in the SOURCE field of the logonid records for those users permitted to log on to only TERMC terminals.
- Specify GROUP1 in the SOURCE field of the logonid records for those users permitted to log on to TERMA, TERMB, and TERMC terminals.

You can also have a set of X-SGP records cross-reference another set of X-SGP records. Extending the scenario just described, suppose that you want to permit some users to have access to a system from the set of terminal groups indicated in the X-SGP GROUP1 record and from another set of terminal groups defined in the record X-SGP GROUP2. In this case, you could define an X-SGP record called GROUPA. GROUPA consists of GROUP1 and GROUP2.

You can use this X-SGP to X-SGP cross-reference ability as an indexing process to define up to 25 levels.

X-SGP Record Fields

The combined field lengths for X-SGP records are limited only by the size of the record, which is approximately 4K. You can define the following fields in an X-SGP record:

Record ID	Fields
grpname	SOURCE | GROUP INCLUDE(*entry1,…,entryn*) EXCLUDE(*entry1,…,entryn*)

Field Descriptions

grpname
Specifies a one to eight-character name of the source group.

SOURCE|GROUP
Specifies the function of the record. SOURCE indicates that the record defines a group of sources by logical or physical source names. The default value is SOURCE.

GROUP indicates that the record defines a group of source groups. Specify GROUP only when you want to define an X-SGP record that cross-references other X-SGP records. See the previous section Cross-Referencing Source Gourp (X-SGP) Records for an example.

INCLUDE(*entry1,...,entryn*)
Specifies the terminals, group of terminals, or group of terminal groups that you want to include in the record.

Each entry (whether it is a terminal ID or a record ID) can consist of a maximum of eight characters. Separate the entries with commas or blanks. Entries can be masked using the asterisk or dash characters. For details on how to use masking characters, see Masking Logonid Records in the "Maintaining Logonid Records" chapter. This field is limited only by the size of the record, that is, approximately 4K.

- If you specify SOURCE, INCLUDE specifies the individual terminals, logical or physical, or a combination of both, that belong to this source group.
- If you specify the GROUP keyword, INCLUDE specifies the record IDs of the X-SGP records that are cross-referenced by this X-SGP record.

You can specify terminal IDs or record IDs; you cannot specify both in the same X-SGP record. That is, if you want to use both the SOURCE and the GROUP functions, you must define a separate X-SGP record for each and define a unique record ID for each record.

EXCLUDE(*entry1,...,entryn*)
Specifies the terminals, group of terminals, or group of terminal groups that you **do not** want to include in the record. Specify EXCLUDE entries only when you mask the INCLUDE field. For example, suppose you have 20 terminals whose IDs all begin with LV1 and you want to include all but two of these terminals in your source group. Specify LV1- in the INCLUDE field and specify the full terminal IDs of the two terminals you want to exclude, LV105 and LV106, in the EXCLUDE field.

Each entry (whether it is a terminal ID or a record ID) can consist of a maximum of eight characters. Separate the entries with commas or blanks. Entries can be masked. For details on how to use masking characters, see Masking Logonid Records in the "Maintaining Logonid Records" chapter. This field is limited only by the size of the record, that is, approximately 4K.

- If you specify SOURCE, EXCLUDE specifies the individual terminals that are excluded from the group defined by the INCLUDE field.
- If you specify GROUP, EXCLUDE specifies the record IDs to be excluded from the set of X-SGP records that this X-SGP record cross-references.

You can specify terminal IDs or record IDs; you cannot specify both in the same X-SGP record. That is, if you want to use both the SOURCE and the GROUP functions, you must define a separate X-SGP record for each and define a unique record ID for each record.

EXCLUDE statements only apply to the group specified; they **do not** apply to any groups of groups. If an EXCLUDE is done for a group that includes another group, a security exposure might result.

How eTrust CA-ACF2 Sorts X-SGP Records

When you specify the EXCLUDE field, remember that eTrust CA-ACF2 performs sort processing from the most specific to the most general. The most specific entry is always chosen as the valid entry, because it is matched first.

Therefore, if you specify LV1*5 in the INCLUDE field, and LV105 in the EXCLUDE field, LV105 is excluded, even though it matches the masked field, because LV105 is more specific. More importantly, remember the EXCLUDE entry can **override** the INCLUDE entry. For example, if you had specified LV1- in the INCLUDE field and LV1 in the EXCLUDE field, the LV1 entry overrides the LV1- entry because it is more specific. Also, by default, if the INCLUDE field and the EXCLUDE field have the same entry, the entry is excluded from the group.

Cross-Reference Resource Group (X-RGP) Records

You can run eTrust CA-ACF2 Note 4 resource group definitions with X-RGP records. However, when both eTrust CA-ACF2 Note 4 and X-RGP records are active, eTrust CA-ACF2 gives control to eTrust CA-ACF2 Note 4 first. Therefore, it is possible that some X-RGP records might not be processed.

Consider the following information when choosing between Note 4 and X-RGP records:

- If you have a large number of transactions but only want to protect a few, you should use Note 4 and E-SGP records. This method lets you create a rule with all asterisks to allow users access to all transactions and a more specific rule that allows access to a specific transaction for a specific user as shown in the following sample:

```
$KEY(********) TYPE(ITR)
 - UID(-) ALLOW

$KEY(PAY*) TYPE(ITR)
 - UID(PAY1) ALLOW
```

- eTrust CA-ACF2 processes resource group records differently using Note 4 than it does using X-RGP records. For this reason, sites that try to convert from using Note 4 to X-RGP records should delete the $KEY(********) rule. Here is why:
  - When you use Note 4, eTrust CA-ACF2 uses the Prevalidation exit to obtain the name of the resource group from an E-SGP record. Then eTrust CA-ACF2 uses the group name to validate access to the resource. If eTrust CA-ACF2 cannot find the E-SGP record, it searches the directory. If eTrust CA-ACF2 finds the entry in the directory, it locates the record and validates access to the resource. Here is when eTrust CA-ACF2 locates the $KEY(********) rule. If eTrust CA-ACF2 cannot find a matching rule, it denies access.
  - When you use X-RGP records, eTrust CA-ACF2 searches the resource directory first. If you have a $KEY(********) rule that allows access to all transactions, then eTrust CA-ACF2 uses this rule to allow access before it finds the more specific rule. X-RGP processing is never performed.

    If eTrust CA-ACF2 finds a matching entry, it locates the resource rule and performs access validation. If eTrust CA-ACF2 cannot find a matching entry in the directory, it searches all the X-RGP records. When eTrust CA-ACF2 finds a matching entry in the INCLUDE or EXCLUDE lists of an X-RGP record, it performs access validation. If the access is allowed, X-RGP processing ends. If access is denied, eTrust CA-ACF2 continues to search more X-RGP records until it finds another matching entry in an INCLUDE or EXCLUDE list, looking for a record that allows access. If eTrust CA-ACF2 cannot find an X-RGP record that allows access, access is denied.
- X-RGP records let you use masking.
- X-RGP records let you exclude transactions from a resource group.

For more information about eTrust CA-ACF2 Note 4, see Appendix B, "eTrust CA-ACF2 Notes," in the *Systems Programmer Guide*.

What Resource Group Records Do

The X-RGP record cross-references the resource name and the $TYPE control statement in a resource rule, and can also cross-reference another X-RGP record. For resource access validations, eTrust CA-ACF2 matches the X-RGP record ID with the resource name specified in the $KEY field of the resource rule record. eTrust CA-ACF2 matches the value defined in the TYPE field of the X-RGP record with the $TYPE control statement in the resource rule.

To let users access a group of transactions, do the following:

- Ensure that the type codes of the resources are defined in the GSO RESDIR or INFODIR record **before** you try to create the resource group. The rules should not be defined as transient.
- Specify the individual transactions that belong in the group.
- Assign the group a resource group name (which is the record ID of the X-RGP record).
- Specify the type code of the resource rule in an X-RGP record.
- Write one resource rule for all of the transactions that belong to the resource group by using the X-RGP record ID. Specify the resource group name (the X-RGP record ID) in the $KEY field of the resource rule record and ensure that the same type code is used for both the resource rule and the X-RGP record. XREF(RGP) resource validation does **not** support the "extended" resource rule format. Even if your resource group names consist of multiple qualifiers, you must specify the complete resource group name on the $KEY statement of the corresponding resource rule.
- Rebuild the resource directory for the resource type using the REBUILD console operator command.
- Rebuild the cross-reference table using the NEWXREF console operator command.

For more information about resource rules, see "Maintaining Resource Rules." For information about the eTrust CA-ACF2 console operator commands, see the *Systems Programmer Guide*.

Cross-Referencing X-RGP Records

Besides cross-referencing resource rules, X-RGP records can cross-reference each other. This cross-reference function lets you define groups in larger groups. For example, you have three groups of transactions. Some users can access all three groups while others can access only one of the three groups. You must define each of the three individual groups of transactions and one larger group that consists of all three individual groups to eTrust CA-ACF2.

To do this, define a total of four X-RGP records, one for each individual group of transactions, and one for the larger group that consists of all three of the individual transaction groups. Use the names of the transaction groups for the record IDs of the X-RGP records. You will have X-RGP records with record IDs such as TRANA, TRANB, TRANC, and GROUP1, where GROUP1 consists of TRANA, TRANB, and TRANC.

After you define all of the X-RGP records, write one resource rule for each X-RGP record, and use the record IDs (TRANA, TRANB, TRANC, GROUP1) for the $KEY resource names. You also must ensure that the type codes you specify in the X-RGP records match the type codes in the corresponding resource rule records.

You can also have a set of X-RGP records cross-reference another set of X-RGP records. Extending the scenario just described, suppose that you want some users to have access to the set of transaction groups indicated in the X-RGP GROUP1 record and another set of transaction groups defined in the X-RGP GROUP2 record. In this case, you define an X-RGP record called GROUPA. GROUPA consists of GROUP1 and GROUP2. You can use this X-RGP to X-RGP cross-reference ability as an indexing process to define as many as 25 levels.

How eTrust CA-ACF2 Processes Requests for Resource Access

This example assumes that a rule does not exist that could be used to **directly** (that is, outside of resource grouping) validate access to the specific resource. If a resource rule exists that directly matches (with masking) the requested resource name, resource group processing is not performed.

Assume a resource belongs to more than one resource group. When access to the resource is attempted, eTrust CA-ACF2 first tries to find a resource rule that has a $KEY that matches the requested resource name. If eTrust CA-ACF2 does not find a rule, eTrust CA-ACF2 searches for resource rules that have $KEY values that correspond to the names of the resource groups that you have defined. Authorization is determined when a rule permits the access, regardless of logging, or when no more rule sets are found with a $KEY that corresponds to the group names whose groups contain the resource.

For example, in the validation process for a transaction (resource), if eTrust CA-ACF2 finds a rule set that causes the access to be allowed and logged **before** it finds a rule set that only allows the access, the access is allowed and logged. No additional rule sets are checked. The rule that only allows the access is never checked.

When you implement resource access controls using the resource grouping facility, you must carefully consider the way you want the accesses authorized and logged. Consider using the ACFRGP utility, documented in the *Reports and Utilities Guide,* to provide information about existing resource group definitions and about how eTrust CA-ACF2 currently validates those resources. From this basis, you can make decisions about any changes you might need to make to achieve the accesses you desire to those resources.

X-RGP Record Fields

The combined field lengths are limited only by the size of the X-RGP record. You can define the following fields in an X-RGP record:

Record ID	Fields
grpname	RESOURCE|GROUP INCLUDE(*entry1,…,entryn*) EXCLUDE(*entry1,…,entryn*) TYPE(*code*)

Field Descriptions

grpname
Specifies a one to 24-character name of the resource group. XREF(RGP) resource validation does **not** support the "extended" resource rule format. Even if your resource group names consist of multiple qualifiers, you must specify the complete resource group name on the $KEY statement of the corresponding resource rule.

RESOURCE|GROUP
Specifies the function of the record. RESOURCE indicates that the record defines a group of resources. The default value is RESOURCE.

GROUP indicates that the record defines a group of resource groups. Specify GROUP only when you want to define an X-RGP record that cross-references other X-RGP records. See Cross-Referencing Source Group (X-SGP) Records earlier in this chapter for an example.

INCLUDE(*entry1,...,entryn*)

Specifies the resources or group of resource groups that you want to include in the record. This field accepts a maximum of 40 characters per resource name. For extended resources, eTrust CA-ACF2 first checks access based on the resource rule up to 40 characters; if no rule is found, the process is repeated using the first qualifier of the resource name. Each entry must be separated by a comma or a blank.

- If you specify RESOURCE, the INCLUDE field specifies the resource names that belong in this resource group. For resource names, this field accepts a maximum of 40 characters per entry. Separate entries with a comma or a blank.
- If you specify GROUP, the INCLUDE field specifies the record IDs of the X-RGP records that are cross-referenced by this X-RGP record. For record ID entries, this field accepts a maximum of 24 characters per entry. Separate entries with a comma or a blank.

You can specify resource names or record IDs; you cannot specify both in the same X-RGP record. That is, if you want to use both the RESOURCE and the GROUP functions, you must define a separate X-RGP record for each and define a unique record ID for each record.

You can mask resource or record ID entries. The field is limited only by the size of the record; that is, the amount of storage that is dedicated to the record determines the maximum size of the INCLUDE field.

EXCLUDE(*entry1,...,entryn*)

Specifies the resources or group of resource groups that you want to exclude from the record.

- If you specify RESOURCE, the EXCLUDE field specifies the individual resources that are excluded from the group defined by the INCLUDE field. For resource entries, this field accepts a maximum of 40 characters per entry. Separate entries with a comma or a blank.
- If you specify GROUP, the EXCLUDE field specifies the record IDs to be excluded from the set of X-RGP records to which this X-RGP record cross-references. For record ID entries, this field accepts a maximum of 24 characters per entry. Separate entries with a comma or a blank.

You can specify resources or record IDs; you cannot specify both in the same X-RGP record. That is, if you want to use both the RESOURCE and the GROUP functions, you must define a separate X-RGP record for each and define a unique record ID for each record. You can mask resource or record ID entries.

You should use the EXCLUDE field only when you mask the INCLUDE field. For example, you have 20 transactions whose names all begin with TR1 and you want to include all but two of these transactions in your resource group. Specify TR1- in the INCLUDE field, and specify the full transaction names of the two transactions you want to exclude, TR13 and TR17, in the EXCLUDE field.

EXCLUDE statements only apply to the group specified; they **do not** apply to any groups of groups. If an EXCLUDE is done for a group that includes another group, a security exposure might result.

TYPE(*code*)

Specifies the type code for the resource rule to which the X-RGP record cross-references. The TYPE parameter can only be used when you specify RESOURCE. It cannot be used if you specify GROUP. The standard type codes that can be used are described in the chapter, "Maintaining Resource Rules." These type codes must be stored in the resident directory **before** you can create the resource group record.

Note: For mixed case resource TYPEs, the INCLUDE and EXCLUDE values must be typed in mixed case. Use only one kind of TYPE code – mixed case or non-mixed case – for each X(RGP) record. If the TYPE keyword contains both mixed case types and non-mixed case types, the INCLUDE and EXCLUDE lists are upper cased by the ACF command processor. Use the SHOW CLASMAP subcommand of the ACF command to find out which resource types are mixed case. See the "Maintaining Global System Options Records" chapter for more information on the CLASMAP GSO record.

How eTrust CA-ACF2 Sorts X-RGP Records

eTrust CA-ACF2 performs sort processing from the most specific to the most general. For example, if you specify LV**5 in the INCLUDE field and LV105 in the EXCLUDE field, the terminal identified as LV105 is excluded because it more specifically matches the item in the EXCLUDE list.

If you specify LV1- in the INCLUDE field and LV1 in the EXCLUDE field, the terminal identified as LV1 is excluded because it more specifically matches the item in the EXCLUDE list.

If you specify LV3 in the INCLUDE field and LV* in the EXCLUDE field, the terminal identified as LV3 is included because it most specifically matches the item in the INCLUDE list.

If the INCLUDE field and the EXCLUDE field have an identical entry, the entry is excluded from the group. For example, if the security officer includes LV20 and LV21 and excludes LV20, LV20 is excluded.

Using the ISPF Panels

You can use the ISPF panels or the TSO ACF command and subcommands to create, modify, display, and delete structured infostorage records for the XREF records.

To create an XREF record, select option 9 XREF from the eTrust CA-ACF2 Security ISPF Option Selection Menu as shown in the following example:

```
-------------- eTrust CA-ACF2 Security ISPF OPTION SELECTION MENU -------------
OPTION  ===>

   1  RULES     - PROCESS eTrust CA-ACF2 ACCESS AND GENERALIZED RESOURCE RULES
   2  LOGONIDS  - eTrust CA-ACF2 LOGONID CREATION/MAINTENANCE FACILITY
   3  SYSTEM    - eTrust CA-ACF2 SHOW COMMANDS
   4  REPORTS   - eTrust CA-ACF2 REPORT PROGRAM PROCESSOR
   5  UTILITIES - PROCESS eTrust CA-ACF2 UTILITIES
   6  GSO       - GLOBAL SYSTEM OPTIONS SERVICES
   7  NET       - NETWORKING SYSTEM OPTIONS SERVICES
   8  CAC       - MVS DATABASE CACHE RECORD SERVICES
   9  XREF      - SOURCE/RESOURCE AND GROUP RECORD SERVICES
  10  MAC       - MANDATORY ACCESS CONTROL ADMINISTRATION
  11  CPF       - COMMAND PROPAGATION FACILITY SERVICES
  12  FIELD     - RECORD LEVEL PROTECTION CONTROLS
  13  TARGETS   - SET CPF TARGET NODES, DEFAULTS IN USE
  14  PROFILE   - PROCESS PROFILE INFORMATION RECORDS
  15  SMS       - PROCESS DFSMS SUPPORT RECORDS
  16  ENTRY     - PROCESS ENTRY SOURCE RECORDS
  17  SHIFT     - PROCESS SHIFT/ZONE RECORDS
  18  RACDCERT  - PROCESS KEYRING/CERTIFICATE COMMANDS
  19  C-CIC     - PROCESS C-CIC CICS INITIALIZATION RECORDS
```

eTrust CA-ACF2 displays the eTrust CA-ACF2 Security Source And Resource Grouping panel.

```
------------- eTrust CA-ACF2 Security SOURCE AND RESOURCE GROUPING -------------
OPTION  ===>

   1  INSERT   - INITIALLY DEFINE XREF RECORDS

   2  CHANGE   - CHANGE EXISTING XREF RECORDS

   3  LIST     - DISPLAY XREF RECORDS

   4  DELETE   - DELETE AN ENTIRE XREF RECORD

   5  TARGETS  - SET CPF TARGET NODES, DEFAULTS IN USE
```

If you want to process an ACF command at a node other than the default node or the current target setting, select the TARGET option and specify the target nodes **before** you select INSERT, CHANGE, LIST, or DELETE.

Creating XREF Records

Select option 1 INSERT from the eTrust CA-ACF2 Security Source and Resource Grouping panel. eTrust CA-ACF2 displays the following panel:

```
---------------------------- INSERT AN XREF RECORD ----------------------------
COMMAND ===>

  INSERT

  RECORD TYPE   ===> RGP       (RGP, SGP)

  CHANGE TYPE   ===> ADD       (ADD, DEL, REP)

  SYSID         ===>           SYSTEM ID FOR XREF RECORD
  USING SYSID   ===>           OPTIONAL SYSTEM ID FOR COPY OF EXISTING VALUES

  USING RECID   ===>                          OPTIONAL PROTOTYPE RECORD NAME
  RECID         ===>

  FUNCTION      ===>           (RECORD TYPE=RGP (RESOURCE, GROUP))
                               (RECORD TYPE=SGP (SOURCE, GROUP))
  INCLUDE LIST  ===>

  EXCLUDE LIST  ===>

  RESOURCE TYPES ==>                          RESOURCE/GROUP TYPE CODES
```

Panel Field Descriptions

The fields of the panel are described in the following:

RECORD TYPE
Specify RGP to create a resource group or SGP to create a source group.

CHANGE TYPE
Specify one of the following three options. ADD is the default.

- ADD—adds the sources or resources that you specify in the INCLUDE LIST or EXCLUDE LIST field.
- REP—replaces the sources or resources that you specify in the INCLUDE LIST or EXCLUDE LIST field.
- DEL—removes the sources or resources that you specify in the INCLUDE LIST or EXCLUDE LIST field.

SYSID
Specify the system ID to which this record applies. You can also specify a mask. For details on how to specify system IDs, see the "Maintaining Structured Infostorage Records" chapter.

USING SYSID
Specify the system ID where a source or resource group record exists that you want to use as a model for this record.

RECID
Specify the one to eight-character name of the source or the one to 24-character name of the resource group.

USING RECID
Specify the name of a source or resource group record that you want to use as a model.

FUNCTION
Specify SGP to indicate that this record is for a group of sources or RGP if this record is for a group of resources.

INCLUDE LIST
Specify the names of the sources or resources that you want to include in this record.

EXCLUDE LIST
Specify the names of the sources or resources that you want to exclude from this record. Specify this field if you specify a mask in the INCLUDE LIST field.

RESOURCE TYPES
Specify the list of three-character type codes for the resources you specified in the INCLUDE LIST and EXCLUDE LIST.

Changing XREF Records

Select option 2 CHANGE from the eTrust CA-ACF2 Security Source and Resource Grouping panel. eTrust CA-ACF2 displays the following panel:

```
---------------------------- CHANGE AN XREF RECORD -----------------------------
COMMAND ===>


  CHANGE

  RECORD TYPE   ===> RGP       (RGP, SGP)

  CHANGE TYPE   ===> ADD       (ADD, DEL, REP)

  SYSID         ===>           SYSTEM ID FOR XREF RECORD
  MASK SYSID    ===>           MASKED SYSTEM ID FOR XREF RECORD

  MASK RECID    ===>                          RECID MASKED VALUE
  RECID         ===>

  FUNCTION      ===>           (RECORD TYPE=RGP (RESOURCE, GROUP))
                               (RECORD TYPE=SGP (SOURCE, GROUP))
  INCLUDE LIST  ===>

  EXCLUDE LIST  ===>

  RESOURCE TYPES ==>                          RESOURCE/GROUP TYPE CODES
```

Panel Field Descriptions

The fields of the panel are described in the following:

RECORD TYPE
Specify RGP to change a resource group or SGP to change a source group.

CHANGE TYPE
Specify one of the following three options. ADD is the default.

- ADD—adds the sources or resources that you specify in the INCLUDE LIST or EXCLUDE LIST field.
- REP—replaces the sources or resources that you specify in the INCLUDE LIST or EXCLUDE LIST field.
- DEL—removes the sources or resources that you specify in the INCLUDE LIST or EXCLUDE LIST field.

SYSID
Specify the system ID to which this record applies. You can also specify a mask. For details on how to specify system IDs, see the "Maintaining Structured Infostorage Records" chapter.

MASK SYSID
Specify a mask of system IDs to which this record applies.

MASK RECID
Specify a mask of record IDs that you want to change with this command.

RECID
Specify the one to eight-character name of the source group or the one to 24-character name of the resource group.

FUNCTION
Specify SGP to indicate that this record is for a group of sources or RGP if this record is for a group of resources.

INCLUDE LIST
Specify the names of the sources or resources that you want to include in this record.

EXCLUDE LIST
Specify the names of the sources or resources that you want to exclude from this record. Specify this field if you specify a mask in the INCLUDE LIST field.

RESOURCE TYPES
Specify the list of three-character type codes for the resources you specified in the INCLUDE LIST.

Listing XREF Records

Select option 3 LIST from the eTrust CA-ACF2 Security Source and Resource Grouping panel. eTrust CA-ACF2 displays the following panel:

```
----------------------------- LIST AN XREF RECORD -----------------------------
COMMAND ===>

  LIST

  RECORD TYPE   ===> RGP       (RGP, SGP)

  SYSID         ===>           SYSTEM ID FOR XREF RECORD

  MASK SYSID    ===>           MASKED SYSTEM ID FOR XREF RECORD

  MASK RECID    ===>                          RECID MASKED VALUE

  RECID         ===>
```

Panel Field Descriptions

The fields of the panel are described in the following:

RECORD TYPE
Specify RGP to list a resource group or SGP to list a source group.

SYSID
Specify the system ID to which this record applies. You can also specify a mask. For details on how to specify system IDs, see the "Maintaining Structured Infostorage Records" chapter.

MASK SYSID
Specify a mask of system IDs to which this record applies.

MASK RECID
Specify a mask of record IDs that you want to view.

RECID
Specify the one to eight-character name of the source group or the one to 24-character name of the resource group.

Deleting XREF Records

Select option 4 DELETE from the eTrust CA-ACF2 Security Source and Resource Grouping panel. eTrust CA-ACF2 displays the following panel:

```
--------------------------- DELETE AN XREF RECORD ---------------------------
COMMAND ===>

  DELETE

  RECORD TYPE   ===> RGP       (RGP, SGP)

  SYSID         ===>           SYSTEM ID FOR XREF RECORD

  MASK SYSID    ===>           MASKED SYSTEM ID FOR XREF RECORD

  MASK RECID    ===>                          RECID MASKED VALUE

  RECID         ===>
```

Panel Field Descriptions

The fields of the panel are described in the following:

RECORD TYPE
Specify RGP to delete a resource group or SGP to delete a source group.

SYSID
Specify the system ID to which this record applies. You can also specify a mask. For details on how to specify system IDs, see the "Maintaining Structured Infostorage Records" chapter.

MASK SYSID
Specify a mask of system IDs to which this record applies.

MASK RECID
Specify a mask of record IDs that you want to delete.

RECID
Specify the one to eight-character name of the source group or the one to 24-character name of the resource group.

Using the ACF Command

You can use the standard ACF command and subcommands to create, modify, display, and delete structured infostorage records for the XREF records.

You can use the following ACF subcommands to implement XREF records:

- ACCESS
- *CHANGE
- CHKCERT
- CONNECT
- *DELETE
- END or QUIT
- EXPORT
- GENCERT
- GENREQ
- HELP
- *INSERT
- *LIST
- MLSLABEL
- MLWRITE
- REKEY
- REMOVE
- ROLLOVER
- SET or T
- SHOW
- SN
- SYNCH

The common subcommands ACCESS, CHKCERT, CONNECT, EXPORT, END, QUIT, GENCERT, GENREQ, HELP, MLSLABEL, MLWRITE, REKEY, REMOVE, ROLLOVER, SHOW, SN, and SYNCH operate under all settings. The following sections describe in detail each of the other eTrust CA-ACF2 subcommands marked with asterisks (*).

SET Subcommand

To maintain XREF infostorage records, establish the appropriate ACF command setting and type code. The syntax is:

```
Set|T    [XREF(Sgp|Rgp)]
         [TARGET(null]=|?|nodemask1,...,nodemask100)]
         [SYSid(sysid)|DIVision(div)]
         [MSYSid(sysidmask)|MDIV(divmask)]
```

Parameter Descriptions

The parameters for the SET subcommand are described in the following:

XREF(Sgp | Rgp)
Specifies the type of record you want to process: SGP (source group records) or RGP (resource group records).

TARGET(null | = | ? | *nodemask1,...,nodemask100*)
Specifies that you want to process XREF infostorage records for a target node or group of nodes.

- **null**—indicates you want to process ACF subcommands at the home node only.
- **=** (equal sign)—indicates that you want to process ACF subcommands at the home node only.
- **?** (question mark)—indicates that you want to process ACF subcommands at the default target nodes.
- ***nodemask1,...,nodemask100***—indicates the target nodes where you want the ACF subcommands to be processed. You can specify up to 100 node names or masks. Separate entries with a comma or blanks.

If you want to process an ACF command at a node other than the default node or the current target setting, you must select the TARGET parameter **before** any other parameters.

SYSid(*sysid*) | DIVision(*div*)
Specifies the system ID to which this record applies. Enter the one to eight-character SYSID name. Masking does not apply. If you specify an asterisk (*), eTrust CA-ACF2 considers it part of the SYSID name. To process records for several system IDs, use the MSYSID parameter.

You can also specify the SYSID to which an XREF record applies by using the INSERT, CHANGE, LIST, and DELETE subcommands. DIVISION is a synonym for SYSID.

MSYSid(*sysidmask*) | MDIV(*divmask*)
Specifies a group of SYSIDs to which this record applies. MDIV is a synonym for MSYSID.

INSERT Subcommand

The INSERT subcommand lets you create an XREF record for X-SGP or X-RGP records, depending on the type you specified in the SET command.

The INSERT subcommand has the following syntax:

```
Insert   {*|recid [Type(typelist)]|Using(modelrecid) newrecid [Type(typelist)]}
         [SYSid(?|sysid)|DIVision(?|div)]
         [USYSid(?|sysid)|UDIVision(?|div)]
         [Source|Group]
         [Resource|Group]
         [TARGET(null]=|?|nodemask1,...,nodemask100)]
         [Include(entry1,...,entryn)]
         [Exclude(entry1,...,entryn)]
         [ADD|REP|DEL]
```

Parameter Descriptions

The INSERT subcommand accepts the following parameters:

*** (asterisk)**
Specifies the last record you processed while in this ACF command setting. The asterisk specifies only one record. You cannot use the asterisk to represent a record mask.

recid
Specifies the name of the XREF record you want to create. You cannot use masking characters to specify a group of records. Record IDs for source group records can be from one to eight characters. Record IDs for resource group records can be from one to 24 characters.

Type(*typelist*)
Specifies the type codes of the resources in the INCLUDE and EXCLUDE lists. You cannot mask this field. This field applies to X-RGP resource records only. It does not apply to X-RGP group records. The ADD, REP, and DEL parameters apply to the types you specify.

The type codes you specify must be defined in a resident directory **before** you process the resource group record.

USING(*oldrecid*) *newrecid*
Identifies a model XREF record that you want to use to create a new XREF record. All values from the model record are inserted into the new record. Any other fields and values you specify add to or replace the fields and values in the new record.

If you specify the USYSID parameter, the USING parameter is not required. eTrust CA-ACF2 assumes that you want to copy the record for the SYSID specified in the USYSID parameter, but with the same record identification as the record you are inserting.

SYSid(?|*sysid*)|DIVision(?|*div*)
Specifies the one to eight-character SYSID to which this record applies. Specify a question mark (?) to indicate that you want the record to apply to the currently active SYSID.

If you do not specify a SYSID (or USYSID), eTrust CA-ACF2 uses the SYSID value you specified when you established the ACF command setting. If you did not supply a SYSID when you established the ACF command setting, eTrust CA-ACF2 uses the default value defined at eTrust CA-ACF2 startup time. For more information, see the "Maintaining Global System Options Records" chapter. DIVISION is a synonym for SYSID.

USYSid(?|*sysid*)|UDIV(?|*div*)
Specifies the SYSID of a model XREF record you want to use to create a new XREF record. Specify this record ID with the USING or RECID parameter. If you specify a question mark (?) instead of a SYSID, the default SYSID is used. UDIV is a synonym for USYSID.

This time-saving feature lets you insert a record that is similar to a record previously defined for another SYSID. To change any fields and values that differ between the model and the new record, use the ADD, REP, or DEL parameters (described in the following).

TARGET(null|=|?|*nodemask1,...,nodemask100*)
Specifies that you want to process XREF infostorage records for a target node or group of nodes.

- **null**—indicates that you want to process ACF subcommands at the home node only.
- **=** (equal sign)—**i**ndicates that you want to process ACF subcommands at the home node only.
- **?** (question mark)—indicates that you want to process ACF subcommands at the default target nodes.
- ***nodemask1,...,nodemask100***—indicates the target nodes where you want the ACF subcommands to be processed. You can specify up to 100 node names or masks. Separate entries with a comma or blanks.

Source|Group
Specify SOURCE to create a source group record. Specify GROUP to create a group of source group records. These parameters apply to the X-SGP setting only.

Resource|Group
Specify RESOURCE to create a resource group record. Specify GROUP to create a group of resource group records. These parameters apply to the X-RGP setting only.

Include(*entry1,...,entryn*)
Specifies a group of sources, source groups, resources, or resource groups to which this record applies. The ADD, REP, and DEL parameters apply to the names you specify in this field.

Exclude(*entry1,...,entryn*)
Specifies a group of sources, source groups, resources, or resource groups to which this record **does not** apply. Specify names here only if you specify a mask in the INCLUDE parameter. The ADD, REP, and DEL parameters apply to the names you specify in this field.

ADD
Indicates that you want to add the specified sources or resources to those copied from the model record. If a value already exists in a record field, the new value replaces the copied value. This parameter applies to multi-value fields only. The new value is added to the existing values. ADD is the default.

REP
Indicates that you want to replace the specified sources or resources in the model record with those you specify. If any source or resource is not part of the model record, eTrust CA-ACF2 adds that source or resource to the record. This parameter applies to multi-value fields only.

DEL
Indicates that you want to delete the specified source or resource from the newly inserted record. If a field can contain multiple values, then eTrust CA-ACF2 deletes just the values you specify from the field. This parameter applies to multi-value fields only.

CHANGE Subcommand

The CHANGE subcommand lets you change an existing XREF record. Before changing resource group records, issue a LIST LIKE command to see which records would be affected by the change. To avoid confusion, change these records one at a time by specifying the unique key of the record to which the change applies.

The CHANGE subcommand has the following syntax:

```
CHAnge   {*|recid [Type(typelist)|LIKE(recidmask) [Type(typelist)]}
         [SYSid(?|sysid)|DIVision(?|div)]
         [MSYSid(?|sysidmask)|MDIVision(?|divmask)]
         [Source|Group]
         [Resource|Group]
         [TARGET(null|=|?|nodemask1,...,nodemask100)]
         [Include(entry1,...,entryn)]
         [Exclude(entry1,...,entryn)]
         [ADD|REP|DEL]
```

Parameter Descriptions

The CHANGE subcommand takes the following parameters:

*** (asterisk)**
Specifies that you want to change the last record you processed since you established this ACF command setting. (The asterisk does not work in the case of multiple records or masking.)

recid
Specifies the name of the XREF record that you want to change. Record IDs for source group records can be from one to eight characters. Record IDs for resource group records can be from 1 to 24 characters.

TYPE(*typelist*)
Specifies a list of type codes of the resources in the INCLUDE and EXCLUDE lists. You cannot mask this field. This field applies to X-RGP resource records only. It does not apply to X-RGP group records. The ADD, REP, and DEL parameters apply to the type codes you specify.

The type codes you specify must be defined in a resident directory **before** you process the resource group record.

LIKE(*recidmask*)
Specifies a mask for the XREF records that you want to change. You cannot abbreviate the LIKE parameter. This masking follows the same conventions that apply to logonids, as described in the chapter, "Maintaining Logonid Records."

SYSid(?|*sysid*)|DIVision(?|*div*)
Specifies the one to eight-character SYSID to which this record applies. Specify a question mark (?) to indicate that you want the record to apply to the currently active SYSID.

If you do not specify a SYSID (or MSYSID), eTrust CA-ACF2 uses the SYSID value you specified when you established the ACF command setting. If you did not supply a SYSID when you established the ACF command setting, eTrust CA-ACF2 uses the default value defined at eTrust CA-ACF2 startup time. For more information, see What Are SYSIDs? in the chapter entitled, "Maintaining Structured Infostorage Records." DIVISION is a synonym for SYSID.

MSYSid(?|*sysidmask*)|MDIVision(?|*divmask*)
Specifies a mask to indicate the SYSIDs to which this changed record applies. This SYSID mask must contain at least one asterisk or a trailing dash. Structured infostorage records that you change using this parameter have the same record ID, but different SYSIDs. MDIV is a synonym for MSYSID.

Source|Group
Specify SOURCE to change a source record. Specify GROUP to change a source group record. These parameters apply to the X-SGP setting only.

REsource|Group
Specify RESOURCE to create a resource record. Specify GROUP to create a resource group record. These parameters apply to the X-RGP setting only.

TARGET(null|=|?|*nodemask1,...,nodemask100*)
Specifies that you want to process XREF records for a target node or group of nodes.

- **null**—indicates that you want to process ACF subcommands at the home node only.
- **=** (equal sign)—indicates that you want to process ACF subcommands at the home node only.
- **?** (question mark)—indicates that you want to process ACF subcommands at the default target nodes.
- ***nodemask1,...,nodemask100***—indicates the target nodes where you want the ACF subcommands to be processed. You can specify up to 100 node names or masks. Separate entries with a comma or blanks.

Include(*entry1,...,entryn*)
Specifies a group of sources, source groups, resources, or resource groups to which this record applies. The ADD, REP, and DEL parameters apply to the names you specify in this field.

Exclude(*entry1,...,entryn*)
Specifies a group of sources, source groups, resources, or resource groups to which this record **does not** apply. Specify names here only if you specify a mask in the INCLUDE parameter. The ADD, REP, and DEL parameters apply to the names you specify in this field.

ADD
Indicates that you want to add the specified sources or resources to those in the existing record. If a value already exists in a record field, the new value replaces the copied value. This parameter applies to multi-value fields only. The new value is added to the existing values. ADD is the default.

REP
Indicates that you want to replace the sources and resources in an existing record with the fields and values you specify. If any field you specify is not part of the existing record, eTrust CA-ACF2 adds that field and its value to the record. This parameter applies to multi-value fields only.

DEL
Indicates that you want to delete the specified sources and resources from the existing record. This parameter applies to multi-value fields only. ETrust CA-ACF2 deletes the values you specify from the field.

LIST Subcommand

The LIST subcommand lets you display the contents of an XREF infostorage record. The LIST subcommand has the following syntax:

```
List     { *|recid|LIKE(recidmask)}
        [ SYSid(?|sysid)|DIVision(?|div)]
        [ MSYSid(?|sysidmask)|MDIVision(?|divmask)]
        [ TARGET(null|=|?|nodemask1,...,nodemask100)]
```

Parameter Descriptions

The LIST subcommand takes the following parameters:

*** (asterisk)**
Specifies that you want to view the last record you processed since you established this ACF command setting. You cannot use the asterisk to display more than one record. Use the LIKE parameter.

recid
Specifies the name of the XREF record that you want to display. Record IDs for source group records can be from one to eight characters. Record IDs for resource group records can be from one to 24 characters.

LIKE(*recidmask*)
Specifies a group of structured infostorage records you want displayed. You cannot abbreviate the LIKE parameter. Masking follows logonid masking conventions as described in the "Maintaining Logonid Records" chapter.

SYSid(?|*sysid*)|DIVision(?|*div*)
Specifies the one to eight-character SYSID of the record that you want to display. Specify a question mark (?) to indicate that you want the record to apply to the currently active SYSID.

If you do not specify a SYSID (or MSYSID), eTrust CA-ACF2 uses the SYSID value specified when you established the ACF command setting. If you did not supply a SYSID when establishing this setting, eTrust CA-ACF2 uses the default value defined at eTrust CA-ACF2 startup time. DIVISION is a synonym for SYSID. For more information, see the "Maintaining Global System Options Records" chapter.

MSYSid(?|*sysidmask*)|MDIVision(?|*divmask*)
Specifies a mask to indicate the SYSIDs of the record that you want to display. This SYSID mask must contain at least one asterisk or a trailing dash.

TARGET(null|=|?|*nodemask1,...,nodemask100*)
Specifies that you want to process XREF infostorage records for a target node or group of nodes.

- **null**—indicates that you want to process ACF subcommands at the home node only.
- **=** (equal sign)—indicates that you want to process ACF subcommands at the home node only.
- **?** (question mark)—indicates that you want to process ACF subcommands at the default target nodes.
- ***nodemask1,...,nodemask100***—indicates the target nodes where you want the ACF subcommands to be processed. You can specify up to 100 node names or masks. Separate entries with a comma or blanks.

DELETE Subcommand

The DELETE subcommand lets you delete a cross-reference record. The DELETE subcommand has the following syntax:

```
DELete □  {*|recid|LIke(recidmask)}
          [SYSid(?|sysid)|DIVision(?|div)]
          [MSYSid(?|sysidmask)|MDIVision(?|divmask)]
          [TARGET(null|=|?|nodemask1,...,nodemask100)]
```

Parameter Descriptions

The DELETE subcommand takes the following parameters:

*** (asterisk)**
Specifies that you want to delete the last record you processed since you established this ACF command setting. You cannot use the asterisk to delete multiple records. Use the LIKE parameter.

recid
Specifies the name of the XREF record that you want to delete. Record IDs for source group records are from one to eight characters. Record IDs for resource group records are from 1 to 24 characters.

LIke(*recidmask*)
Specifies a group of XREF records that you want to delete. This masking follows the same conventions that apply to logonids, as described in the "Maintaining Logonid Records" chapter. Use extreme care when deleting multiple records with this parameter.

When you issue an online command that deletes multiple records from the eTrust CA-ACF2 databases, eTrust CA-ACF2 prompts you to confirm the DELETE LIKE request. If you respond **Y**, the command executes. If you respond **N** or anything else, eTrust CA-ACF2 ignores the command and prompts for the next command.

SYSID(?|sysid)|DIVision(?|*div*)
Specifies the one to eight-character SYSID of the record that you want to delete. Specify a question mark (?) to indicate that you want to delete the record from the currently active SYSID.

If you do not specify a SYSID (or MSYSID), eTrust CA-ACF2 uses the SYSID value you specified when you established the ACF command setting. If you did not supply a SYSID when you established the ACF command setting, eTrust CA-ACF2 uses the default value defined at eTrust CA-ACF2 startup time. For more information, see the section on the concept of SYSID. DIVISION is a synonym for SYSID.

MSYSid(?|*sysidmask*)|MDIVision(?|*divmask*)
Specifies a mask to indicate the SYSIDs of the record that you want to delete. This SYSID mask must contain at least one asterisk or a trailing dash.

TARGET(null|=|?|*nodemask1,...,nodemask100*)
Specifies that you want to process XREF records for a target node or group of nodes.

- **null**—indicates that you want to process ACF subcommands at the home node only.
- **=** (equal sign)—indicates that you want to process ACF subcommands at the home node in addition to any node masks defined in this parameter.
- **?** (question mark)—indicates that you want to process ACF subcommands at the default target nodes.
- ***nodemask1,...,nodemask100***—indicates the target nodes where you want the ACF subcommands to be processed. You can specify up to 100 node names or masks. Separate entries with a comma or blanks.

X-SGP Record Examples

Suppose a site has terminals that are defined as LV*xx* where *xx* is a number from 00 to 99. The security administrator wants to ensure that users access a data set only from terminals LV20 through LV30 with the exception of terminal LV25. No one should use LV25 to access the data set. The security administrator needs to insert an X-SGP record as follows:

```
acf
 ACF
set x(sgp)
 XRE F
insert lterms source include(lv2*,lv30) exclude(lv2,lv25)
```

LTERMS
The record name. The record ID is the name that the security administrator assigns to this group of terminals.

LV2*

The masked ID that matches any existing terminal ID that begins with the characters LV2.

If this site also has terminals defined as LV2A, LV2B, LV2X and so forth, and the security administrator wants to restrict access, the situation is more complicated because LV2A, LV2B, and LV2X match the criteria specified in the preceeding insert example. In this case, the only way to accomplish inclusion of the proper terminals is to specify the following:

```
acf
 ACF
set x(sgp)
 XRE F
insert lterms source include(lv20,lv21,lv22,lv23,lv24,lv26,lv27,lv28,lv29,lv30)
```

Now suppose that the security administrator creates an X-SGP record for a group of terminals and gives it the record ID LTERMS. And he creates another X-SGP record for a different group of terminals called DTERMS. If he wants to let some users have access to both LTERMS and DTERMS terminals, he can insert an X-SGP record as follows:

```
acf
 ACF
set x(sgp)
 XRE F
insert groupa group include(lterms,dterms)
```

GROUPA

The record name of this X-SGP record, which is the name of the set of X-SGP records to which this X-SGP record is cross-referenced.

LTERMS

The record name of the individual X-SGP records that belong to this set of X-SGP records identified as GROUPA.

DTERMS

The record name of the individual X-SGP records that belong to this set of X-SGP records identified as GROUPA.

The security administrator must specify **LTERMS** in the SOURCE field of the data set access rule entry to refer to users who access the data set from the LTERMS group of terminals and DTERMS in the SOURCE field of the data set access rule entry to refer to users who can access the data set from the DTERMS group of terminals. Also, the security administrator must specify **GROUPA** in the SOURCE field of the data set access rule entry to refer to users who are permitted access to the data set from both terminal groups (LTERMS and DTERMS).

X-RGP Record Examples

Suppose a security administrator wants to grant IMS users access to transactions TR20 through TR30, but not transaction TR25. Transactions are identified as TR*xx* where *xx* is a number 00 through 99. The security administrator then inserts an X-RGP record as follows:

```
acf
 ACF
set x(rgp)
 XRE F
insert trana resource include(tr2-,tr30) exclude(tr25) type(itr)
```

TRANA
The record name that the security administrator selects to assign to this group of transactions.

TR2–
The masked ID to which any transaction ID that begins with the characters TR2 matches.

ITR
The resource rule type code for IMS transactions.

The resource rule for this group of transactions might look like the following:

```
$KEY(TRANA) TYPE(ITR)
 UID (PERCLK) ALLOW
```

Now suppose the security administrator wants to permit some users to access the TRANA group of transactions and another group of transactions that he has identified as TRANB in another X-RGP record. The security administrator inserts an X-RGP record as follows:

```
acf
 ACF
set x(rgp)
 XRE F
insert group1 group include(trana,tranb)
```

GROUP1
The record name of this X-RGP record that is the name of the set of X-RGP records to which this X-RGP record refers.

TRANA
The record ID of the individual X-RGP records that belong to this set of X-RGP records identified as GROUP1.

TRANB
The record ID of the individual X-RGP records that belong to this set of X-RGP records identified as GROUP1.

To follow through with his security plan, the security administrator must be sure to specify TRANA in the $KEY field of the resource rule for the **TRANA** group of transactions, TRANB in the $KEY field of the resource rule for the **TRANB** group of transactions, and **GROUP1** in the $KEY field of the resource rule for the set of transaction groups (TRANA and TRANB). **ITR must be the type code specified for all these rules**. In the previous example, TRANA contains certain transactions, TRANB contains certain transactions and GROUP1 includes transactions from TRANA and TRANB. If someone attempts access to a transaction that is in TRANA (TR20), it is included in GROUP1 because all transactions in TRANA are included in GROUP1.

If a TRANA rule exists that permits a user access, and a GROUP1 rule exists that prevents that user access, validation processing eventually grants access because **some** rule permits access. Access is also granted if the TRANA rule prevents access and the GROUP1 rule grants access.

As an example, assume the payroll manager needs access to all the transactions in both TRANA and TRANB. It is necessary to write only a GROUP1 rule. It is possible that the TRANA and the TRANB rule prevent access for the payroll manager.

```
$KEY(TRANA) TYPE(ITR)
 UI D(PERCLK) ALLOW

$KEY(TRANB) TYPE(ITR)
 UI D(ACTCLK) ALLOW

$KEY(GROUP1) TYPE(ITR)
 UI D(PAYMGR) ALLOW
```

The TRANA rule permits access to personnel clerks. The TRANB rule permits access for accounting clerks. Both rules prevent the payroll manager from access because they do not include his UID. The GROUP1 rule permits his access. Because GROUP1 includes TRANA and TRANB, access is granted to the payroll manager even though the TRANA and TRANB rules prohibit his access.

Migration Considerations

This section describes migrating E-SGP records to X-SGP records and migrating from using Note 4 to using X-RGP records.

E-SGP Records to X-SGP Records

The ACFESGP conversion utility converts E-SGP records to X-SGP records. See the description of ACFESGP in the "Utilities for eTrust CA-ACF2 Administration" chapter in the *Reports and Utilities Guide*.

eTrust CA-ACF2 provides the following support routines for source grouping:

ACF00SGP (also called ACFSGRP)
This routine provides the name of the first-level source group to which a source belongs.

ACF00SSL
This routine provides the names of all the groups that include a particular source.

ACF00SRL
This routine provides the names of all the groups that include a particular resource.

ACF00SST
This routine determines if a given source is in a specific source group.

For more information about these support routines, see the "Interfacing with eTrust CA-ACF2 chapter in the *Systems Programmer Guide*.

Note 4 to X-RGP Records

The ACFESGP conversion utility converts Note 4 definitions to X-RGP records.

You can use the ACFRGP utility to obtain a list of all the resource groups to which a particular resource belongs. This utility lists the groups in the order that validation occurs. For more information about this utility, see the "Utilities for eTrust CA-ACF2 Administration" chapter in the *Reports and Utilities Guide*.

Activating XREF Records

After you insert or change XREF records, you must activate them using the NEWXREF command. For more information about the NEWXREF command, see the *Systems Programmer Guide*.

Activating X-SGP Records

To activate X-SGP records, issue the following operator command:

```
F ACF2,NEWXREF,TYPE(SGP)
```

where SGP indicates that this is a source group record. This specification activates E-SGP records and X-SGP records.

This command dynamically updates the system's active cross-reference tables for both E-SGP records and X-SGP records. It lets eTrust CA-ACF2 activate X-SGP record fields that have been updated since the last system startup or since the last time this command was issued.

Activating X-RGP Records

To activate X-RGP records, issue the following operator command:

```
F ACF2,NEWXREF,TYPE(RGP)
```

where RGP indicates that this is a resource group record. This specification activates X-RGP records.

This command dynamically updates the system's active cross-reference table for the X-RGP records. It lets eTrust CA-ACF2 activate X-RGP record fields that have been updated since the last system startup or since the last time this command was issued.

Note: To activate resource rules that correspond to new X-RGP groups, you must build a resident directory so that resource group validation processing occurs. Use the REBUILD console operator command to build the directory. See the *Systems Programmer Guide* for information about the REBUILD command.

Chapter 10

Maintaining Shift and Zone Records

Shift records validate access to the system, data sets, and resources based on the time of day you specify. eTrust CA-ACF2 stores shift records in the Infostorage database. They have a type code of SFT and a one to eight-character name. They define the days, dates, and times when users can access the system.

You use zone records to validate system access only. They are stored in the Infostorage database. They have a type code of ZON and a one to three-character name. They let eTrust CA-ACF2 adjust CPU time when it validates shift records across different time zones.

This chapter describes:

- Examples of shift and zone records
- Shift and zone record fields
- Using the ISPF panels for maintaining shift and zone records
- Using the ACF command under the SHIFT setting

Shift Record Example

Here is an example of a shift record:

```
ACF60062 SHIFT REGULAR STORED BY ADMISO ON 01/23/98-10:38
 DAYS(MO,TU,WE,TH,FR) TIME(0800-1700) NTIME(1200-1300)
  INCLUDE(NHOLIDAY)
```

The first line of the display from the ACF command indicates that this shift record, named REGULAR, was created by a security administrator with the logonid ADMISO. Only a user with the SECURITY privilege can create shift records. ADMISO created this record on January 23, 1998, at 10:38 AM.

The parameters on the next two lines specify the days of the week, times of day, and special days to which this record applies. The following list describes these parameters and how eTrust CA-ACF2 interprets them.

DAYS(MO,TU,WE,TH,FR)
The days of the week this record applies to are Monday, Tuesday, Wednesday, Thursday, and Friday. If you specify SHIFT(REGULAR) in a logonid record, it can access the system on Monday through Friday only. If you specify SHIFT(REGULAR) in an access or resource rule entry, eTrust CA-ACF2 allows access to the data set or resource only during the shift named REGULAR.

TIME(0800-1700)
The time of day this record applies to is 08:00 through 16:59 of the days specified in the DAYS parameter. eTrust CA-ACF2 rounds times down to five-minute increments beginning with 00 and ending with 55. For example, if you build a shift record with TIME(0601-1659), access is denied at 1656 because 1659 is rounded down to 1655.

If you specify SHIFT(REGULAR) in a logonid record, that logonid can access the system on Monday through Friday, from 8:00 AM to 4:55 PM only. This means that if the logonid tries to access the system at 4:56 PM, eTrust CA-ACF2 denies the access. If the logonid accesses the system at 4:55 PM, eTrust CA-ACF2 permits the access and does not kick it off the system at 5:00 PM. But, if the logonid logs off the system at 6:00 P.M. and tries to log on again at 6:03 P.M., eTrust CA-ACF2 denies it access.

NTIME(1200-1300)
The shift excludes the lunch hour 12:00 through 12:59 of the specified days. This parameter adjusts the value in the TIME parameter. If you specify SHIFT(REGULAR) in a logonid record, it can access the system on Monday through Friday, from 8:00 AM to 11:55 AM and from 1:00 PM to 4:55 PM only. This means that if the logonid tries to access the system at 12:02 PM, eTrust CA-ACF2 denies it access.

INCLUDE(NHOLIDAY)
The shift includes the days, dates, and times that are specified or excluded by another shift record named NHOLIDAY.

To activate the shift record (REGULAR) to define when a logonid can access the system or system resources, specify the name of the shift record (REGULAR) in the SHIFT field of the logonid record. To activate the shift record (REGULAR) to define when a data set or resource can be accessed, specify the SHIFT keyword and the name of the shift record (REGULAR) in the rule entry.

Zone Record Example

Here is an example of a zone record:

```
ACF60062 ZONE SYD STORED BY ADMISO ON 10/23/98-10:42
 ADJUST(+0600)
```

ZONE SYD
Indicates that the name of this zone record is SYD (to represent a time zone for Sydney, Australia).

ADJUST(+0600)
Defines the time zone as being six hours ahead of the zone in which the CPU is running.

This lets eTrust CA-ACF2 adjust time zones to perform proper shift validation. To activate the zone record (SYD) to define the time zone for users in Sydney, Australia, you must specify SYD in the ZONE field of their logonid records.

Shift and Zone Record Fields

A shift record must contain both a DAYS value and a TIME value. The NDAYS and NTIME fields specify days and hours to exclude from the days and times specified in corresponding DAYS and TIME fields. If you specify NDAY or NTIME, you must also specify DAYS and TIME. Otherwise, there is no value specified for NDAYS or NTIME to exclude from and the shift records allow no access at all.

DAYS
Specifies the days of the week or individual dates when access is granted.

NDAYS
Specifies the days of the week or individual dates when access is not permitted. If you specify NDAYS without specifying a value for DAYS, you create a shift record with no permissible access days.

TIME
Specifies the times during the specified days or dates when access is granted. During eTrust CA-ACF2 processing, times are rounded down to five-minute increments beginning with 00 and ending with 55.

NTIME
Specifies the times during the specified days or dates when access is not permitted. If you specify NTIME without specifying a TIME value, you create a shift record with no permissible entry time. During eTrust CA-ACF2 processing, times are rounded down to five-minute increments beginning with 00 and ending with 55.

INCLUDE
Specifies another shift record that is included as part of this shift record.

A shift can specify multiple days or dates, time periods, and other shifts, as necessary. However, eTrust CA-ACF2 denies access unless one of the day fields and one of the time fields are specified on the shift record.

Zone records can have an ADJUST field to define the offset between the time zone where an access attempt is being made and the time zone where the CPU is running.

Fields that Affect Shift Records

The LOGSHIFT field in a logonid record indicates that the logonid can access the system outside any shift specified in the SHIFT field of the logonid record. The SHIFT parameter in a data set access rule or resource rule restricts access to that data set or resource to the times defined in the corresponding SHIFT record. eTrust CA-ACF2 logs attempted system accesses outside the defined shift and reports them on the Invalid Password/Authority Log (ACFRPTPW). eTrust CA-ACF2 logs attempted accesses to data sets outside the specified shift and reports them on the Data Set/Program Event Log (ACFRPTDS). eTrust CA-ACF2 logs attempted accesses to resources outside the specified shift and reports them on the Generalized Resource Event Log (ACFRPTRV). These reports are described in the *Reports and Utilities Guide*.

Using the ISPF Panels

Select option 17 SHIFT from the eTrust CA-ACF2 Security ISPF Option Selection Menu panel to administer shift and zone records.

```
--------------- eTrust CA-ACF2 Security ISPF OPTION SELECTION MENU -------------
OPTION ===>

  1 RULES   - PROCESS eTrust CA-ACF2 ACCESS AND GENERALIZED RESOURCE RULES
  2 LOGONIDS - eTrust CA-ACF2 LOGONID CREATION/MAINTENANCE FACILITY
  3 SYSTEM  - eTrust CA-ACF2 SHOW COMMANDS
  4 REPORTS  - eTrust CA-ACF2 REPORT PROGRAM PROCESSOR
  5 UTILITIES - PROCESS eTrust CA-ACF2 UTILITIES
  6 GSO     - GLOBAL SYSTEM OPTIONS SERVICES
  7 NET     - NETWORKING SYSTEM OPTIONS SERVICES
  8 CAC     - MVS DATABASE CACHE RECORD SERVICES
  9 XREF    - SOURCE/RESOURCE AND GROUP RECORD SERVICES
 10 MAC     - MANDATORY ACCESS CONTROL ADMINISTRATION
 11 CPF     - COMMAND PROPAGATION FACILITY SERVICES
 12 FIELD   - RECORD LEVEL PROTECTION CONTROLS
 13 TARGETS  - SET CPF TARGET NODES, DEFAULTS IN USE
 14 PROFILE  - PROCESS PROFILE INFORMATION RECORDS
 15 SMS     - PROCESS DFSMS SUPPORT RECORDS
 16 ENTRY   - PROCESS ENTRY SOURCE RECORDS
 17 SHIFT   - PROCESS SHIFT/ZONE RECORDS
 18 RACDCERT - PROCESS KEYRING/CERTIFICATE COMMANDS
 19 C-CIC   - PROCESS C-CIC CICS INITIALIZATION RECORDS
```

The eTrust CA-ACF2 Security Shift/Zone Record Menu panel is displayed:

```
-------------- eTrust CA-ACF2 Security SHIFT/ZONE RECORD MENU ------------------
OPTION ===>

  1 SHIFT   - PROCESS eTrust CA-ACF2 SHIFT RECORDS
  2 ZONE    - PROCESS eTrust CA-ACF2 ZONE RECORDS
```

If you select option 1 SHIFT to process shift records, the eTrust CA-ACF2 Security Shift Record Services panel is displayed:

```
--------------- eTrust CA-ACF2 Security SHIFT RECORD SERVICES ------------------
OPTION ===>

  1 INSERT  - INITIALLY DEFINE SHIFT RECORDS

  2 CHANGE  - CHANGE EXISTING SHIFT RECORDS

  3 LIST    - DISPLAY SHIFT RECORDS

  4 DELETE  - DELETE AN ENTIRE SHIFT RECORD

  5 TARGETS - SET CPF TARGET NODES, DEFAULTS IN USE
```

If you select option 2 ZONE to process zone records, the eTrust CA-ACF2 Security Zone Record Services panel is displayed:

```
-------------- eTrust CA-ACF2 Security ZONE RECORD SERVICES -------------------
OPTION ===>

  1 INSERT  - INITIALLY DEFINE ZONE RECORDS

  2 CHANGE  - CHANGE EXISTING ZONE RECORDS

  3 LIST    - DISPLAY ZONE RECORDS

  4 DELETE  - DELETE AN ENTIRE ZONE RECORD

  5 TARGETS - SET CPF TARGET NODES, DEFAULTS IN USE
```

Note: To process an ACF command at a node other than the default or the current target setting, select the TARGET option and specify the target nodes before you select the INSERT, CHANGE, LIST or DELETE option.

Creating Shift Records

Select option 1 INSERT from the eTrust CA-ACF2 Shift Record Services panel to create a shift record. The Shift Record Administration panel is displayed:

```
------------ eTrust CA-ACF2 Security SHIFT RECORD ADMINISTRATION --------------
COMMAND ===>

 INSERT SHIFT RECORDS

  RECID  ==>


PRESS ENTER TO CONTINUE OR END TO CANCEL.
```

The fields on the panel are described in the following:

RECID
Specify the one to eight-character name of the record ID you want to create.

After entering the record ID of the record you want to create, press ENTER to specify values for the other fields. For a more detailed description of the fields, see the sections Shift and Zone Record Fields and Change Subcommand-Shift Records. In the DAYS and NDAYS fields on the following panels, enter a Y next to MO, TU, WE, TH, FR, SA or SU to add them to the shift record or leave them blank.

```
-------------- eTrust CA-ACF2 Security SHIFT RECORD ADMINISTRATION -------------
COMMAND ===>

 INSERT SHIFT RECORDS

  DAYS MO ==>  TU ==>  WE ==>  TH ==>  FR ==>  SA ==>  SU ==>

       ==>      ,      ,      ,      ,
       ==>      ,      ,      ,      ,
       ==>      ,      ,      ,      ,
       ==>      ,      ,      ,      ,

  TIME   ==>       ,       ,       ,       ,
       ==>       ,       ,       ,       ,
       ==>       ,       ,       ,       ,
       ==>       ,       ,       ,       ,

PRESS ENTER TO CONTINUE OR END TO CANCEL.
```

In the open spaces under DAYS and NDAYS, you can specify dates in the form *mm/dd/yy*, *dd/mm/yy*, or *yy/mm/dd*. You define the format in the DATE field of the GSO OPTS record as described in eTrust CA-ACF2 Option Specifications (OPTS) in the "Maintaining Global System Options Records" chapter of the *Administrator Guide*.

In the open spaces under TIME and NTIME, specify a range of hours and minutes in the form *hhmm-hhmm*, based on a 24-hour clock.

```
-------------- eTrust CA-ACF2 Security SHIFT RECORD ADMINISTRATION -------------
COMMAND ===>

 INSERT SHIFT RECORDS

  NDAYS MO ==>  TU ==>  WE ==>  TH ==>  FR ==>  SA ==>  SU ==>

       ==>      ,      ,      ,      ,
       ==>      ,      ,      ,      ,
       ==>      ,      ,      ,      ,
       ==>      ,      ,      ,      ,

  NTIME   ==>       ,       ,       ,       ,
       ==>       ,       ,       ,       ,
       ==>       ,       ,       ,       ,
       ==>       ,       ,       ,       ,

PRESS ENTER TO CONTINUE OR END TO CANCEL.
```

When you finish specifying date and time information, press ENTER.

To include other shift records as part of the new record you are creating, specify the names of those records on the following panel:

```
------------- eTrust CA-ACF2 Security SHIFT RECORD ADMINISTRATION -------------
COMMAND ===>

 INSERT SHIFT RECORDS

  INCLUDE  ==>     ,     ,     ,     ,
       ==>     ,     ,     ,     ,
       ==>     ,     ,     ,     ,
       ==>     ,     ,     ,     ,

PRESS ENTER TO CONTINUE OR END TO CANCEL.
```

Press ENTER to complete creation of this SHIFT record.

Changing Shift Records

Select option 2 CHANGE from the eTrust CA-ACF2 Security Shift Record Services panel to modify existing shift records. The Change A Shift Record panel is displayed . Specify the type of change you want to make and the record ID to which these changes apply:

```
----------------- eTrust CA-ACF2 Security CHANGE A SHIFT RECORD ----------------
COMMAND ===>

 CHANGE

 CHANGE TYPE  ===> ADD    (ADD, DEL OR REP)

 MASK RECID  ===>          RECID MASKED VALUE

 RECID     ===> TESTSFT
```

Enter specific changes to days and times on this panel and press ENTER:

```
------------- eTrust CA-ACF2 Security SHIFT RECORD ADMINISTRATION -------------
COMMAND ===>

 CHANGE SHIFT RECORDS

  DAYS MO ==>  TU ==>  WE ==>  TH ==>  FR ==>  SA ==>  SU ==>

       ==>     ,     ,     ,     ,
       ==>     ,     ,     ,     ,
       ==>     ,     ,     ,     ,
       ==>     ,     ,     ,     ,

  TIME   ==>       ,       ,       ,       ,
       ==>       ,       ,       ,       ,
       ==>       ,       ,       ,       ,
       ==>       ,       ,       ,       ,

 PRESS ENTER TO CONTINUE OR END TO CANCEL.
```

Enter changes to NDAY and NTIME values on this panel and press ENTER:

```
------------ eTrust CA-ACF2 Security SHIFT RECORD ADMINISTRATION --------------
COMMAND ===>

 CHANGE SHIFT RECORDS

  NDAYS MO ==>  TU ==>  WE ==>  TH ==>  FR ==>  SA ==>  SU ==>

        ==>     ,     ,     ,     ,
        ==>     ,     ,     ,     ,
        ==>     ,     ,     ,     ,
        ==>     ,     ,     ,     ,

  NTIME   ==>       ,       ,       ,       ,
        ==>       ,       ,       ,       ,
        ==>       ,       ,       ,       ,
        ==>       ,       ,       ,       ,

PRESS ENTER TO CONTINUE OR END TO CANCEL.
```

Change any INCLUDE information using this panel. Press ENTER for the changes to take effect.

```
--------------- eTrust CA-ACF2 Security SHIFT RECORD ADMINISTRATION ------------
COMMAND ===>

 CHANGE SHIFT RECORDS

  INCLUDE  ==>     ,     ,     ,     ,
        ==>     ,     ,     ,     ,
        ==>     ,     ,     ,     ,
        ==>     ,     ,     ,     ,

PRESS ENTER TO CONTINUE OR END TO CANCEL.
```

Displaying Shift Records

Select option 3 LIST from the eTrust CA-ACF2 Security Shift Record Services panel to display a shift record. The List a Shift Record panel is displayed:

```
------------------- eTrust CA-ACF2 Security LIST A SHIFT RECORD ---------------
COMMAND ===>

 LIST

 MASK RECID  ===>          RECID MASKED VALUE

 RECID     ===>
```

Specify the name of the record or a masked name of the record you want to display and press ENTER.

Deleting Shift Records

Select option 4 DELETE from the eTrust CA-ACF2 Security Shift Record Services panel to delete a shift record. The Delete a Shift Record panel is displayed:

```
---------------- eTrust CA-ACF2 Security DELETE A SHIFT RECORD ----------------
COMMAND ===>

 DELETE

 MASK RECID  ===>          RECID MASKED VALUE

 RECID     ===>
```

Specify the name of the record or a masked name of the record you wish to delete and press ENTER.

Creating Zone Records

Select option 1 ZONE on the eTrust CA-ACF2 Security Zone Record Services panel to insert zone records. The following panel is displayed:

```
----------- eTrust CA-ACF2 Security SHIFT RECORD ADMINISTRATION ---------------
COMMAND ===>

 INSERT ZONE RECORDS

  RECID  ==>

  ADJUST  ==>
```

After specifying a record ID for this zone record and the time differential that applies for this record, press ENTER to insert this new zone record.

Changing Zone Records

Complete the fields on this panel to modify the information in a particular zone record or a group of zone records.

```
------------------ eTrust CA-ACF2 Security CHANGE A ZONE RECORD ----------------
COMMAND ===>

 CHANGE ZONE RECORDS


 MASK RECID  ===>          RECID MASKED VALUE

 RECID     ===>

 ADJUST    ===>
```

Press ENTER to activate these changes.

You can display zone records by selecting option 3 LIST on the eTrust CA-ACF2 Security Zone Record Services panel and completing the panels that appear.

You can delete zone records by selecting option 4 DELETE on the eTrust CA-ACF2 Security Zone Record Services panel and completing the panels that appear.

Using the ACF Command

To maintain shift or zone records, you must establish the SHIFT setting of the ACF command. To do this, you must specify SHIFT and the SFT type code or the ZON type code.

To maintain shift records, issue the ACF command, establish the SHIFT(SFT) setting, and issue an INSERT subcommand:

```
acf
 ACF
set shift(sft)
 SHIFT
insert regular days(mo,tu,we,th,fr) time(0800-1700)
```

In this example, the INSERT subcommand creates a shift record named REGULAR. This shift is defined as the weekdays Monday through Friday during the hours of 08:00 through 16:59.

To create a zone record, issue the ACF command, establish the SHIFT(ZON) setting, and issue an INSERT subcommand:

```
acf
 ACF
set shift(zon)
 ZONE
insert syd adjust(+0600)
```

In this example, the INSERT subcommand creates a zone record named SYD. This record defines Sydney time as six hours ahead of the CPU time.

After you establish the SHIFT or ZONE setting, you can issue any of the following ACF subcommands:

- ACCESS
- *CHANGE
- CHKCERT
- CONNECT
- *DELETE
- END or QUIT

- EXPORT
- GENCERT
- GENREQ
- HELP
- *INSERT
- *LIST
- REKEY
- REMOVE
- ROLLOVER
- SET or T
- SHOW
- SN

The common subcommands ACCESS, CHKCERT, CONNECT, EXPORT, END, QUIT, GENCERT, GENREQ, REKEY, REMOVE, ROLLOVER, HELP, SET (T), SHOW, and SN operate under all settings. The following sections describe the function, syntax, and parameters of the other subcommands, marked by the asterisks (*).

Since the syntax of the INSERT and CHANGE subcommands are different for processing shift records and zone records, we present the descriptions for both. The DELETE and LIST subcommands use the same syntax to process shift and zone records so we present one description for each subcommand.

INSERT Subcommand—Shift Records

The INSERT subcommand lets you insert new shift records. The INSERT subcommand has the following syntax:

```
Insert  {*|recid}
     {Days(MO,TU,WE,TH,FR,SA,SU,mm/dd/yy,...,mm/dd/yy)}
     {Time(hhmm-hhmm,...,hhmm-hhmm)}
     {NDays(MO,TU,WE,TH,FR,SA,SU,mm/dd/yy,...,mm/dd/yy)}
     {NTime(hhmm-hhmm,...,hhmm-hhmm)}
     [Include(recid1,...,recidn)]
     [TARGET(null|=|?|nodemask,...,nodemask)]
```

Parameter Descriptions

The INSERT subcommand takes the following parameters:

*** (asterisk)**
Specifies the name of the last shift record processed since you established the SHIFT(SFT) setting.

recid
Specifies a one to eight-character name of the shift record to be inserted.

Days(MO,TU,WE,TH,FR,SA,SU,*mm/dd/yy,...,mm/dd/yy*)
Specifies two-character abbreviations for days of the week to be included in the shift (that is, SU for Sunday, MO for Monday, and so forth). This parameter can also include numeric dates in the format *mm/dd/yy*, *dd/mm/yy*, or *yy/mm/dd*. This format is defined in the DATE field of the GSO OPTS record, as described in eTrust CA-ACF2 Option Specifications (OPTS) in the "Maintaining Global System Options Records" chapter of the *Administrator Guide*. Separate multiple days and dates with commas or blank characters. Year designations of 70-99 assume a date in the 20th century (1970-1999); year designations of 00-69 assume a date in the 21st century (2000-2069).

NDays(MO,TU,WE,TH,FR,SA,SU,*mm/dd/yy,...,mm/dd/yy*)
Specifies the days and dates that should be excluded from those specified in the DAYS parameter. If you specify NDAYS, but no value for DAYS, you create a shift record that does not permit access on any day. This parameter can include two-character abbreviations for days of the week and numeric dates in the format *mm/dd/yy*, *dd/mm/yy*, or *yy/mm/dd*. Year designations of 70-99 assume a date in the 20th century (1970-1999); year designations of 00-69 assume a date in the 21st century (2000-2069). Separate multiple days and dates with commas or blank characters.

Time(*hhmm-hhmm,...,hhmm-hhmm*)
Specifies the range of hours and minutes, based on a 24-hour clock, to be included in the shift. Times are converted to five-minute intervals. For example, TIME(0800-1700) represents the hours from 08:00 through 16:59. However, TIME(0800-1659) represents the hours of 08:00 through 16:55. You can specify multiple time entries by separating them with commas or blank characters, for example, TIME(0800-1700,1800-1900). The TIME parameter should be specified, because it defaults to no allowable time period.

NTime(*hhmm-hhmm,...,hhmm-hhmm*)
Specifies the time period to be excluded from that specified by the TIME parameter. If you specify NTIME, but no value for TIME, you create a shift record that does not permit access at any time. For example, if a site wanted to permit access during the hours 09:00 through 16:59 but not during 12:00 through 12:59, the site could specify the shift record parameters:

```
time(0900-1700) ntime(1200-1300)
```

Include(*recid1,...,recidn*)
Specifies any existing shift record to be included as part of the one being defined. For example, a shift record named NORMAL can be included in a shift record named SPECIAL:

```
insert normal days(mo,tu,we,th,fr) time(0900-1700)

insert special include(normal) days(sa)
```

The shift record named SPECIAL grants access not only during weekdays from 09:00 through 16:59, but also during those hours on Saturday.

TARGET(null|=|?|*nodemask,...,nodemask*)
Specifies that you want to process shift records for a target node or group of nodes.

- **null**—indicates that you want to process ACF subcommands at the home node only. Specify SET TARGET() to specify null.
- **=** (equal sign)—indicates that you want to process ACF subcommands at the home node in addition to any node masks defined in this parameter.
- **?** (question mark)—indicates that you want to process ACF subcommands at the default target nodes.
- ***nodemask,...,nodemask***—indicates that you want to process ACF subcommands at the specified target nodes. You can specify up to 100 node names or masks. Use commas or blanks to separate entries.

All shift records must contain a DAYS or NDAYS parameter. Furthermore, eTrust CA-ACF2 does not automatically grant access for days, dates, and times excluded from the NDAYS and NTIMES parameters. You must specify the DAYS or TIMES you want to allow.

Activating Shift Records

When you insert or change any shift record, eTrust CA-ACF2 automatically reloads the new information at midnight (24:00) each day. To reload that information sooner, enter the F ACF2,NEWSHIFT console operator command.

INSERT Subcommand—Zone Records

The INSERT subcommand also lets you insert a zone record. This INSERT subcommand has the following syntax:

```
Insert  {*|recid}
     [Adjust(+hhmm|-hhmm)]
     [TARGET(null|=|?|nodemask,...,nodemask)]
```

Parameter Descriptions

The INSERT subcommand takes the following parameters:

*** (asterisk)**
Specifies that you want to process the last zone record you worked on during this session.

recid
Specifies the name of a one to three-character time zone.

Adjust(+*hhmm*|-*hhmm*)
Specifies a positive or negative value in hours and minutes to be added or subtracted from the processing CPU time. For example, if the processing CPU is in London, the time zone for Sydney might be ADJUST(+0800). A time zone you might specify for New York is ADJUST(-0800).

TARGET(null|=|?|*nodem*
Specifies that you want to process zone records for a target node or group of nodes.

- **null**—indicates that you want to process ACF subcommands at the home node only. Specify SET TARGET() to specify null.
- **=** (equal sign)—indicates that you want to process ACF subcommands at the home node in addition to any node masks defined in this parameter.
- **?** (question mark)—indicates that you want to process ACF subcommands at the default target nodes.
- ***nodemask,...,nodemask***—indicates that you want to process ACF subcommands at the specified target nodes. You can specify up to 100 node names or masks. Use commas or blanks to separate entries.

If you want to process an ACF command at a node other than the default node or the current target setting, you must select the TARGET parameter **before** any other parameters.

CHANGE Subcommand—Shift Records

The CHANGE subcommand lets you add, replace, or delete fields from an existing shift record. This CHANGE subcommand has the following syntax:

```
CHAnge  {*|recid|Like(recidmask)}
     [ADD|REP|DEL]
     {Days(MO,TU,WE,TH,FR,SA,SU,mm/dd/yy,...,mm/dd/yy)}
     {Time(hhmm-hhmm,...,hhmm-hhmmn)}
     {NDays(MO,TU,WE,TH,FR,SA,SU,mm/dd/yy,...,mm/dd/yy)}
     {NTime(hhmm-hhmm,...,hhmm-hhmm)}
     [Include(recid1,...,recidn]
     [TARGET(null|=|?|nodemask,...,nodemask)]
```

For example, the following shift record is named REGULAR:

```
acf
ACF
set shift(sft)
 SHIFT
list regular
 ACF60062 SHIFT REGULAR STORED BY ADMISO ON 10/23/98-10:38
 DAYS(MO,TU,WE,TH,FR) TIME(0800-1700)
```

To add new values to a field in a shift record, you must issue the CHANGE subcommand with at least the name of the record, field, and new value to be added. By default, if you do not specify the ADD, REP, or DEL parameter, eTrust CA-ACF2 assumes the ADD parameter as in the following example:

```
change regular days(sa)
```

This subcommand adds Saturday (SA) to any other days already specified in the DAYS field of the REGULAR shift record.

After you issue the CHANGE subcommand, the system responds with the new contents of the record:

```
list regular
 ACF60062 SHIFT REGULAR STORED BY ADMISO ON 10/23/98-10:38
 DAYS(MO,TU,WE,TH,FR) TIME(0800-1700)

 Change regular days(sa)
 DAYS(MO,TU,WE,TH,FR,SA) TIME(0800-1700)
```

If the value SA had already existed in the record, that value would remain. You can enter the REP and DEL parameters to replace or delete values. The CHANGE subcommand takes the following parameters. The asterisk (*), the record name, and the LIKE, ADD, REP, and DEL parameters are positional. The asterisk or the record name must immediately follow the CHANGE subcommand keyword. If specified, the ADD, REP, or DEL parameter must always come next.

Parameter Descriptions

The CHANGE subcommand takes the following parameters:

*** (asterisk)**
Specifies the name of the last shift record processed since the SHIFT(SFT) setting was established.

recid
Specifies a one to eight-character name of the shift record to be changed.

Like(*recidmask*)
Specifies a mask for the names of the shift records to be changed. To mask the names of shift records, follow the same conventions that apply to logonids, as described in the "Maintaining Logonid Records" chapter of the *Administrator Guide*.

ADD

Indicates that any values you specify for the DAYS, NDAYS, TIME, NTIME, and INCLUDE parameters are added to any existing values in the corresponding shift record fields. If the new value exists in a shift record field, that value remains.

```
list regular
 ACF60062 SHIFT REGULAR STORED BY ADMISO ON 10/23/98-10:38
 DAYS(MO,TU,WE,TH,FR) TIME(0800-1700)

change regular add days(fr,sa) ndays(03/24/98)
 DAYS(MO,TU,WE,TH,FR,SA) NDAYS(03/24/98) TIME(0800-1700)
```

This example of the CHANGE subcommand adds Friday (FR) and Saturday (SA) to the DAYS field of the shift record (Friday already exists in that field so this value remains) and adds an NDAYS field.

REP

Indicates that any values you specify for the DAYS, NDAYS, TIME, NTIME, and INCLUDE parameters replace the corresponding shift record fields. If you specify a new field, the field and its value get added to the record.

```
list regular
 ACF60062 SHIFT REGULAR STORED BY ADMISO ON 10/23/98-10:38
 DAYS(MO,TU,WE,TH,FR) TIME(0800-1700)

change regular rep days(fr,sa) ndays(03/24/98)
 DAYS(FR,SA) NDAYS(03/24/98) TIME(0800-1700)
```

This example of the CHANGE subcommand replaces the other values (MO, TU, WE, TH) with Friday (FR) and Saturday (SA) in the DAYS field of the shift record and adds the NDAYS field.

DEL

Indicates that any values you specify for the DAYS, NDAYS, TIME, NTIME, and INCLUDE parameters get deleted from the corresponding shift record fields. If a value you specify does not exist in a field, eTrust CA-ACF2 ignores that value.

```
list regular
 ACF60062 SHIFT REGULAR STORED BY ADMISO ON 10/23/98-10:38
 DAYS(MO,TU,WE,TH,FR) TIME(0800-1700)

change regular del days(fr) time(0800-1700,2200-2300)
 DAYS(MO,TU,WE,TH)
```

This example of the CHANGE subcommand deletes Friday (FR) from the DAYS field of the shift record. It also deletes the TIME field and ignores the value 2200-2300 because it does not exist.

Days(MO,TU,WE,TH,FR,SA,SU,*mm/dd/yy,...,mm/dd/yy*)
Specifies two-character abbreviations for days of the week (that is, SU for Sunday, MO for Monday, and so forth). This parameter can also include numeric dates in the format *mm/dd/yy*, *dd/mm/yy*, or *yy/mm/dd*. You define the format in the DATE field of the GSO OPTS record, as described in eTrust CA-ACF2 Option Specifications (OPTS) in the "Maintaining Global System Options Records" chapter of the *Administrator Guide*. Year designations of 70-99 assume a date in the 20th century (1970-1999); year designations of 00-69 assume a date in the 21st century (2000-2069). Separate multiple days and dates with commas or blank characters.

NDays(MO,TU,WE,TH,FR,SA,SU,*mm/dd/yy,...,mm/dd/yy*)
Specifies the days and dates that should not be included with those specified in the DAYS parameter. This parameter can include two-character abbreviations for days of the week and numeric dates in the format *mm/dd/yy*, *dd/mm/yy*, or *yy/mm/dd*. Year designations of 70-99 assume a date in the 20th century (1970-1999); year designations of 00-69 assume a date in the 21st century (2000-2069). Separate multiple days and dates with commas or blank characters.

Time(*hhmm-hhmm,...,hhmm-hhmm*)
Specifies a range of hours and minutes, based on a 24-hour clock. For example, TIME(0800-1700) represents the hours from 08:00 through 16:59. You can specify multiple time entries by separating each entry with commas or blank characters, for example, TIME(0800-1700,1800-1900).

NTime(*hhmm-hhmm,...,hhmm-hhmm*)
Specifies the time period to be excluded from that specified by the TIME parameter. You can specify multiple time entries by separating each entry with commas or blank characters, for example, NTIME(0800-1700,1800-1900).

Include(*recid1,...,recidn*)
Specifies any existing shift record to be included as part of the one being defined. For example, a shift record named REGULAR (listed in the following) includes the shift named NHOLIDAY. You can issue a CHANGE subcommand to change the record REGULAR so that it includes the shift OFFTIME instead:

```
list regular
ACF60062 SHIFT REGULAR STORED BY ADMISO ON 10/23/98-10:38
 DAYS(MO,TU,WE,TH,FR) TIME(0800-1700) INCLUDE(NHOLIDAY)

change regular rep include(offtime)
 DAYS(MO,TU,WE,TH,FR) TIME(0800-1700) INCLUDE(OFFTIME)
```

Notice that the INCLUDE parameter replaces the NHOLIDAY shift record with the OFFTIME record.

TARGET(null|=|?|*nodemask,...,nodemask*)

Specifies that you want to process shift records for a target node or group of nodes.

- **null**—indicates that you want to process ACF subcommands at the home node only. Specify SET TARGET() to specify null.
- **=** (equal sign)—indicates that you want to process ACF subcommands at the home node in addition to any node masks defined in this parameter.
- **?** (question mark)—indicates that you want to process ACF subcommands at the default target nodes.
- ***nodemask,...,nodemask***—indicates that you want to process ACF subcommands at the specified target nodes. You can specify up to 100 node names or masks. Use commas or blanks to separate entries.

If you want to process an ACF command at a node other than the default node or the current target setting, you must select the TARGET parameter **before** any other parameters.

CHANGE Subcommand—Zone Records

The CHANGE subcommand lets you change the offset specified in a zone record. This CHANGE subcommand has the following syntax:

```
CHAnge  {*|recid|Like(recidmask)}
     [Adjust(+hhmm|-hhmm)]
     [TARGET(null|=|?|nodemask,...,nodemask)]
```

The following zone record named SYD defines the user's time as being eight hours ahead of the executing CPU time:

```
list syd
 ACF60062 ZONE SYD STORED BY ADMISO ON 10/23/98-10:52
 ADJUST(+0800)
```

To change this offset, issue the CHANGE subcommand with the name of the record and new offset:

```
change syd adjust(+0700)
 ADJUST(+0700)
```

Parameter Descriptions

The CHANGE subcommand takes the following parameters:

*** (asterisk)**
Specifies the name of the last zone record processed since you established the SHIFT(ZON) setting.

recid
Specifies a one to eight-character name of the zone record to be changed.

Like(*recidmask*)
Specifies a mask for the names of the zone records to be changed. To mask the names of zone records, follow the same conventions that apply to logonids.

Adjust(+*hhmm*|-*hhmm*)
Specifies a positive or negative value in hours and minutes to be added or subtracted from the processing CPU time. For example, if the processing CPU is in London, you might specify a time zone for Sydney with the parameter ADJUST(+0800). You might specify a time zone for New York with ADJUST(-0800).

TARGET(null|=|?|*nodemask,...,nodemask*)
Specifies that you want to process zone records for a target node or group of nodes.

- **null**—indicates that you want to process ACF subcommands at the home node only. Specify SET TARGET() to specify null.
- **=** (equal sign)—indicates that you want to process ACF subcommands at the home node in addition to any node masks defined in this parameter.
- **?** (question mark)—indicates that you want to process ACF subcommands at the default target nodes.
- ***nodemask,...,nodemask***—indicates that you want to process ACF subcommands at the specified target nodes. You can specify up to 100 node names or masks. Use commas or blanks to separate entries.

  If you want to process an ACF command at a node other than the default node or the current target setting, you must select the TARGET parameter **before** any other parameters.

LIST Subcommand

The LIST subcommand lets you list shift or zone records. The LIST subcommand has the following syntax:

```
List   {*|recid|Like(recidmask)}
     [TARGET(null|=|?|nodemask,...,nodemask)]
```

Under the SHIFT(SFT) setting, you can list a shift record by issuing the LIST subcommand with the shift record name:

```
list regular
 ACF60062 SHIFT REGULAR STORED BY ADMISO ON 10/23/98-10:38
 DAYS(MO,TU,WE,TH,FR) TIME(0800-1700)
```

Under the SHIFT(ZON) setting, you can list a zone record by issuing the LIST subcommand with the zone record name:

```
list syd
 ACF60062 ZONE SYD STORED BY ADMISO ON 10/23/98-10:52
 ADJUST(+0800)
```

Parameter Descriptions

The LIST subcommand takes the following parameters:

*** (asterisk)**
Specifies the name of the last shift or zone record processed since you established the current SHIFT setting.

recid
Specifies a one to eight-character name of the shift or zone record to be listed.

Like(*recidmask*)
Specifies a mask for the names of the shift or zone records to be listed. To mask the names of shift or zone records, follow the same conventions that apply to logonids, as described in the chapter entitled, "Maintaining Logonid Records."

TARGET(null|=|?|*nodemask,...,nodemask*)
Specifies that you want to process shift and zone records for a target node or group of nodes.

- **null**—indicates that you want to process ACF subcommands at the home node only. Specify SET TARGET() to specify null.
- **=** (equal sign)—indicates that you want to process ACF subcommands at the home node in addition to any node masks defined in this parameter.
- **?** (question mark)—indicates that you want to process ACF subcommands at the default target nodes.
- ***nodemask,...,nodemask***—indicates that you want to process ACF subcommands at the specified target nodes. You can specify up to 100 node names or masks. Use commas or blanks to separate entries.

If you want to process an ACF command at a node other than the default node or the current target setting, you must select the TARGET parameter **before** any other parameters.

DELETE Subcommand

The DELETE subcommand lets you delete shift or zone records, respectively. The DELETE subcommand has the following syntax:

```
DELete  {*|recid|LIke(recidmask)}
     [TARGET(null|=|?|nodemask,...,nodemask)]
```

Under the SHIFT(SFT) setting, you can delete a shift record by issuing the DELETE subcommand with the shift record name:

```
delete regular
 DELETED
```

Parameter Descriptions

The DELETE subcommand takes the following parameters:

*** (asterisk)**
Specifies the name of the last shift or zone record processed since you established the current SHIFT setting.

recid
Specifies a one to eight-character name of the shift or zone record to be deleted.

LIke(*recidmask*)
Specifies a mask for the names of the shift or zone records to be deleted. To mask the names of shift or zone records, follow the same conventions that apply to logonids, as described earlier.

When you issue an online command that deletes multiple records from the eTrust CA-ACF2 databases, eTrust CA-ACF2 prompts you to confirm the DELETE LIKE request. If you respond **Y**, eTrust CA-ACF2 executes the command. If you respond **N** or any other response, eTrust CA-ACF2 ignores the command and prompts you for the next command.

TARGET(null|=|?|*nodemask,...,nodemask*)
Specifies that you want to process shift and zone records for a target node or group of nodes.

- **null**—indicates that you want to process ACF subcommands at the home node only. Specify SET TARGET() to specify null.
- **=** (equal sign)—indicates that you want to process ACF subcommands at the home node in addition to any node masks defined in this parameter.

- **?** (question mark)—indicates that you want to process ACF subcommands at the default target nodes.
- ***nodemask,...,nodemask***—indicates that you want to process ACF subcommands at the specified target nodes. You can specify up to 100 node names or masks. Use commas or blanks to separate entries.

If you want to process an ACF command at a node other than the default node or the current target setting, you must select the TARGET parameter **before** any other parameters.

Chapter 11

Maintaining Structured Infostorage Records

Among other things, structured infostorage records let you:

- Define system options
- Specify record-level protection expressions
- Define extended user authentication records that store information about logonids

Like logonid records, structured infostorage records have different types of fields that you can define and change individually. For example, you can change a bit setting in a structured infostorage record from UADS to NOUADS, just as you can change a logonid privilege from AUDIT to NOAUDIT.

Like shift records, you can change a field in a structured infostorage record, as shown in the following:

```
change opts maxvio(8)
```

Or, you can delete a value from a multi-value field, such as:

```
change resvols volmask(twr-) del
```

Like other infostorage records, eTrust CA-ACF2 stores structured infostorage records in the Infostorage database. However, unlike shift, zone, entry, and scope records, many structured infostorage records have defaults. The defaults are specified in the record structure block (RSB) that defines the record.

Structured infostorage records also let you manipulate records at the field level, unlike unstructured records. Other structured infostorage records are more like scope records. Just as scope records define the limitations of a privileged eTrust CA-ACF2 user, some structured infostorage records define the fields on a screen that a user can update, such as RECORD definition records.

Classifying Structured Infostorage Records

eTrust CA-ACF2 classifies all infostorage records by their class, type code, division, and record ID as described in the following sections. Once you understand how eTrust CA-ACF2 stores and retrieves infostorage records, you can better understand how to maintain them.

Class

Class is an eight-byte name that corresponds to the CLASS field of the GSO APPLDEF record. eTrust CA-ACF2 translates this name into a one-character code that indicates the infostorage class to which this record belongs. Some eTrust CA-ACF2 defined classes include:

C
Control records

D
DB2 records

E
Entry records

F
Field records

I
Identity records

P
Profile records

R
Resource rule records

S
Scope records

T
Shift records

X
Cross-reference records

You can define your class codes using record structure blocks (RSBs) and the GSO APPLDEF record. (For details on creating RSBs, see Creating Structured Infostorage Records in the "Special Usage Considerations" chapter in the *Systems Programmer Guide*.) eTrust CA-ACF2 reserves many class codes for its internal use. You can define class codes using the numbers 0 through 9.

Type Code

The type code is a three-character code that corresponds to the TYPE field of the GSO APPLDEF record. The type code identifies a specific type of record within a class. Some examples of type codes are:

- **SRC**—source records
- **CKC**—CICS transactions
- **GSO**—global system options records

These three type codes belong to different classes: SRC belongs to class E, CKC belongs to class R, and GSO belongs to class C.

Division

Division is part of the infostorage key that is used to allow multiple records within a specific type code. The value of the division code is an arbitrary value selected by the application that uses the record. For instance, the GSO record uses the SYSID as the division code. In the case of DB2 and IMS, the division value is defined via the APPLDEF record associated with these subsystems. The DIVISION option is not a part of the following infostorage records: Classes associated with E(Entry Lists), F(Field Records), R(Resource Rules), S(Shift Records) and T(Time Records) records. SYSID is also not valid for CONTROL(SMS) records.

Record ID

Record ID is a one to 40-byte code that corresponds to the RECID field of the GSO APPLDEF record. The length of the record ID is specified in the RECIDLEN field. eTrust CA-ACF2 allows 40 characters for the division and record ID. Therefore, if the length of the division name is 8 characters, then eTrust CA-ACF2 lets you specify a record ID of up to 32 characters.

For most structured infostorage records, a 16-byte character code defines the name of the record. The RECID for field records can be 24 characters long. The RECID for resource rules is a maximum of 40 characters. Resource rule processing does allow you to specify up to 256 characters when using the rsrcname parameter in a rule entry. For more information, see the "Maintaining Resource Rules" chapter.

What Are SYSIDs?

SYSID and DIVISION are synonymous when used with GSO records. A SYSID is a one to eight-byte character string that specifies the ID of the system to which a structured infostorage record applies. The content of the SYSID string can be arbitrary or it can be an actual SYSID. The SYSID string you select when you start eTrust CA-ACF2 remains in effect until the operator changes it.

How eTrust CA-ACF2 Determines the SYSID

eTrust CA-ACF2 determines the SYSID from the following:

- If eTrust CA-ACF2 was started automatically at system IPL using the CAISEC*xx* and CAIACF*xx* SYS1.PARMLIB members, eTrust CA-ACF2 obtains the SYSID from the SYSID parameter in the CAIACF*xx* member. For more information about using CAISECxx and CAIACF*xx*, see the *Getting Started* guide.
- If eTrust CA-ACF2 was started using a START command at a system console or in the SYS1.PARMLIB COMMND*xx* member, eTrust CA-ACF2 extracts the SYSID string from the SYSID operand of the START command PARM field. So, if you start the system with this command:

  ```
  S ACF2,PARM='SYSID(PROD)'
  ```

  eTrust CA-ACF2 uses PROD as the SYSID.
- If you start eTrust CA-ACF2 with the CAISECxx and CAIACFxx members, but do not include a SYSID statement, or you start eTrust CA-ACF2 with a START command, but omit the SYSID operand, eTrust CA-ACF2 selects the SMF system ID value defined at IPL time with SYS1.PARMLIB(SMFPRM*xx*).
- The operator can change the SYSID anytime after eTrust CA-ACF2 startup using the SETSYS operand of the z/OS MODIFY command. For example, the active SYSID becomes TEST if the operator enters the following command:

  ```
  F ACF2,SETSYS(TEST)
  ```

  This command also refreshes the GSO records.

During ACF command processing, eTrust CA-ACF2 determines the SYSID stored with a record based on these criteria:

- eTrust CA-ACF2 uses the active SYSID string as the SYSID for all records created during a session if the user does not specify a SYSID when he establishes an ACF command setting, such as CONTROL.
- You can specify which SYSID is stored with a record by specifying a SYSID when you establish the ACF command setting. For example, the system might have been started with a SYSID of CPU1, but the user can begin his ACF command session by entering the following subcommand:

  ```
  set control(gso) sysid(cpu2)
  ```

  As a result of this subcommand, the SYSID for all CONTROL(GSO) records that are created, changed, listed, or deleted is CPU2. CPU2 is used until you specify a different SYSID using another SET subcommand, such as

  ```
  set sysid(cpu3)
  ```

  After entering this subcommand, CPU3 is the default SYSID for the remainder of the CONTROL(GSO) setting.
- You can specify the SYSID or MSYSID parameter with any of the ACF subcommands (INSERT, CHANGE, LIST, and DELETE). These parameters override the SYSID default for the execution of that subcommand only. For example, assume the default SYSID is CPU1 when the following subcommand is entered:

  ```
  change pswd minpswd(5) sysid(cpu2)
  ```

  The MINPSWD field change applies to the GSO PSWD record defined for CPU2, not for the GSO PSWD record defined for the system CPU1.

You can also make an individual ACF subcommand apply to multiple SYSIDs by using the MSYSID parameter. For example, if you specify the following SET subcommand:

```
SET CONTROL(GSO) MSYSID(CPU*)
```

ACF command processing applies to records with any four-character SYSID that begins with CPU. An INSERT subcommand creates a GSO OPTS record for CPU1, CPU2, and CPU3.

Sharing Records Among SYSIDs

If your site uses multiple CPUs or separates your environment into separate production and test regions, you can define a different SYSID for each region. This lets you create a different set of structured infostorage records for each SYSID. You might want to share one or more structured infostorage records among systems.

First, you should define the naming conventions for the SYSID string. For example, to start eTrust CA-ACF2 for different CPUs, the z/OS START command might look like the following:

```
S ACF2,PARM='SYSID(CPU1)'
S ACF2,PARM='SYSID(CPU2)'
```

Suppose you have two CPUs that have SYSIDs of CPU1 and CPU2. When eTrust CA-ACF2 was implemented, you created a unique GSO OPTS record for each SYSID. Now, you want them to share a single OPTS record. Here is what you can do to make this change:

- Establish the CONTROL(GSO) setting and mask the SYSID parameter.

  ```
  acf
   ACF
  set control(gso) sysid(cpu*)
   CONTROL
  ```

- Insert a new OPTS record for CPU* using the OPTS record for CPU1.

  ```
  insert using(opts) usysid(cpu1) opts
  ```

- Change the SYSID to CPU1 and delete the OPTS record for CPU1.

  ```
  set control(gso) sysid(cpu1)
   CONTROL
  delete opts
  ```

- Change the SYSID to CPU2 and delete the OPTS record for CPU2.

  ```
  set control(gso) sysid(cpu2)
   CONTROL
  delete opts
   DELETED
  end
  ```

Since both SYSIDs match the mask, the GSO OPTS record we defined for CPU* applies to both CPU1 and CPU2.

You can apply similar concepts to qualified structured infostorage records such as GSO MAINT*qual* and BLPPGM*qual*. For example, a site has two SYSIDs: CPU1 and CPU2. CPU1 has two MAINT records: MAINT.C1A and MAINT.C1B. CPU2 also has two MAINT records: MAINT.C2A and MAINT.C2B. The two systems also share several MAINT records: MAINT.CU1, MAINT.CU2, and MAINT.CU3. You can create these shared records by setting the SYSID to CPU*.

Changing Records for Multiple SYSIDs

You have seen that you can mask the SYSID parameter of the SET subcommand to make a single record apply to more than one system. You can also change multiple structured infostorage records for any CPU that matches a mask, regardless of the SYSID you defined in the SET subcommand. To change multiple records, use the MSYSID parameter.

Suppose a security administrator begins his session by setting the SYSID to CPU1. After he processes some records for the SYSID CPU1, he decides to make a change to the DATE field of the GSO OPTS record on all SYSIDs that match the mask CPU*.

The following are the commands needed to make these updates:

```
set control(gso) sysid(cpu1)
 CONTROL
  .
  .  (other processing)
  .
change opts msysid(cpu*) date(mdy)
```

As you can see, he specifies the MSYSID parameter and a mask of CPU*. This CHANGE subcommand alters the OPTS records on CPU1, CPU2, and CPU3, for example.

The MSYSID parameter is also useful for displaying a particular record on all SYSIDs. For example, to list the OPTS records for all SYSIDs, enter the following commands:

```
acf
 ACF
set control(gso)
 CONTROL
list msysid(-) opts
```

eTrust CA-ACF2 displays the OPTS records for all SYSIDs because the dash masking character was used. The output contains OPTS records for CPU1, CPU2, CPU3, PROD, TEST, and CPU*, for example.

If a structured infostorage record has a qualifier (such as the GSO MAINT record or the CPF NODEDEF record), you can list all MAINT records for all SYSIDs by entering the following commands:

```
acf
 ACF
set control(gso) sysid(cpu1)
 CONTROL
list msysid(-) like(maint-)
```

Using the ? Parameter

If you do not know the current SYSID value, you can use the ? parameter to indicate that you want the next ACF subcommands to apply to the active SYSID.

For example, suppose the operator started eTrust CA-ACF2 using the following command:

```
S ACF2,PARM='SYSID(prod)'
```

You begin processing GSO records using a SYSID of TEST. Any commands you enter apply to TEST SYSID.

```
 ACF
set control(gso) sysid(test)
 CONTROL
```

After processing several commands, you receive a request to change a parameter of the OPTS record for the production environment. Specify a ? for the SYSID value.

```
change sysid(?) opts maxvio(5)
```

However, if you forget what SYSID is active, you can issue the following two subcommands:

```
set sysid(?)
show mode
```

For our example, eTrust CA-ACF2 displays the following:

```
MODE:  CONTROL,  TYPE:  CPF,  SYSID:  PROD
```

Using the ISPF Panels

The ISPF panels for structured infostorage records are described in the chapter where the records are described. For example, you can find the ISPF panels for GSO records in the "Maintaining Global System Options Records" chapter.

Using the ACF Command

After you enter the ACF command, you can issue any of the following subcommands:

- ACCESS
- *CHANGE
- CHKCERT
- CONNECT
- *DELETE
- END or QUIT
- EXPORT
- GENCERT
- GENREQ
- HELP
- *INSERT
- *LIST
- MLSLABEL
- MLWRITE
- REKEY
- REMOVE
- ROLLOVER
- *SHOW
- *SET or T
- SN

The common subcommands ACCESS, CHKCERT, CONNECT, EXPORT, END, QUIT, GENCERT, GENREQ, MLSLABEL, MLWRITE, REKEY, REMOVE, ROLLOVER, HELP, SET, SHOW, and SN operate under all settings. The other subcommands, marked by asterisks (*), are described in the following sections.

SET Subcommand

To maintain structured infostorage records, establish the appropriate ACF command setting and type code.

```
SET or T    {Control(CAc|CPf|Gso|LDS|Net|SMs|Tso)|Identity(Aut)|Xref(RGP|SGP)}
            [TARGET(null|=|?|nodemask1,...,nodemask100)]
            [SYSid(?|sysid)|DIVision(?|div)]
            [MSYSid(sysidmask)|MDIVision(divmask)]
```

Field records are structured infostorage records. However, like resource rules, you use the COMPILE, DECOMPILE, and DELETE subcommands to maintain field records. For more information, see the "Maintaining Field Records" chapter.

Parameter Descriptions

The parameters for the SET subcommand are described as follows:

Control(CAc|CPf|Gso|LDS|Net|SMs|Tso)
Specifies that you want to maintain control records of one of the following types:

- **CAc**—cache facility records
- **CPf**—Command Propagation Facility records
- **Gso**—global system options records
- LDS—LDAP Directory Synchronization Records
- **Net**—distributed database option records
- **SMs**—DFP/SMS system-managed storage records
- **Tso**—TSO option records.

Identity(Aut)
Specifies that you want to maintain identity records for extended user authentication records. You can specify IDENTITY(AUT) only if you have defined a structured infostorage application for identity records. For more information, see the "Maintaining Identity Records" chapter.

Xref(RGP|SGP)
Specifies that you want to maintain cross-reference records for resource groups (RGP) or source groups (SGP).

TARGET(null|=|?|*nodemask1,...,nodemask100*)
Specifies that you want to process structured infostorage records for a target node or group of nodes.

- **null**—indicates that you want to process ACF subcommands at the home node only. Specify SET TARGET() to specify null.
- **=** (equal sign)—indicates that you want to process ACF subcommands at the home node.
- **?** (question mark)—indicates that you want to process ACF subcommands at the default target nodes.
- ***nodemask,...,nodemask***—indicates that you want to process ACF subcommands at the specified target nodes. You can specify up to 100 VTAM node names or masks. Use commas or blanks to separate entries.

SYSid(?|*sysid*)|DIVision(?|*div*)
Specifies the system ID to which the record applies. A SYSID can also represent a routine or group of routines that are used to process user authentication information. It can also be a program or product that your site uses to perform user authentication. Enter the one to eight-character SYSID name or use asterisk (*) to mask the name using standard masking convention. A dash (–) is not an acceptable masking character for SYSID name. DIVISION is a synonym for SYSID.

MSYSid(*sysidmask*)|MDIVision(*divmask*)
Specifies a mask to indicate the group of system IDs to which the record applies. The SYSID mask must contain at least one asterisk or a trailing dash. Structured infostorage records that you process using this parameter have the same record IDs, but different SYSIDs. DIVISION is a synonym for SYSID.

INSERT Subcommand

The INSERT subcommand lets you create a structured infostorage record. The INSERT subcommand has the following syntax:

```
Insert    {*|recid|USING(modelrecid) newrecid}
          [SYSid(?|sysid)|DIVision(?|div)]
          [USYSid(?|sysid)|UDIV(?|div))]
          [TARGET(null|=|?|nodemask,...,nodemask)]
          [field,...,field]
          [ADD|REP|DEL]
```

For example, you can enter the INSERT subcommand with the SYSID, record ID, and fields to be defined in the record:

```
acf
 ACF
set control(gso) sysid(CPU1)
 CONTROL
insert sysid(cpu1) pswd maxtry(2) minpswd(5)
```

This sample subcommand inserts the GSO PSWD record. It defines the password options MAXTRY and MINPSWD. When eTrust CA-ACF2 is running under the SYSID of CPU1, users have a maximum of two attempts to enter the correct password before the TSO session is canceled. Also, passwords must contain at least five characters.

If the SYSID or DIVISION parameter is not specified, then any fields specified are defined for the currently active SYSID.

Parameter Descriptions

The INSERT subcommand accepts the following parameters:

*** (asterisk)**
Specifies the last record you processed while in this ACF command setting. The asterisk specifies only one record. You cannot use the asterisk to represent a record mask.

recid
Specifies the name of the structured infostorage record you want to create. You cannot use masking characters to specify a group of records. You can append a qualifier to some record IDs, for example, APPLDEF*qual*. The length of the qualifier depends on the record type. For GSO records, the qualifier and the record ID together can be up to 16 characters. So, for example, the qualifier for APPLDEF records can be up to 9 characters. For specific information on how to specify record IDs, see the chapter where the record is described. These record IDs are described later in this guide.

USING(*modelrecid*) *newrecid*
Identifies a model record that you want to use to create a new structured infostorage record. All values from the model record are inserted into the new record (except fields where ZERO=YES is specified in the @CFDE macro). Any other fields and values you specify add to, replace, or delete the fields and values in the new record (based on the ADD, REP, or DEL parameter specified).

SYSID(?|*sysid*)|DIVision(?|*div*)
Specifies the one to eight-character SYSID to which this record applies. Specify a question mark (?) to indicate that you want the record to apply to the currently active SYSID.

If you do not specify a SYSID (or USYSID), eTrust CA-ACF2 uses the SYSID value you specified when you established the ACF command setting. If you do not supply a SYSID when you establish the ACF command setting, eTrust CA-ACF2 uses the default value defined at eTrust CA-ACF2 startup time. For more information, see What Are SYSIDs? earlier in this chapter. DIVISION is a synonym for SYSID.

USYSid(?|*sysid*)|UDIV(?|*div*)
Specifies the SYSID of a model record you want to use to create a new structured infostorage record. Specify this record ID with the USING parameter. If you specify a question mark (?) instead of a SYSID, the default SYSID is used. UDIV is a synonym for USYSID.

This time-saving feature lets you insert a record that is similar to a record previously defined for another SYSID. You can change any fields and values that differ between the model and the new record, use the ADD, DEL, or REP parameters (described in the following).

TARGET(null|=|?|*nodemask,...,nodemask*)
Specifies that you want to process structured infostorage records for a target node or group of nodes. Specify node names or masks. Separate entries with commas or blanks.

- **null—i**ndicates that you want to process ACF subcommands at the home node only. Specify SET TARGET() to specify null.
- **=** (equal sign)—indicates you want to process ACF subcommands at the home node in addition to any node masks defined in this parameter.
- **?** (question mark)—indicates you want to process ACF subcommands at the default target nodes.
- ***nodemask,...,nodemask*—**indicates that you want to process ACF subcommands at the specified target nodes. You can specify up to 100 VTAM node names or masks. Use commas or blanks to separate entries.

If you want to process an ACF command at a node other than the default node or the current target setting, you must select the TARGET parameter **before** any other parameters.

field,...,field
Specifies the fields and any values that you want to add, replace, or delete (see the following) to the new record. For the specific field names in each record, see the chapter where the record is described. These general rules apply to field names:

- Turn on bit fields by stating the field name. Turn them off by prefixing the field name with NO. For example, to remove the STC option found in the OPTS record, use NOSTC.
- Specify the desired value in parentheses for fields with variable or character values, for example, TIME(12:30). To nullify a field, specify the field with no value, as in TIME().
- Specify the multiple values in parentheses for fields that are defined with the capacity to contain multiple values. For example, PGMS(IMASPZAP AMASPZAP SUPERZAP). Use a space or comma as a valid delimiter in the parentheses.

ADD
Indicates that you want to add the specified fields and values to those copied from the model record. This parameter applies only to multi-value fields. ADD is the default.

REP
Indicates that you want to replace the fields and values in the model record with the fields and values you specify. This parameter applies only to multi-value fields. If any field you specify is not part of the model record, eTrust CA-ACF2 adds that field and its value to the record.

DEL
Indicates that you want to delete the specified field values from the newly inserted record. This parameter applies only to multi-value fields.

CHANGE Subcommand

The CHANGE subcommand lets you change an existing structured infostorage record. The CHANGE subcommand has the following syntax:

```
CHAnge    {*|recid|LIKE(recidmask)}
          [SYSid(?|sysid)|DIVision(?|div)]
          [MSYSid(sysidmask)|MDIVision(divmask)]
          [TARGET(null|=|?|nodemask,...,nodemask)]
          [field,...,field]
          [ADD|REP|DEL]
```

Note: You cannot specify CHANGE LIKE for individual GSO records. CHANGE LIKE is only applicable for GSO records containing a suffix.

Note: Only the suffix of a GSO record is maskable.

The following CHANGE subcommand alters the BACKUP record so that the time of the automatic backup is scheduled for 02:00:

```
acf
 ACF
set control(gso)
 CONTROL
change backup time(02:00)
```

Since no SYSID is specified, the subcommand affects the BACKUP record for the currently active SYSID.

Parameter Descriptions

The CHANGE subcommand takes the following parameters:

*** (asterisk)**
Specifies that you want to change the last record you processed since you established this ACF command setting. (You cannot use an asterisk to change multiple records, masked records, or the last record processed if the sysid associated with the last record processed is different from the current sysid.

recid
Specifies the record ID that you want to change.

LIKE(*recidmask*)
Specifies a mask for structured infostorage records that you want to change. You cannot abbreviate the LIKE parameter. This masking follows the same conventions that apply to logonids, as described in the chapter entitled, "Maintaining Logonid Records."

SYSid(?|*sysid*)|DIVision(?|*div*)
Specifies the one to eight-character SYSID to which this record applies. Specify a question mark (?) to indicate that you want the record to apply to the currently active SYSID.

If you do not specify a SYSID (or MSYSID), eTrust CA-ACF2 uses the SYSID value you specified when you established the ACF command setting. If you do not supply a SYSID when you establish the ACF command setting, eTrust CA-ACF2 uses the default value defined at eTrust CA-ACF2 startup time. For more information, see What Are SYSIDs? earlier in this chapter. DIVISION is a synonym for SYSID.

MSYSid(*sysidmask*)|MDIVision(*divmask*)
Specifies a mask to indicate the SYSIDs to which this changed record applies. This SYSID mask must contain at least one asterisk or a trailing dash. Structured infostorage records that you change using this parameter have the same record ID, but different SYSIDs. MDIV is a synonym for MSYSID.

TARGET(null|=|?|*nodemask1,...,nodemask100*)
Specifies that you want to process structured infostorage records for a target node or group of nodes.

- **null**—indicates that you want to process ACF subcommands at the home node only.
- **=** (equal sign)—indicates that you want to process ACF subcommands at the home node in addition to any node masks defined in this parameter.
- **?** (question mark)—indicates that you want to process ACF subcommands at the default target nodes.
- ***nodemask,...,nodemask***—indicates the target nodes where you want the ACF subcommands to be processed. You can specify up to 100 VTAM node names or masks. Separate entries with a comma or blanks.

If you want to process an ACF command at a node other than the default node or the current target setting, you must select the TARGET parameter **before** any other parameters.

field,...,field

Specifies the fields and any values that you want to add, replace, or delete (see the following) to the new record. For the specific field names in each record, see the chapter where the record is described.

These general rules apply to field names:

- Turn on bit fields by stating the field name. Turn them off by prefixing the field name with NO. For example, to remove the STC option found in the OPTS record, use NOSTC.
- Specify the desired value in parentheses for fields with variable or character values, for example, TIME(12:30). To nullify a field, specify the field with no value, as in TIME().
- Specify the multiple values in parentheses for fields that are defined with the capacity to contain multiple values. For example, PGMS(IMASPZAP AMASPZAP SUPERZAP). Use a space or comma as a valid delimiter in the parentheses.

ADD

Indicates that you want to add the specified fields and values to those in the existing record. This parameter applies only to multi-value fields. ADD is the default.

REP

Indicates that you want to replace the fields and values in an existing record with the fields and values you specify. This parameter applies only to multi-value fields. If any field you specify is not part of the existing record, eTrust CA-ACF2 adds that field and its value to the record.

DEL

Indicates that you want to delete the specified field values from the existing record. This parameter applies only to multi-value fields.

LIST Subcommand

The LIST subcommand lets you display the contents of a structured infostorage record. The LIST subcommand has the following syntax:

```
List      {*|recid|LIKE(recidmask)}
          [SYSid(?|sysid)|DIVision(?|div)]
          [MSYSid(sysidmask)|MDIVision(divmask)]
          [TARGET(null|=|?|nodemask1,...,nodemask100)]
```

Parameter Descriptions

The LIST subcommand takes the following parameters:

*** (asterisk)**
Specifies that you want to view the last record you processed since you established this ACF command setting. You cannot use the asterisk to display more than one record; use the LIKE parameter for this. You cannot use an asterisk to list the last record processed if the sysid associated with the last record processed is different from the current sysid.

recid
Specifies the record ID of the structured infostorage record that you want to display.

LIKE(*recidmask*)
Specifies a group of structured infostorage records that you want to display. You cannot abbreviate the LIKE parameter. This masking follows the same conventions that apply to logonids, as described in the "Maintaining Logonid Records" chapter.

SYSid(?|*sysid*)|DIVision(?|*div*)
Specifies the one to eight-character SYSID of the record that you want to display. Specify a question mark (?) to indicate that you want the record to apply to the currently active SYSID.

If you do not specify a SYSID (or MSYSID), eTrust CA-ACF2 uses the SYSID value you specified when you established the ACF command setting. If you do not supply a SYSID when you establish the ACF command setting, eTrust CA-ACF2 uses the default value defined at eTrust CA-ACF2 startup time. For more information, see What Are SYSIDs? earlier in this chapter. DIVISION is a synonym for SYSID.

MSYSid(*sysidmask*)|MDIVision(*divmask*)
Specifies a mask to indicate the SYSIDs of the record that you want to display. This SYSID mask must contain at least one asterisk or a trailing dash. Structured infostorage records that you display using this parameter have the same record ID, but different SYSIDs. MDIV is a synonym for MSYSID.

TARGET(null|=|?|*nodemask1,...,nodemask100*)
Specifies that you want to process structured infostorage records for a target node or group of nodes.

- **null**—indicates that you want to process ACF subcommands at the home node only.
- **=** (equal sign)—indicates that you want to process ACF subcommands at the home node.
- **?** (question mark)—indicates that you want to process ACF subcommands at the default target nodes.
- ***nodemask1,...,nodemask100***—indicates the target nodes where you want the ACF subcommands to be processed. You can specify up to 100 VTAM node names or masks. Separate entries with a comma or blanks.

If you want to process an ACF command at a node other than the default node or the current target setting, you must select the TARGET parameter **before** any other parameters.

DELETE Subcommand

The DELETE subcommand lets you delete a structured infostorage record. The DELETE subcommand has the following syntax:

```
DELete    {*|recid|LIke(recidmask)}
          [SYSid(?|sysid)|DIVision(?|div)]
          [MSYSid(sysidmask)|MDIVision(divmask)]
          [TARGET(null|=|?|nodemask1,...,nodemask100)
```

Parameter Descriptions

The DELETE subcommand takes the following parameters:

*** (asterisk)**
Specifies that you want to delete the last record you processed since you established this ACF command setting. You cannot use the asterisk to delete multiple records; use the LIKE parameter for this. You cannot use an asterisk to delete the last record processed if the sysid associated with the last record processed is different from the current sysid.

recid
Specifies the record ID of the structured infostorage record that you want to delete.

LIke(*recidmask*)
Specifies a group of structured infostorage records that you want to delete. This masking follows the same conventions that apply to logonids, as described in the "Maintaining Logonid Records" chapter. Use extreme caution when deleting multiple records with this parameter.

When you issue an online command that deletes multiple records from the eTrust CA-ACF2 databases, eTrust CA-ACF2 prompts you to confirm the DELETE LIKE request. If you respond **Y**, the command executes. If you respond **N** or anything else, eTrust CA-ACF2 ignores the command and prompts for the next command.

SYSid(?|*sysid*)|DIVision(?|*div*)
Specifies the one to eight-character SYSID of the record that you want to delete. Specify a question mark (?) to indicate that you want to delete the record for the currently active SYSID.

If you do not specify a SYSID (or MSYSID), eTrust CA-ACF2 uses the SYSID value you specified when you established the ACF command setting. If you do not supply a SYSID when you establish the ACF command setting, eTrust CA-ACF2 uses the default value defined at eTrust CA-ACF2 startup time. For more information, see What Are SYSIDs? earlier in this chapter. DIVISION is a synonym for SYSID.

MSYSid(*sysidmask*)|MDIVision(*divmask*)
Specifies a mask to indicate the SYSIDs of the record that you want to delete. This SYSID mask must contain at least one asterisk or a trailing dash. Structured infostorage records that you delete using this parameter have the same record ID, but different SYSIDs. MDIV is a synonym for MSYSID.

TARGET(null|=|?|*nodemask1,...,nodemask100*)
Specifies that you want to process structured infostorage records for a target node or group of nodes.

- **null**—indicates that you want to process ACF subcommands at the home node only.
- **=** (equal sign)—indicates that you want to process ACF subcommands at the home node.
- **?** (question mark)—indicates that you to process ACF subcommands at the default target nodes.
- ***nodemask1,...,nodemask100***—indicates the target nodes where you want the ACF subcommands to be processed. You can specify up to 100 VTAM node names or masks. Separate entries with a comma or blanks.

If you want to process an ACF command at a node other than the default node or the current target setting, you must select the TARGET parameter **before** any other parameters.

Displaying System Options and the Active SYSID

The SHOW subcommand displays the currently active SYSID and the fields of a certain structured infostorage record. It also displays many of the system options in effect.

The SHOW subcommand has the following syntax:

```
SHow      [ACF2]
          [ACTive]
          [ALl]
          [APpldef]
          [CErtmap]
          [CLasmap]
          [CPf]
          [CRitmap]
          [DB2]
          [Ddsn]
          [Fields(recid)]
          [LINklst]
          [MLid]
          [MLS[STATUS|ALL|LEVEL|CATEGORIES|SECLABELS(ALPHA|LOW-HI|HI-LOW)]
          [Mode]
          [MUsass]
          [NJe(nodename)]
          [OMVS[ALL|GROUPS(mmmm[-nnnn])|SUPERUSERS|USERS(mmmm[-nnnn])]
          [PGms]
          [PRograms]
          [REsident]
          [RSRctype[(null|D|R)]]
          [RSVwords]
          [SAFdef]
          [STate]
          [SYSPlex]
          [SYstems]
          [TNg]
          [TSo]
          [Unixopts]
          [Zerofld s]
```

The SHOW subcommand accepts only one parameter at a time. All parameters except for MODE and FIELDS operate under the CONTROL(GSO) setting as described in the description of the SHOW subcommand. (See "Overview of eTrust CA-ACF2.")

The MODE and FIELDS parameters are described in the following:

Mode

Displays the current setting of the ACF command, the type code, the currently active SYSID and target node:

```
acf
 ACF
set control(gso)
 CONTROL
show mode

  MODE: CONTROL   TYPE: GSO    SYSID: CPU1  TARGET:  PROD1 NY1 NY2
```

FIELDS(*recid*)
Displays the names of the fields in the logonid or structured infostorage record. The SHOW subcommand display can also indicate which fields can be changed.

```
acf
 ACF
set control(gso)
 CONTROL
show fields(blppgm)

-- BLPPGM --

*LIBRARY   *PGM
```

This subcommand SHOW FIELDS(BLPPGM) displays the names of the fields in the GSO BLPPGM record. Before each field name, an asterisk (*) or plus sign (+) might appear to indicate one of the following:

*** (asterisk)**
Indicates that the user who issued the SHOW subcommand can change this field.

+ (plus sign)
Indicates that the user who issued the SHOW subcommand can change this field and the field might contain multiple values.

If you specify the FIELDS parameter without a record ID, the SHOW FIELDS subcommand displays the names of the fields in the logonid record.

See “Overview of eTrust CA-ACF2” for sample output from the SHOW subcommands.

Chapter 12

Maintaining Cache Records

Using Cache Records

Use cache records to implement the eTrust CA-ACF2 Security database cache facility. The cache facility is a performance feature that speeds eTrust CA-ACF2 validation processing. For example, your site might have several hundred or perhaps a thousand users all logging on to the system between 9:00 AM and 9:10 AM. By using the cache facility, you can reduce the time it takes to perform system entry validation for every user and increase the performance of your system during that 10-minute period each day. Many sites might not need to use the cache facility.

The cache is an area of storage in the eTrust CA-ACF2 address space that contains copies of selected records from the three eTrust CA-ACF2 databases: Infostorage, Rule, and Logonid. The cache facility lets eTrust CA-ACF2 have quick access to these records. When you use the cache facility, you greatly reduce the processing time (I/O time) required to access records stored in the eTrust CA-ACF2 databases.

You implement the cache facility using the GSO OPTS record and the GSO SYNCOPTS record. Specify the eTrust CA-ACF2 database records to be copied into the cache and the cache environment itself, such as the amount of storage it requires in the cache records. Set the CACHE option in the GSO OPTS record to enable the cache facility. To synchronize the cache facility with eTrust CA-ACF2 databases that are shared among CPUs, use the GSO SYNCOPTS record.

Brief descriptions of how the GSO records are used for cache processing are provided in this chapter; however, detailed information about the GSO record fields are provided in the "Maintaining Global System Options Records" chapter. Also, because the process of implementing cache records is nearly identical to the process of implementing GSO records, this chapter cross-references the "Maintaining Global System Options Records" chapter whenever information is applicable to both. This chapter describes the types of cache record and how to administer the cache facility.

Example of a Cache Record

As with the structured infostorage records described in the chapter on "Maintaining Structured Infostorage Records," each cache record consists of three components: the SYSID, the record ID, and the data fields. When listed (displayed on a terminal screen), a cache record is formatted as shown in the following example:

```
CPU1 / CACHOPTS LAST CHANGED BY ADMISO ON 09/15/98-10:03
                    INFORECS(500) LIDRECS(700) MONITOR PAGEPCT(20)
                    RULERECS(900)
```

CPU1
The SYSID associated with this cache record. Use of the SYSID is described later in this chapter.

CACHOPTS
The record ID. The CACHOPTS record is used to define the processing environment of the cache facility.

The information appearing on the second and third lines shows the fields that are used with the CACHOPTS record and their current values. For example, the PAGEPCT field indicates the percentage of auxiliary storage slots that are available for caching.

INFORECS(500)
The number of infostorage records initially copied into the cache.

LIDRECS(700)
The number of logonid records initially copied into the cache.

MONITOR
The cache monitoring feature is active.

PAGEPCT(20)
The percent of auxiliary storage slots that are available for caching.

RULERECS(900)
The number of rule records initially copied into the cache.

Cache Record IDs

Record IDs for cache records are defined by Computer Associates and are not modifiable. You can append qualifiers to some records. The function of each record ID is as follows:

CACHOPTS
Specifies control information for the cache environment.

INFOEXCL
Specifies the infostorage records that are always excluded from the cache. The records specified in the INFOEXCL record are stored only in the eTrust CA-ACF2 Infostorage database; eTrust CA-ACF2 is not permitted to copy these records to the cache. Therefore, normal eTrust CA-ACF2 processing is performed for records specified on the INFOEXCL record.

INFOPRIM
Specifies the infostorage records that are placed into the cache when you initialize the eTrust CA-ACF2 cache. The records you specify are copied from the eTrust CA-ACF2 Infostorage database to the cache.

LIDSEXCL
Specifies the logonid records that are always excluded from the cache. The records specified in the LIDSEXCL record are stored only in the eTrust CA-ACF2 Logonid database; eTrust CA-ACF2 is not permitted to copy these records to the cache. Therefore, normal eTrust CA-ACF2 processing is performed for records specified on the LIDSEXCL record.

LIDSPRIM
Specifies the logonid records that are placed into the cache when you initialize the eTrust CA-ACF2 cache. The records you specify are copied from the eTrust CA-ACF2 Logonid database to the cache.

RULEEXCL
Specifies the data set access rule records that are never cached. The records specified in the RULEEXCL record are stored only in the eTrust CA-ACF2 Rule database; eTrust CA-ACF2 is not permitted to copy these records to the cache. Normal eTrust CA-ACF2 processing is performed for records specified on the RULEEXCL record.

RULEPRIM
Specifies the data set access rule records that are placed into the cache when you initialize the eTrust CA-ACF2 cache. The records you specify are copied from the eTrust CA-ACF2 Rule database to the cache.

CACHE Option Specifications (CACHOPTS)

The CACHOPTS record specifies the processing information used to control the cache environment. It defines the number of records contained in the cache, the amount of storage used for the cache, and whether the cache facility is monitored.

You can define only one CACHOPTS record per CPU. Use the CACHOPTS record only to implement the cache facility; it is not required to implement eTrust CA-ACF2.

If eTrust CA-ACF2 cannot locate a user-defined CACHOPTS record during eTrust CA-ACF2 cache initialization, it issues a prompt on the console screen that requests authority to insert a default CACHOPTS record. Therefore, if you want to use the default field values, you do not need to create the CACHOPTS record.

The record format and field descriptions follow:

Record ID	Fields
CACHOPTS	PAGEPCT(10 | *value*) MONITOR | NOMONITOR INFORECS(2500 | *value*) LIDRECS(5000 | *value*) RULERECS(1000 | *value*)

Field Descriptions

PAGEPCT(10 | *value*)

Indicates the percentage of auxiliary storage slots that are available for caching. The default value is 10, but you can specify any value from 1 through 100. For assistance in determining this value, see your systems programmer and the "Installation Options" chapter in the *Systems Programmer Guide*.

MONITOR | NOMONITOR

Indicates whether the cache monitoring function is active. The default value is NOMONITOR.

The cache monitoring function gathers performance statistics and writes the information to RMF. The monitoring function can be used to obtain RMF status reports on the cache facility. For information regarding use of the RMF Facility, see the "Installation Options" chapter in the *Systems Programmer Guide*.

INFORECS(2500 | *value*)

Specifies an estimate of the initial number of infostorage records to be copied into the cache. After eTrust CA-ACF2 copies the specified number of records into the cache, a secondary allocation of half the initial number takes effect. When the number of records copied into the cache reaches the secondary allocation value, the secondary allocation is allowed again. This process continues until the records exceed the percentage of storage slots available for caching. The default value is 2,500, but you can specify any value from 1 through 16,777,215.

For example, if the initial allocation is 3,000, the secondary allocation is 1,500. After eTrust CA-ACF2 copies the initial and secondary number of records, 4,500, into the cache, it allocates enough storage for 1,500 more records.

LIDRECS(5000 | *value*)

Specifies an estimate of the initial number of logonid records to be copied into the cache. After eTrust CA-ACF2 copies the specified number of records into the cache, a secondary allocation of half the initial number takes effect. When the number of records copied into the cache reaches the secondary allocation value, the secondary allocation is allowed again. This process continues until the records exceed the percentage of storage slots available for caching. The default value is 5,000, but you can specify any number from 1 through 16,777,215.

For example, if the initial allocation is 6,000, the secondary allocation is 3,000. After eTrust CA-ACF2 copies the initial and secondary number of records, 9,000, into the cache, it allocates enough storage for 3,000 more records.

RULERECS(1000 | *value*)

Specifies an estimate of the initial number of data set access rule records to be copied into the cache. After eTrust CA-ACF2 copies the specified number of records into the cache, a secondary allocation of half the initial number takes effect. When the number of records copied into the cache reaches the secondary allocation value, the secondary allocation is allowed again. This process continues until the records exceed the percentage of storage slots available for caching. The default value is 1,000, but you can specify any number from 1 through 16,777,215.

For example, if the initial allocation is 2,500, the secondary allocation is 1,250. After eTrust CA-ACF2 copies the initial and secondary number of records, 3,750, into the cache, enough storage for 1,250 more records is allocated.

Infostorage Exclude Records (INFOEXCL)

The INFOEXCL record specifies the infostorage records that are never copied into the cache.

If you need more than one INFOEXCL record, you can append a qualifier to the record name in the format INFOEXCL*qual* to generate a unique record ID (for example, INFOEXCL001 or INFOEXCL.esrc). This optional qualifier can be up to eight characters, and must immediately follow the characters INFOEXCL. If you use a period (.) as part of the qualifier string for the record name, eTrust CA-ACF2 counts it as one of the eight characters.

You do not need to specify an INFOEXCL record.

The record format and field descriptions follow:

Record ID	Fields
INFOEXCL*qual*	APPLS(*appl1,appl2,…,appl100*)

Field Descriptions

APPLS(*appl1,appl2,...,appl100*)

Specifies up to 100 application names that are not to be copied into the cache. You must separate the names with commas. Each application name consists of a specific record class code and type code, a class code and a mask, or a masking character (asterisk (*) or dash (–)). Standard eTrust CA-ACF2 masking conventions apply.

For example, if you want to prevent GSO records from being cached, enter **CGSO** where **C** is the control class of records, and **GSO** indicates the type code for Global System Options records. If you want to prevent all control records from being cached, enter **C–,** where **C** is the control class of records, and **–** is the masking symbol that indicates that all records that belong to the control class (for example, GSO and TSO) are not copied into the cache. If you want to prevent all infostorage records from being placed in the cache, enter **–**.

Logonid Exclude Records (LIDSEXCL)

The LIDSEXCL record specifies the logonid records that are never copied into the cache.

If you need more than one LIDSEXCL record, you can append a qualifier to the record name in the format LIDSEXCL*qual* to generate a unique record ID (for example, LIDSEXCL001 or LIDSEXCL.adm). This optional qualifier can be up to eight characters, and must immediately follow the characters INFOEXCL. If you use a period (.) as part of the qualifier string for the record name, eTrust CA-ACF2 counts it as one of the eight characters.

You do not need to specify a LIDSEXCL record.

The record format and field descriptions follow:

Record ID	Fields
LIDSEXCL*qual*	LIDS(*logonid1,logonid2,…,logonid50*)

Field Descriptions

LIDS(*logonid1,logonid2,...,logonid50*)
Specifies up to 50 logonid records that you do not want to be copied into the cache. You must separate the logonids with commas. Each logonid is an eight-character maskable value. Thus, you can define a specific logonid, or you can define a group of logonids using standard eTrust CA-ACF2 masking conventions.

For example, if you want to prevent all of the accounting department's logonids from being cached, specify **ACC–**. If you want to exclude only one logonid that belongs to a specific accountant, specify **ACCLS,** where **ACC** represents the accounting department and **LS** represents Larry Smith, a particular accountant.

Rule Exclude Records (RULEEXCL)

The RULEEXCL record specifies the data set access rule records that are never copied into the cache.

If you need more than one RULEEXCL record, you can append a qualifier to the record name in the format RULEEXCL*qual* to generate a unique record ID (for example, RULEEXCL001 or RULEEXCL.sys1). This optional qualifier can be up to eight characters, and must immediately follow the characters INFOEXCL. If you use a period (.) as part of the qualifier string for the record name, eTrust CA-ACF2 counts it as one of the eight characters.

You do not need to specify a RULEEXCL record.

The record format and field descriptions follow:

Record ID	Fields
RULEEXCL*qual*	INDEX(*index1,index2,…,index50*)

Field Descriptions

INDEX(*index1,index2,...,index50*)
Specifies up to 50 high-level index names that you do not want to be copied into the cache. You must separate the indexes with commas. Each name is an eight-character maskable value. Thus, you can define a specific data set access rule record, using its full high-level index name (the $KEY value), or you can define a group of rule records using standard eTrust CA-ACF2 masking conventions.

For example, if you want to prevent all of the payroll rule records from being cached, specify **PAY–**. If you want to exclude only the accountants' payroll rule record, specify **ACCPAY** where **ACC** represents the accounting department, and **PAY** represents the accounting department's payroll records. For more information about high-level indexes, see the "Maintaining Access Rules" chapter.

Infostorage Prime Records (INFOPRIM)

The INFOPRIM record specifies the infostorage records that are copied into the cache during eTrust CA-ACF2 cache initialization.

If you need more than one INFOPRIM record, you can append a qualifier to the record name in the format INFOPRIM*qual* to generate a unique record ID (for example, INFOPRIM001 or INFOPRIM.esrc). This optional qualifier can be up to eight characters, and must immediately follow the characters INFOPRIM. If you use a period (.) as part of the qualifier string for the record name, eTrust CA-ACF2 counts it as one of the eight characters.

You do not need to specify an INFOPRIM record.

The record format and field descriptions follow:

Record ID	Fields
INFOPRIM*qual*	APPLS(*appl1,app12,…,appl100*)

Field Descriptions

APPLS(*appl1,appl2,...,appl100*)
Specifies up to 100 infostorage application names that you want copied into the cache. You must separate the names with commas. Each application name consists of a specific record class code and type code, a class code and a mask, or a masking symbol (asterisk (*) or dash (–)). Standard eTrust CA-ACF2 masking conventions apply.

For example, if you want to cache source records, enter **ESRC** where **E** is the entry class of records, and **SRC** indicates the type code for source records. If you want to cache all entry records, enter **E–**,where **E** is the entry class of records, and **–** is the masking symbol that indicates that all records that belong to the entry class (that is, source or source group) are copied into the cache.

Logonid Prime Records (LIDSPRIM)

The LIDSPRIM record specifies the logonid records that are copied into the cache during eTrust CA-ACF2 cache initialization.

If you need more than one LIDSPRIM record, you can append a qualifier to the record name in the format LIDSPRIM*qual* to generate a unique record ID (for example, LIDSPRIM001 or LIDSPRIM.adm). This optional qualifier can be up to eight characters, and must immediately follow the characters LIDSPRIM. If you use a period (.) as part of the qualifier string for the record name, eTrust CA-ACF2 counts it as one of the eight characters.

You do not need to specify a LIDSPRIM record.

The record format and field descriptions follow:

Record ID	Fields
LIDSPRIM*qual*	LIDS(*logonid1,logonid2,…,logonid50*)

Field Descriptions

LIDS(*logonid1,logonid2,...,logonid50*)
Specifies up to 50 logonid records that you want copied to the cache. You must separate the logonids with commas. Each logonid is an eight-character maskable value. Thus, you can define a specific logonid or you can define a group of logonids using standard eTrust CA-ACF2 masking conventions.

For example, if you want to cache all of the data processing department's logonids, specify **DP–**. If you want to include only one logonid belonging to a specific programmer, specify **DPJA** where **DP** represents the data processing department and **JA** represents Jane Anderson, a particular programmer.

Rule Prime Records (RULEPRIM)

The RULEPRIM record specifies the data set access rule records that are copied into the cache during eTrust CA-ACF2 cache initialization.

If you need more than one RULEPRIM record, you can append a qualifier to the record name in the format RULEPRIM*qual* to generate a unique record ID (for example, RULEPRIM001 or RULEPRIM.sys1). This optional qualifier can be up to eight characters, and must immediately follow the characters RULEPRIM. If you use a period (.) as part of the qualifier string for the record name, eTrust CA-ACF2 counts it as one of the eight characters.

You do not need to specify a RULEPRIM record.

The record format and field descriptions follow:

Record ID	Fields
RULEPRIM*qual*	INDEX(*index1,index2,…,index50*)

Field Descriptions

INDEX(*index1,index2,...,index50*)
Specifies up to 50 high-level index names that you want copied into the cache. You must separate the indexes with commas. Each name is an eight-character maskable value. Thus, you can define a specific data set access rule record, using its full high-level index name (the $KEY value), or you can define a group of rule records using standard eTrust CA-ACF2 masking conventions.

For example, if you want to cache all of the inventory rule records, specify **INV–**. If you want to cache only one specific subset of inventory rule records, specify INV002 where INV002 is the $KEY value of the rule record you want to cache. For more information about high-level indexes, see the "Maintaining Access Rules" chapter.

SYSID Use

CONTROL(CAC) records use the SYSID parameter in the same manner as all other structured infostorage records (for example, GSO records). This section provides examples of SYSID use for cache records. For detailed information about SYSID concepts and use, see the "Maintaining Structured Infostorage Records" chapter.

The following is an example of the ACF command syntax used to insert a cache record with a default SYSID string:

```
acf
 ACF
set control(cac) sysid(cpu1)
 CONTROL SYSID: CPU1
insert lidsprim lids(accls,dp-)
```

You can also specify the SYSID on the same line as the INSERT subcommand.

```
acf
 ACF
set control(cac)
 CONTROL
insert sysid(cpu1) lidsprim lids(accl,dp-)
```

Creating and Maintaining Cache Records

Define cache record options to eTrust CA-ACF2 using the ISPF panels or by establishing the CONTROL(CAC) setting of the ACF command. You can insert, change, list, and delete records using the ACF subcommands.

Logonid Privileges Required to Implement Cache Records

To create or maintain cache records, you must have the SECURITY field specified in your logonid record. For information about the SECURITY logonid field, see the "Maintaining Logonid Records" chapter.

Refreshing Cache Records

You can refresh cache records at cache startup and after the cache is activated. When you start the cache, eTrust CA-ACF2 automatically refreshes all the cache records, including the CACHOPTS record.

When the cache is active, you can refresh the CACHOPTS, INFOEXCL, LIDSEXCL, and RULEEXCL cache records with the REFRESH command.

These examples illustrate how to refresh the cache records when the cache is active:

```
F ACF2,REFRESH(ALL),TYPE(CAC)
```

or

```
F ACF2,SETTYPE(CAC)
F ACF2,REFRESH(ALL)
```

The REFRESH command never refreshes the INFOPRIM, LIDSPRIM, and RULEPRIM records. To refresh all cache records, you must stop and start the cache facility or eTrust CA-ACF2.

To refresh only the CACHOPTS and LIDSEXCL records, use the following console operator commands:

```
F ACF2,REFRESH(CACHOPTS,LIDSEXCL),TYPE(CAC)
```

Or

```
F ACF2,SETTYPE(CAC)
F ACF2,REFRESH(CACHOPTS,LIDSEXCL)
```

The REFRESH command can specify one or any combination of the CACHOPTS, INFOEXCL, LIDSEXCL, or RULEEXCL cache records.

Stopping and starting the cache facility is described later in this chapter in Using the CACHE Console Operator Command.

For information about the REFRESH process, see the description about eTrust CA-ACF2 startup information in the "Installation Options" chapter in the *Systems Programmer Guide*. For syntax requirements for the REFRESH operator command, see Appendix C, "Console Operator Commands" in the *Systems Programmer Guide*.

Using the ISPF Panels

From the eTrust CA-ACF2 Security ISPF Option Selection Menu, select option 8 CAC.

```
-------------- eTrust CA-ACF2 Security ISPF OPTION SELECTION MENU --------------
OPTION  ===>

   1  RULES     - PROCESS eTrust CA-ACF2 ACCESS AND GENERALIZED RESOURCE RULES
   2  LOGONIDS  - eTrust CA-ACF2 LOGONID CREATION/MAINTENANCE FACILITY
   3  SYSTEM    - eTrust CA-ACF2 SHOW COMMANDS
   4  REPORTS   - eTrust CA-ACF2 REPORT PROGRAM PROCESSOR
   5  UTILITIES - PROCESS eTrust CA-ACF2 UTILITIES
   6  GSO       - GLOBAL SYSTEM OPTIONS SERVICES
   7  NET       - NETWORKING SYSTEM OPTIONS SERVICES
   8  CAC       - MVS DATABASE CACHE RECORD SERVICES
   9  XREF      - SOURCE/RESOURCE AND GROUP RECORD SERVICES
  10  MAC       - MANDATORY ACCESS CONTROL ADMINISTRATION
  11  CPF       - COMMAND PROPAGATION FACILITY SERVICES
  12  FIELD     - RECORD LEVEL PROTECTION CONTROLS
  13  TARGETS   - SET CPF TARGET NODES, DEFAULTS IN USE
  14  PROFILE   - PROCESS PROFILE INFORMATION RECORDS
  15  SMS       - PROCESS DFSMS SUPPORT RECORDS
  16  ENTRY     - PROCESS ENTRY SOURCE RECORDS
  17  SHIFT     - PROCESS SHIFT/ZONE RECORDS
  18  RACDCERT  - PROCESS KEYRING/CERTIFICATE COMMANDS
  19  C-CIC     - PROCESS C-CIC CICS INITIALIZATION RECORDS
  20  LDS       - PROCESS LDAP DIRECTORY SERVICES
```

The eTrust CA-ACF2 Security MVS Database Cache Record Services panel is displayed:

```
--------- eTrust CA-ACF2 Security MVS DATABASE CACHE RECORD SERVICES -----------
OPTION  ===>

   1  INSERT   - INITIALLY DEFINE CAC RECORDS

   2  CHANGE   - CHANGE EXISTING CAC RECORDS

   3  LIST     - DISPLAY CAC RECORDS AND PARAMETERS

   4  DELETE   - DELETE AN ENTIRE CAC RECORD

   5  SHOW ALL - SHOW ALL ACFFDR AND CAC OPTIONS IN EFFECT

   6  FIELDS   - SHOW CAC RECORD FIELD NAMES

   7  TARGETS  - SET CPF TARGET NODES, DEFAULTS IN USE
```

If you want to process an ACF command at a node other than the default node or the current target setting, you must select the TARGET option and specify the target nodes **before** you select the INSERT, CHANGE, LIST, or DELETE options.

Creating Cache Records

Select option 1 INSERT on the eTrust CA-ACF2 Security MVS Database Cache Record Services panel to create a cache record. The Insert A Cache Record panel is displayed:

```
------------------ eTrust CA-ACF2 Security INSERT A CACHE RECORD ---------------
COMMAND ===>

  INSERT

  CHANGE TYPE   ===> ADD       (ADD, DEL, REP)

  SYSID         ===>           SYSTEM ID FOR CACHE RECORD

  USING SYSID   ===>           OPTIONAL SYSTEM ID FOR COPY OF EXISTING VALUES

  USING RECID   ===>                   OPTIONAL PROTOTYPE RECORD NAME

  RECID         ===>          (CACHOPTS, INFOEXCL, INFOPRIM, LIDSEXCL,
                               LIDSPRIM, RULEEXCL, RULEPRIM)

 SPECIFICATION OF RECID WILL RESULT IN DISPLAY OF APPROPRIATE PANEL
 FOR OPTION SPECIFIED
```

Panel Field Descriptions

CHANGE TYPE
Specify one of the following types of changes to the record:

- **ADD**—adds the fields to an existing record.
- **DEL**—deletes the fields from an existing record.
- **REP**—replaces the fields in an existing record with the fields you specify.

SYSID
Specify the system ID for the cache record that you want to create.

USING SYSID
Specify a system ID of another cache record that you want to use to create a cache record on the current system.

USING RECID
Specify the name of a record that you want to use as a model to create this record.

RECID
Specify the one to eight-character name of the record ID that you want to create.

Changing Cache Records

To change a cache record, select option 2 CHANGE on the eTrust CA-ACF2 Security MVS Database Cache Record Services panel. The Change A Cache Record panel is displayed:

```
---------------- eTrust CA-ACF2 Security CHANGE A CACHE RECORD ----------------
COMMAND ===>

  CHANGE

  CHANGE TYPE   ===> ADD       (ADD, DEL, REP)

  SYSID         ===>           SYSTEM ID FOR CAC RECORD

  MASK SYSID    ===>           MASKED SYSTEM ID FOR CAC RECORD

  MASK RECID    ===>                   RECID MASKED VALUE

  RECID         ===>          (CACHOPTS, INFOEXCL, INFOPRIM, LIDSEXCL,
                               LIDSPRIM, RULEEXCL, RULEPRIM)

 SPECIFICATION OF RECID WILL RESULT IN DISPLAY OF APPROPRIATE PANEL
 FOR OPTION SPECIFIED
```

Panel Field Descriptions

CHANGE TYPE
Specify one of the following types of changes to the record:

- **ADD**—adds the fields to an existing record.
- **DEL**—deletes the fields from an existing record.
- **REP**—replaces the fields in an existing record with the fields you specify.

SYSID
Specify the system ID for the cache record that you want to change.

MASK SYSID
Specify a mask to indicate the system IDs of the records to which this change applies.

MASK RECID
Specify a mask to indicate the records to which this change applies.

RECID
Specify the one to eight-character name of the record ID that you want to change.

Displaying Cache Records

Select option 3 LIST on the eTrust CA-ACF2 Security MVS Database Cache Record Services panel to display cache records. The eTrust CA-ACF2 List A Cache Record panel is displayed:

```
---------------- eTrust CA-ACF2 Security LIST A CACHE RECORD ------------------
COMMAND ===>

  LIST

  SYSID         ===           SYSTEM ID FOR CAC RECORD

  MASK SYSID    ===>           MASKED SYSTEM ID FOR CAC RECORD

  QUALIFIER     ===>                   CAC RECORD QUALIFIER

  MASK RECID    ===>                   RECID MASKED VALUE

  RECID         ===>          (CACHOPTS, INFOEXCL, INFOPRIM, LIDSEXCL,
                               LIDSPRIM, RULEEXCL, RULEPRIM)
```

Panel Field Descriptions

SYSID
Specify the system ID for the cache record that you want to list.

MASK SYSID
Specify a system ID mask to display cache records for a group of systems.

QUALIFIER
Specify a qualifier to uniquely define the record.

MASK RECID
Specify a record name mask to list a group of cache records.

RECID
Specify the one to eight-character name of the record ID that you want to list.

DopxIM, RULEEXCL, RULEPRIM)

Panel Field Descriptions

SYSID
Specify the system ID for the cache record that you want to delete.

MASK SYSID
Specify a system ID mask to delete cache records for a group of systems.

QUALIFIER
Specify a qualifier to uniquely define the record.

MASK RECID
Specify a record name mask to delete a group of cache records.

RECID
Specify the one to eight-character name of the record ID that you want to delete.

Displaying Cache Options

Selecting option 5 SHOW ALL on the eTrust CA-ACF2 Security MVS Database Cache Record Services panel issues an eTrust CA-ACF2 SHOW ALL subcommand to display system information which includes the active cache options for that system. Sample panel displays for the SHOW subcommands are provided in "Overview of eTrust CA-ACF2."

Displaying Field Names for a Cache Record

Select option 6 FIELDS on the eTrust CA-ACF2 MVS Database Cache Record Services panel to display the current system's field names for a cache record. Specify the cache record (CACHOPTS, INFOEXCL, INFOPRIM, LIDSEXCL, LIDSPRIM, RULEEXCL, or RULEPRIM) for which you want the field names displayed at the following prompt:

```
ENTER CAC RECORD NAME _
```

For help information, press ENTER without specifying anything at the prompt. The fields for the various cache records are described earlier in this chapter. See the appropriate record for field descriptions.

Setting Target Nodes

Select option 7 TARGETS on the eTrust CA-ACF2 MVS Database Cache Record Services panel to set the target nodes where you want the commands to take effect. The Set CPF Target Nodes panel is displayed:

```
--------------------------- SET CPF TARGET NODES ----------------------------
COMMAND ===>

  SET THE CPF TARGET NODE(S) FOR ALL LOGONID AND INFOSTORAGE DATABASE
  PROCESSING.  THE TARGET NODE LIST IS RETAINED ACROSS ISPF SESSIONS.

  THE NODE NAMES MAY BE MASKED.  IF THEY ARE BLANK, eTrust CA-ACF2 WILL USE
  THE DEFAULT TARGET NODES.  ENTERING THE LAST ENTRY WILL PROMPT FOR
  ADDITIONAL LINES.

          CMDWAIT   ===>      (Y,N)  SYNCHRONOUS/ASYNCHRONOUS
                    ===>              ===>
                    ===>              ===>
                    ===>              ===>
                    ===>              ===>
                    ===>              ===>
                    ===>              ===>
                    ===>              ===>
                    ===>              ===>
                    ===>              ===>
                    ===>              ===>
```

Enter up to 100 target nodes. The CPF address space must be active on each node and you must have defined the nodes in CPF NODEDEF and OPTIONS records for the commands to take effect.

Using the ACF Command

Use the same ACF subcommands and syntax to create and maintain cache records as you use to create and maintain any structured infostorage record. Specifically, the ACF subcommands supported by the cache facility are the following:

- CHANGE
- DELETE
- END or QUIT
- HELP
- INSERT
- LIST
- SET or T
- SN

This section provides an example of how to use an ACF subcommand to list a cache record. For detailed information about how to use the ACF subcommands, see the "Maintaining Structured Infostorage Records" chapter. The following is an example of the ACF command syntax used to list a cache record:

```
acf
 ACF
set control(cac)
 CONTROL
list sysid(cpu1) lidsprim
```

Activating the Cache Facility

To activate cache records, you must set the cache field of the GSO OPTS record for the system using the cache facility. Be sure to specify CACHE when you implement the GSO OPTS record because NOCACHE is the default.

When you initialize eTrust CA-ACF2, eTrust CA-ACF2 reads the GSO OPTS record. If you specify CACHE, eTrust CA-ACF2 builds the cache facility using the field values you specify in the CACHOPTS record. After eTrust CA-ACF2 builds the cache facility, eTrust CA-ACF2 copies the eTrust CA-ACF2 database records you specify in the CAC prime records (INFOPRIM, LIDSPRIM, and RULEPRIM) into the cache. The database records specified in the INFOEXCL, LIDSEXCL, and RULEEXCL records are not copied into the cache.

You can activate the cache facility after eTrust CA-ACF2 has been initialized. First, modify the GSO OPTS record. The following example illustrates how to modify the GSO OPTS record:

```
acf
 ACF
set control(gso) sysid(cpu1)
 CONTROL
change opts cache
```

Next you must enter a REFRESH command to refresh the GSO OPTS record and enable the cache facility. This example illustrates how to refresh the GSO OPTS record.

```
F ACF2,REFRESH(OPTS),SYSID(cpu1)
```

After you modify and refresh the GSO OPTS records, activate the cache facility with this command:

```
F ACF2,CACHE(START)
```

Synchronizing the Cache Facility

eTrust CA-ACF2 provides a function that ensures that all caches are synchronized with the eTrust CA-ACF2 databases. If your site has CPUs sharing eTrust CA-ACF2 databases and at least one of the CPUs is using the cache facility, you should activate the cache synchronizer for each of the CPUs. The cache synchronizer ensures that records in the cache are updated whenever the corresponding records in the eTrust CA-ACF2 databases are updated. If your site's CPUs do not share eTrust CA-ACF2 databases, you do not need to activate the cache synchronizer.

Every cache synchronizer uses a common synchronization file shared by the CPUs. When you activate the synchronizers, they add, read, and delete entries in the common synchronization file. These entries are the eTrust CA-ACF2 record IDs of the shared database records that must be updated in the caches of the other CPUs. Records in the caches are synchronized with the records in the shared databases. Each CPU uses the same records for eTrust CA-ACF2 processing.

GSO Records Used for Synchronization

eTrust CA-ACF2 uses the values you specify in the GSO SYNCOPTS record to determine whether to activate a synchronizer. The synchronizer uses the values you specify in this record to determine the time interval that elapses between accesses to the synchronization file. You also specify, in the GSO SYNCOPTS record, the maximum number of reads permitted for each entry in the synchronization file before an entry is deleted. The eTrust CA-ACF2 default GSO SYNCOPTS record specifies NOACTIVATE. So, if you want to use the synchronizer, you must define a GSO SYNCOPTS record with ACTIVATE.

To enable eTrust CA-ACF2 to activate a synchronizer, set the SHRDASD field of the GSO OPTS record. This field indicates that your site's CPUs are sharing a eTrust CA-ACF2 database.

For specific information about defining the GSO SYNCOPTS and the GSO OPTS records, see the "Maintaining Global System Options Records" chapter.

Using the CACHE Console Operator Command

This section describes the purpose of each option of the CACHE command.

Command Syntax

Use the standard eTrust CA-ACF2 console operator command syntax to enter the cache facility console operator options.

```
F ACF2,CACHE(option)
```

Option Descriptions

You can use the following options with the CACHE command:

STOP

Terminates cache facility processing. When you execute the STOP option, all queued work associated with the cache facility is completed before the cache facility stops. When the cache facility stops, eTrust CA-ACF2 versions all cache-related storage in its eTrust CA-ACF2 address space. When the cache facility is inactive, all standard eTrust CA-ACF2 processing occurs.

Using the STOP and START options eliminates the need to reinitialize eTrust CA-ACF2 after CAC prime records (INFOPRIM, LIDSPRIM, and RULEPRIM) have been updated.

START

Activates the cache facility. When you execute the START option, eTrust CA-ACF2 refreshes all cache records. It activates the new and updated cache records on the system. It builds the cache, it copies the database records you specify in the CAC prime records into the cache, and it activates cache facility processing.

Using the STOP and START options eliminates the need to reinitialize eTrust CA-ACF2 after CAC prime records (INFOPRIM, LIDSPRIM, and RULEPRIM) have been updated.

STATUS

Provides information about the cache facility. When you execute the STATUS command, eTrust CA-ACF2 displays the following types of information on the console screen:

- Whether the cache synchronizer is active
- Whether the cache is currently active
- Whether the cache is completely primed
- The number of records primed

- The number of records in the cache
- The number of reads and writes processed
- Service times

SYNCRESET

Removes a hold on the synchronization file and releases the file for use. Use the SYNCRESET option after the following message has appeared repeatedly:

```
ACFCC202  ACF2 CACHE SYNCHRONIZATION DATA SET LOCKED BY ANOTHER COMPLEX
```

This message indicates that the synchronization file has been locked by another complex and is not released.

DEMAND|NODEMAND

Lets you prevent the cache facility from using too much page space due to demand caching. Demand caching occurs when the cache facility cannot find a record in the cache. The cache facility retrieves the record from the eTrust CA-ACF2 database and puts it in the cache. The record remains in the cache for future processing. If the required record is specified on a cache exclude record (INFOEXCL, LIDSEXCL, or RULEEXCL), it is not copied into the cache.

The cache facility automatically performs demand caching unless you execute the NODEMAND option. The NODEMAND option terminates demand caching. Use DEMAND to reactivate demand caching.

MONITOR|NOMONITOR

Lets you turn the cache monitoring function on or off without having to change and refresh the CACHOPTS record. For more information about the monitoring function, see the description on the CACHOPTS record fields provided earlier in this chapter.

HELP

Provides online descriptions of the other cache facility console operator options.

Technical Information

Following is a list of technical issues that might affect how you decide to use the cache facility:

- Product version dependencies
- eTrust CA-ACF2 startup information
- Storage requirements
- Cache synchronization information
- Performance statistics and RMF reports

These issues are described in the *Systems Programmer Guide*. Also, you should consult with your systems programmer.

Chapter 13

Using the Command Propagation Facility

If your organization runs eTrust CA-ACF2 on multiple CPUs that are networked and more than one set of eTrust CA-ACF2 databases are associated with the network, or your eTrust CA-ACF2 network includes CA-Common Services nodes or eTrust CA-Top Secret nodes, you might want to use the Command Propagation Facility (CPF).

CPF lets a security administrator display information and fully maintain the Logonid database and structured and unstructured records in the Infostorage database on multiple networked nodes from a single node. Records in the Rules database and compiled records in the Infostorage database can be maintained using the RECKEY command. Extended Command Propagation adds the ability to propagate all ACALT API requests.

CPF also provides password synchronization across a defined network. If a user changes their password during system entry validation, CPF automatically updates the password on all defined nodes. Extended Password Synchronization adds the ability to propagate all administrative password changes and suspensions without requiring Command Propagation to be active.

CPF can streamline eTrust CA-ACF2 administration and let your site centralize control of the eTrust CA-ACF2 databases. CPF also allows the synchronization of passwords and user suspensions between eTrust CA-ACF2, eTrust CA-Top Secret, and CA-Common Services nodes. This chapter describes how to use CPF.

What Is CPF?

CPF consists of two main functions: Command Propagation and Password Synchronization. It is possible to activate Command Propagation, Password Synchronization, or both.

Command Propagation

eTrust CA-ACF2 CPF Command Propagation propagates INSERT, CHANGE, DELETE, and LIST commands for logonid records and non-compiled Infostorage records to the requested eTrust CA-ACF2 remote nodes. eTrust CA-ACF2 CPF also propagates the RECKEY command for modifying compiled infostorage records, and access and resource rules, and the DELETE command for access and resource rules. In addition, if Extended CPF is active, ACALT requests are also propagated. eTrust CA-ACF2 CPF Command Propagation does not propagate SHOW, SYNCH, COMPILE, DECOMPILE, and STORE commands to remote nodes.

Note: If a command fails at the home node, it is not propagated to the remote nodes.

Password Synchronization

eTrust CA-ACF2 CPF Password Synchronization propagates system entry password changes and suspensions due to password violations to all specified eTrust CA-ACF2 and eTrust CA-Top Secret remote nodes. In addition, if Extended CPF is active, all administrative password changes, suspensions, and password expirations to existing logonids will be propagated to all specified eTrust CA-ACF2 and eTrust CA-Top Secret nodes.

Note: If a command or password change fails at the home node, it is not propagated to the remote nodes.

CAICCI and CPF

CPF uses CAICCI to communicate information between network nodes. The Assured Delivery feature of CAICCI is used to ensure that requests are propagated to the requested nodes even if the network link is not active at the time the request is made. The Assured Delivery feature also ensures that requests are received if the network link fails in the middle of a transmission. For example, if a request is sent to two nodes and one of the nodes is down, the request is sent and processed immediately on the active node and the request to the inactive node is written to the CAICCI LOGGER database. When the node becomes active, CAICCI sends the request to the node and processing occurs. All requests sent to the LOGGER database are processed in first in first out order so that they are processed in the order that they were issued on the sending node.

CCILGR is required on eTrust CA-ACF2 systems if CPF is active. It is also required on eTrust CA-Top Secret systems if they are going to send or receive CPF requests to and from an eTrust CA-ACF2 system.

Note: For more information about eTrust CA-ACF2 installation requirements, see the *Getting Started*. For details on how to install CAICCI and CAIENF, see the CA-Common Services Getting Started guide.

CONTROL(CPF) Records

eTrust CA-ACF2 provides CONTROL(CPF) structured Infostorage records that let you implement CPF at your node and all remote nodes. The CPF records are OPTIONS and NODEDEF. This section provides an overview of these records and describes how the OPTIONS and NODEDEF records work together to define the processing in a CPF network.

CPF Options (OPTIONS)

The OPTIONS record specifies the name of the current node, (called HOME), the number of days that unprocessed requests will stay on the CAICCI LOGGER database, and the default target nodes for processing of ACF commands and password synchronization requests. The OPTIONS record lets you specify whether synchronous or asynchronous processing is the default for the node. You can also specify whether this node accepts and processes commands sent from nodes not defined with NODEDEF records at this node.

You can specify two journal files on the OPTIONS record, one for requests being sent to remote nodes and one for requests being received from remote nodes. You can also turn the journal process on and off while CPF continues to run. An OPTIONS record must exist at each eTrust CA-ACF2 node for CPF to get started.

The CONTROL(CPF) OPTIONS record specifies the options in effect for using CPF at the node where the record resides. You must define one OPTIONS record on each node in the CPF network. The record format and field descriptions follow:

Record ID	Fields
OPTIONS	CMDWAIT | NOCMDWAIT COMMAND | NOCOMMAND DFTCMD(*node1,...,node100*) DFTPSW(*node1,...,node100*) EXTDCPF | NOEXTDCPF HOME(*currentnode*) JRNLQS | NOJRNLQS JRNLRECV(*datasetname*) JRNLSEND(*datasetname*) JOURNAL | NOJOURNAL LOGDAYS(30 | *dd*) PSWDSYNC | NOPSWDSYNC RCVUND | NORCVUND RSTNLST(BOTH | DFTCMD | DFTPSW | NONE) TRMNLST(BOTH | DFTCMD | DFTPSW | NONE)

Field Descriptions

CMDWAIT | NOCMDWAIT

Defines the default type of processing that will take place on this node. CMDWAIT specifies that commands will be processed on a synchronous basis, requiring the user to wait for commands to complete on ALL specified nodes before the local command completes. NOCMDWAIT specifies that processing will be asynchronous. The local user will receive control when the command has been written to the CAICCI LOGGER database, not when the command has processed and returned from the remote node. This parameter has no effect on password synchronization, which is always processed asynchronously. This parameter can be overridden by the CPFWAIT/NOCPFWAIT parameter on individual eTrust CA-ACF2 commands. NOCMDWAIT is the default.

COMMAND | NOCOMMAND

Indicates whether this node will send command propagation requests through the CPF network. You may run command propagation without password synchronization. NOCOMMAND is the default. When NOCOMMAND is set, ACF commands will not be sent to remote nodes and commands will not be accepted from remote nodes. However, all commands will processed on the local node when NOCOMMAND is set.

Changes to this field are not automatically incorporated when a REFRESH of the OPTIONS record occurs. The command propagation task status can be changed by stopping and restarting CPF only. See the Refreshing CPF Records section for more information.

DFTCMD(*node1,...,node100*)
Identifies the remote eTrust CA-ACF2 node to which CPF can propagate commands. eTrust CA-ACF2 commands will not automatically execute on the HOME node. The HOME node **must** be in this target list if you want to have commands execute on the HOME node while CPF is active. eTrust CA-Top Secret nodes should not be specified in the DFTCMD list. The nodes in this list can be masked using standard eTrust CA-ACF2 masking rules.

Specify the same node names used to define the remote node to CAICCI. A remote node is defined to CAICCI by the CAICCI NODE control option. The node names specified here must match the SYSID parameter of the CAICCI NODE control option. CAICCI control options are defined in the CAIENF parameter file. For more information about CAICCI control options, see the CA-Common Services *Administrator Guide*.

DFTPSW(*node1,...,node100*)
Identifies the remote eTrust CA-ACF2 or eTrust CA-Top Secret nodes to which CPF can propagate password changes. If Extended CPF is active, DFTPSW identifies the remote nodes to which CPF can propagate all password related CHANGE commands. Password synchronization takes place after the password has been changed on the HOME node so the HOME node does not need to be placed in the DFTPSW node list. The nodes in this list can be masked using standard eTrust CA-ACF2 masking rules.

Specify the same node names used to define the remote node to CAICCI. A remote node is defined to CAICCI by the CAICCI NODE control option. The node names specified here must match the SYSID parameter of the CAICCI NODE control option. CAICCI control options are defined in the CAIENF parameter file. For more information about CAICCI control options, see the CA-Common Services *Administrator Guide*.

EXTDCPF|NOEXTDCPF
Specifies whether this node will propagate the additional events that are included in Extended CPF processing. If EXTDCPF and COMMAND are specified, Extended Command Propagation is active. If EXTDCPF and PSWDSYNC are active, Extended Password Sync is active. NOEXTDCPF is the default.

When Extended Password Sync is active, all changes to the PASSWORD, PSWD-EXP, PSWD-VIO, and SUSPEND logonid fields will be propagated to all nodes in the DFTPSW node list or to the override node list, if specified. When Extended Command Propagation is active, ACALT API requests will be sent to the default target node list or to the override node list, if specified, in the ACALT parameter list.

JRNLQS | NOJRNLQS

Enables CPF journaling to begin at the top of the journal file. During the CPF journal initialization process, the entire CPF journal file is read to find the oldest record where CPF will begin journaling data. The JRNLQS option may optimize CPF processing time and minimize CPF initialization time depending on the size of the journal file. The default is NOJRNLQS.

HOME(*currentnode*)

Identifies the eight character name of the node on which this record resides. This field corresponds to the CAICCI SYSID control option defined in the CAIENF parameter file. If HOME is not specified, it will default to the current SYSID that eTrust CA-ACF2 is running under at that point. This field is not maskable. For more information about the CAICCI SYSID control option, see the CA-Common Services Administrator Guide (or CA90s Services CAICCI User Guide).

JRNLRECV(*datasetname*)

Defines the name of the journal file that contains all inbound requests from remote nodes and the responses to the requests. This data set must be predefined and cataloged. The data set is dynamically allocated by eTrust CA-ACF2. It should not be added to the eTrust CA-ACF2 started procedure, and should not be the same as the one defined in JRNLSEND. The journal files cannot be shared among different nodes.

JRNLSEND(*datasetname*)

Defines the name of the journal file that contains all outbound requests from this node and the responses to the requests. This data set must be predefined and cataloged. The data set is dynamically allocated by eTrust CA-ACF2. It should not be added to the eTrust CA-ACF2 started procedure, and should not be the same as the one defined in JRNLRECV. The journal files cannot be shared among different nodes.

JOURNAL | NOJOURNAL

Indicates that the journal file processing should start when CPF starts. If you do not want to start journal file processing when CPF starts, you can start the journal file process at any time using the following command:

```
F ACF2,CPF(JOURNAL)
```

Journal file processing can be terminated using the following command:

```
F ACF2,CPF(NOJOURNAL)
```

LOGDAYS(*30 | dd*)

Specifies the number of days an undelivered request resides on the CAICCI LOGGER database at this node. If CAICCI is unable to deliver a request to another node within the specified time frame, the request is deleted from the CAICCI LOGGER database and is not delivered. When this occurs, an asterisk appears next to the LID in the CPF journal file. The default and maximum value is 30 days. The minimum value is 1 day.

PSWDSYNC | NOPSWDSYNC

Specifies whether this node will send password changes through the CPF network during system entry validation. PSWDSYNC indicates that password changes at system entry time will be sent to every node listed in the DFTPSW node list. You can run password synchronization without running command propagation. If NOPWDYSYNC is set, inbound requests are not processed either. NOPSWDSYNC is the default.

Changes to this field are not automatically incorporated when a REFRESH of the OPTIONS record occurs. The password synchronization task status can be changed by stopping and restarting CPF only. See Refreshing CPF Records later in this chapter for more information.

RCVUND | NORCVUND

Indicates whether the local node will process requests issued from a remote node that has not been defined with a NODEDEF record. The default is NORCVUND. The local node will not process requests from undefined remote nodes. This parameter only applies to Command Propagation requests. Password Synchronization, Extended Password Synchronization, and eTrust CA-Top Secret requests are never accepted from undefined nodes.

RSTNLST(BOTH | DFTCMD | DFTPSW | NONE)

Specifies the node list to use for the propagation of password violation count reductions from the F ACF2,RESET(*logonid*) operator command.

- BOTH—Use the DFTCMD and DFTPSW node lists. Duplicate nodes are removed from the merged list. Both COMMAND and PSWDSYNC should be specified. If COMMAND and/or PSWDSYNC are not specified, the corresponding node list will not be used. For example if NOCOMMAND and PSWDSYNC are specified, only the DFTPSW node list will be propagated to. If NOCOMMAND and NOPSWDSYNC are specified, RESET commands will not be propagated.
- DFTCMD—Use the DFTCMD node list. COMMAND must also be specified. If NOCOMMAND is specified, the RESET command will not be propagated.
- DFTPSW—Use the DFTPSW node list. PSWDSYNC must also be specified. If NOPSWDSYNC is specified, the RESET command will not be propagated.
- NONE—Do not propagate changes. This is the default.

TRMNLST(BOTH|DFTCMD|DFTPSW|NONE)
Specifies the node list to use for the propagation of ACTRM suspend requests.

- BOTH— Use the DFTCMD and DFTPSW node lists. Duplicate nodes are removed from the merged list. Both COMMAND and PSWDSYNC should be specified. If COMMAND and/or PSWDSYNC are not specified, the corresponding node list will not be used. For example if NOCOMMAND and PSWDSYNC are specified, only the DFTPSW node list will be propagated to. If NOCOMMAND and NOPSWDSYNC are specified, ACTRM requests will not be propagated.
- DFTCMD—Use the DFTCMD node list. COMMAND must also be specified. If NOCOMMAND is specified, the ACTRM request will not be propagated.
- DFTPSW—Use the DFTPSW node list. PSWDSYNC must also be specified. If NOPSWDSYNC is specified, the ACTRM request will not be propagated.
- NONE—Do not propagate changes. This is the default.

CPF Node Definition (NODEDEF)

The NODEDEF record defines a remote node to CPF and specifies whether this node can receive or send ACF commands. You should define a NODEDEF record for each node that you want to communicate with at each node. You do not need a NODEDEF record for the HOME node. If a NODEDEF record exists for the HOME node, it is ignored. The NODEDEF records indicate whether the HOME node can receive and send commands to a particular node. If a NODEDEF does not exist for a node, the HOME node will not be able to send commands to that node. If a NODEDEF record does not exist for a node, the HOME node will process inbound commands from that node if RCVUND is set in the CONTROL(CPF) OPTIONS record. If NORCVUND is set, a NODEDEF record must exist to process commands from other nodes.

You must specify a qualifier to identify each NODEDEF record. The value you specify for the qualifier must equal the name used to identify the node to CAICCI. This will be the SYSID field of the NODE parameter specified in the CAICCI parameters. The NODEDEF record format and field descriptions follow:

Record ID	Fields
NODEDEF*qual*	OUTCMD | NOOUTCMD INCMD | NOINCMD DESC(*description*) GATEWAY | NOGATEWAY TSSNODE | NOTSSNODE UNINODE | NOUNINODE VM | NOVM VMINFO | NOVMINFO VMLIDS | NOVMLIDS VMRULE | NOVMRULE VMLACCES | NOVMLACCES VM1ADAY | NOVM1ADAY

Field Descriptions

NODEDEF.*qual*

Specifies a unique name for this NODEDEF record. More importantly, the qualifier after any period must be the same as the node name defined to CAICCI. The qualifier can be up to nine characters, and must immediately follow the characters NODEDEF. If you use a period (.) as part of the qualifier string for the record name, eTrust CA-ACF2 counts it as one of the nine characters.

OUTCMD | NOOUTCMD

Defines whether CPF requests can be sent to the node identified in the qualifier. OUTCMD specifies that commands and password changes can be sent to the node identified in the qualifier

For example, to enter commands at the New York node to change a logonid record in the Chicago database, OUTCMD must be specified in the NODEDEF record for Chicago that resides in the New York Infostorage database. In addition, INCMD must be specified in the NODEDEF record for the New York node that resides in the Chicago Infostorage database to process the command on the Chicago system.

NOOUTCMD specifies that no commands or password changes can be sent from this node to the other node identified in the qualifier. NOOUTCMD is the default.

INCMD|NOINCMD
Defines whether commands, DB, and password synchronization can be received from the node identified in the qualifier. INCMD specifies that CPF requests issued from the node identified in the qualifier can be received.

For example, to enter commands at the New York node to change a logonid record in the Chicago database, INCMD must be specified in the NODEDEF record for the New York node that resides in the Chicago Infostorage database. In addition, OUTCMD must be specified in the NODEDEF record for Chicago that resides in the New York Infostorage database.

NOINCMD specifies that no commands, DB, and password synchronization can be received on this node from the other node identified in the qualifier. NOINCMD is the default.

DESC(*description*)
Lets you enter up to 20 characters of data. We recommend that you use this field to describe the CPF processing or the full node name.

GATEWAY|NOGATEWAY
Identifies this node as a GATEWAY node. When CPF receives a request from a GATEWAY node, the request is executed, and then forwarded to all other defined nodes. When processing requests from non-GATEWAY nodes, the request is executed, and then forwarded to all GATEWAY nodes.

TSSNODE|NOTSSNODE
Identifies this node as an eTrust CA-Top Secret node. All eTrust CA-Top Secret nodes must be designated as TSSNODE nodes. eTrust CA-Top Secret nodes can be specified in the DFTPSW node list and are eligible to receive Password Synchronization requests and Extended Password Synchronization requests.

UNINODE|NOUNINODE
Identifies this node as a Unicenter node. All Unicenter nodes must be designated as UNINODE nodes. When CPF receives a command from a GATEWAY or UNINODE node, the request is executed then forwarded to all other defined nodes. When processing requests from non-GATEWAY nodes, the request is executed and then forwarded to all GATEWAY or UNINODE nodes.

The UNINODE indicator restricts CPF to sending commands entered while in LID or ACF mode. Also, commands that refer to logonids without the UNICNTR attribute set are not sent to nodes with the UNINODE attribute set. This indicator has no effect on password synchronization requests.

VM|NOVM
Identifies the node in this NODEDEF record as a VM operating system node. Specifying this keyword in the NODEDEF record enables the Database Synchronization Component of eTrust CA-ACF2 to communicate with the Database Synchronization Component of eTrust CA-ACF2 for VM.

VMLIDS|NOVMLIDS
Defines whether the Database Synchronization Component will propagate logonid changes to and accept logonid changes from the VM node defined by this NODEDEF record. VMLIDS specifies that logonid changes will be propagated to and accepted from the VM node defined by this NODEDEF record. NOVMLIDS specifies that logonid changes will not be propagated to or accepted from the VM node defined by this NODEDEF record.

VMRULE|NOVMRULE
Defines whether the Database Synchronization Component will propagate updated access rules to and accept updated access rules from the VM node defined by this NODEDEF record. VMRULE specifies that updated access rules will be propagated to and accepted from the VM node defined by this NODEDEF record. NOVMRULE specifies that updated access rules will not be propagated to or accepted from the VM node defined by this NODEDEF record.

VMINFO|NOVMINFO
Defines whether the Database Synchronization Component will propagate updated infostorage records to and accept updated infostorage records from the VM node defined by this NODEDEF record.

VMINFO specifies that updated infostorage records will be propagated to and accepted from the VM node defined by this NODEDEF record. NOVMINFO specifies that updated infostorage records will not be propagated to or accepted from the VM node defined by this NODEDEF record.

Note: The VMLID/NOVMLID, VMRULE/NOVMRULE, and VMINFO/NOVMINFO keywords work with the INCMD/NOINCMD keywords. The Database Synchronization Component uses the command processing thread of CPF. INCMD/NOINCMD must be first set correctly to use these options.

VMLACCES|NOVMLACCES
Specifies whether updates to logonid records due to logons are synchronized. If you specify NOVMLACCES, logonid updates are not synchronized unless the password is changed during logon. If the password is changed, the update is synchronized according to the VMLIDS|NOVMLIDS setting.

VM1ADAY|NOVM1ADAY
Specifies whether to limit updates to logonid records under the VMLACCES setting to just once a day. If NOVMLACCES is specified, the VM1ADAY setting is ignored.

Refreshing CPF Records

To refresh CPF records, use the following eTrust CA-ACF2 modify commands.

OPTIONS Records

To refresh only the OPTIONS record:

```
F ACF2,REFRESH(OPTIONS),TYPE(CPF)
```

Note: Although REFRESH of the CPF OPTIONS record activates options on the currently running system, it does not automatically start or stop the command propagation task (COMMAND field) or the password synchronization task (PSWDSYNC field). You can change the task status by stopping and restarting CPF using the following commands:

```
F ACF2,CPF(STOP)
F ACF2,CPF(START)
```

You can insert several CPF OPTIONS records, each with a different SYSID. This allows you to easily start CPF with different configurations by using the following command:

```
F ACF2,CPF(START) SYSID(new1)
```

NODEDEF Records

To refresh only the NODEDEF records:

```
F ACF2,REFRESH(NODEDEF),TYPE(CPF)
```

All CPF Records

To refresh all CPF records:

```
F ACF2,REFRESH(ALL),TYPE(CPF)
```

Implementing CPF

This section presents the steps you should follow to implement the CPF subsystem for your networked nodes. It refers you to earlier sections of this chapter, to other chapters in this guide, and to other guides in the documentation set for more details.

1. Determine how you want to administer the eTrust CA-ACF2 databases using CPF. Some considerations are:
   - How many sites must be in the CPF network?
   - From which nodes can database updates be made and for which other nodes do those updates apply?
   - Are password changes allowed across any remote node?
   - Is there a central node where security administrators can update or display any other databases in the network?
   - Can other remote nodes update or display the databases of other remote nodes?
2. Ensure that CPF is active at all nodes that you want to place in the CPF network. Ensure that CAIENF and CAICCI were installed at all those sites. See the Getting Started guide for details on how to implement CAICCI and to the CA-Common Services Getting Started (or CA90s Services Installation Guide) for details on how to install CAIENF and CAICCI.
3. Create CPF records on each node in the network. See the CPF Node Definition (NODEDEF) and CPF Options (OPTIONS) sections earlier in this chapter for descriptions of the various fields that you must specify in the following records.
   - Create an OPTIONS record on each CA-ACF2 node in the network.
   - Create NODEDEF records for each node in the network at each node.

   The following commands are an example of the records that would be required to set up CPF on node CPF1. Similar records would be needed on all nodes involved.

A CPF OPTIONS record would be required on each node. The HOME field is the node name of this node as defined in CAICCI. For node CHI, the following OPTIONS record will allow all commands and passwords to be propagated:

```
set control(cpf) sysid(cpf1)

insert options command pswdsync extdcpf home(chi) dftcmd(chi nyc atl)
dftpsw(nyc la den) journal jrnlrecv(cpf1.journal.receive)
jrnlsend(cpf1.journal.send) rstnlst(both)

CPF1 / OPTIONS LAST CHANGED BY SECAD01 ON 09/29/03-12:23
                      NOCMDWAIT COMMAND DFTCMD(CHI NYC ATL)
                      DFTPSW(NYC LA DEN) EXTDCPF HOME(CHI) JOURNAL
                      JRNLRECV(CPF1.JOURNAL.RECEIVE)
                      JRNLSEND(CPF1.JOURNAL.SEND) LOGDAYS(30) PSWDSYNC
                      NORCVUND RSTNLST(BOTH) TRMNLST(NONE)
```

A CPF NODEDEF record would be required for each node that you wish to propagate to (OUTCMD) or receive requests from (INCMD).

```
insert nodedef.nyc incmd outcmd desc(new york acf2)

CPF1 / NODEDEF.NYC LAST CHANGED BY SECAD01 ON 09/29/03-12:28
                      DESC(NEW YORK ACF2) INCMD OUTCMD

insert nodedef.la incmd outcmd desc(los angeles acf2)

CPF1 / NODEDEF.LA LAST CHANGED BY SECAD01 ON 09/29/03-12:29
                      DESC(LOS ANGELES ACF2) INCMD OUTCMD

insert nodedef.atl incmd outcmd desc(atlanta acf2)

CPF1 / NODEDEF.ATL LAST CHANGED BY SECAD01 ON 09/29/03-12:30
                      DESC(ATLANTA ACF2) INCMD OUTCMD

insert nodedef.den incmd outcmd tssnode desc(denver top secret)

CPF1 / NODEDEF.DEN LAST CHANGED BY SECAD01 ON 09/29/03-12:31
                      DESC(DENVER TOP SECRET) INCMD OUTCMD TSSNODE
```

4. Create CPF journal files for each node that will receive CPF commands. See the "Command Propagation Facility (CPF)" chapter in the Implementation and Planning Guide for details.
5. Start CPF at each site. If you are changing CPF records after starting CPF, issue REFRESH commands to activate any CPF records you create. See the Refreshing CPF Records section earlier in this chapter.

6. The SHOW CPF command displays the CPF settings that are currently in effect.

```
show cpf

-- COMMAND PROPAGATION FACILITY --

CURRENT STATUS: ACTIVE

CURRENT SYSID: CPF1        JOURNAL: YES
CURRENT HOME NODE: CHI     LOGDAYS: 30
PASSWORD SYNC:YES          COMMAND: YES
EXTENDED CPF: YES          CMDWAIT: NO
UNDEFINED NODES:NO         JRNL QUICK START: NO

DFTCMD: CHI NYC ATL
DFTPSW: NYC LA DEN
JRNLRECV: CPF1.JOURNAL.RECEIVE
JRNLSEND: CPF1.JOURNAL.SEND
RSTNLST: BOTH
TRMNLST: NONE

-- NODE DEFINITIONS --
NODE   RECEIVE SEND GTWY UNI  TSS   VM   VM   VM   VM   VM    VM
NAME    FROM    TO  NODE NODE NODE NODE LIDS RULE INFO LACCES 1ADAY
======  ====   ==== ==== ==== ==== ==== ==== ==== ==== ====== =====
ATL     YES    YES   NO   NO   NO   NO   NO   NO   NO   NO     NO
CHI     YES    YES   NO   NO   NO   NO   NO   NO   NO   NO     NO
DEN     YES    YES   NO   NO   YES  NO   NO   NO   NO   NO     NO
LA      YES    YES   NO   NO   NO   NO   NO   NO   NO   NO     NO
NYC     YES    YES   NO   NO   NO   NO   NO   NO   NO   NO     NO
```

eTrust CA-ACF2 CPF Processing

CPF and SYSIDs

eTrust CA-ACF2 processes all CPF commands using the current SYSID value at the node from where the command is issued. That is, the command that is issued by CPF at the remote node inherits the MODE (including the SYSID) of the command issued at the home node.

For example, suppose you have two nodes, NYC and CHI. Both of these nodes use two SYSIDs, CPU1, and CPU2. If you issue the following subcommand:

```
CHANGE OPTS CENTRAL TARGET(CHI)
```

If the current SYSID on NYC is CPU1, CPF changes the OPTS record for the CPU1 node on CHI, regardless of which SYSID is active on CHI.

Command Propagation

Command Propagation lets a security administrator display information and fully maintain the Logonid database and structured and unstructured records in the Infostorage database on multiple networked eTrust CA-ACF2 nodes from a single node. eTrust CA-Top Secret nodes are not eligible for Command Propagation.

Local and Remote ACF Subcommands

Local subcommands are commands that are only processed at the node where they are issued. Remote subcommands are those that are eligible to be propagated to remote nodes.

Local Subcommands

The following subcommands are always local:

- HELP
- SYNCH
- SET
- SHOW

This means that you cannot view help files for eTrust CA-ACF2 on another node.

Likewise, you cannot synchronize the Logonid database with SYS1.BRODCAST at a remote node. The SET subcommand might appear to be remote, but it really is not. For example, suppose you issue the following command:

```
SET TARGET(CPU1)
```

This subcommand tells eTrust CA-ACF2 at your site that you want subsequent commands to be processed on system CPU1. It conveys no message to eTrust CA-ACF2 running at CPU1. The CPU1 system does not receive any commands until you issue an INSERT, CHANGE, LIST, or DELETE subcommand.

Remote Subcommands

There are no purely remote subcommands. A subcommand becomes a remote subcommand when:

- You precede it with a SET TARGET subcommand specifying a remote node as a target
- You include a TARGET parameter as part of the subcommand specifying a remote node as a target
- You specify several nodes as default targets in the OPTIONS record at the current node.

Local or Remote Subcommands

These subcommands can be local, remote, or both depending on the values specified in the TARGET operand or the DFTCMD field of the CPF OPTIONS record:

- INSERT
- CHANGE
- LIST
- DELETE

Default Node Lists

The OPTIONS record, described previously, contains several parameters, DFTCMD and DFTPSW, which establish the default target lists for ACF commands and Password changes. The DFTCMD node list is used when propagating commands and the DFTPSW node list is used when synchronizing system entry password changes. If Extended Password Sync is active (EXTDCPF and PSWDSYNC specified in the CPF OPTIONS record), the DFTPSW node list is used when synchronizing administrative password changes.

The DFTCMD and DFTPSW default node lists may contain masked entries. There is a limit of 100 entries in each default node list after the masking has been resolved. There is also a limit of 100 nodes when the default node lists are combined. For example, each node list can contain 100 entries if the same entries are in both lists.

Using the TARGET Parameter

This section describes how to use the TARGET parameter. The TARGET parameter of the SET, INSERT, CHANGE, LIST, RECKEY, and DELETE subcommands provides the ability to override the default node lists specified in the CPF OPTIONS record. The TARGET parameter is valid if command propagation or Extended Password Synchronization is active. If Command Propagation is active, the entire command is propagated to the eTrust CA-ACF2 nodes specified in the TARGET parameter. If Extended Password Synchronization is active, any password related changes are propagated to the nodes specified in the TARGET parameter. If both Command Propagation and Extended Password Synchronization are active, the entire command is propagated to the eTrust CA-ACF2 nodes specified, and any password related changes are propagated to the eTrust CA-Top Secret nodes specified.

If the TARGET parameter is specified, the home node must be specified or the command will not be executed on the home node. If the home node is specified, the entire command is executed on the home node and if the command fails, no propagation to remote nodes occurs.

Scope of the TARGET Parameter

The subcommand that the TARGET parameter is used in determines the scope of the TARGET parameter. When used with the SET subcommand, the value you specify for the TARGET parameter replaces the default node list and applies to all subsequent subcommands.

When used with the CHANGE, INSERT, LIST, RECKEY or DELETE subcommands, the value you specify for the TARGET parameter applies to that subcommand only. Any subsequent ACF subcommands that you enter revert to the default value. **Note:** SET TARGET does **not** apply to the LIST command.

eTrust CA-ACF2 selects the node(s) at which to process a command based on the following hierarchy:

- **Current command**—The TARGET value specified as part of an INSERT, CHANGE, LIST, DELETE, or RECKEY subcommand. This target is active for this subcommand only. It reverts to the previous SET subcommand, if one was issued, or to the default target specified in the OPTIONS record after this subcommand is executed.
- **Current setting**—The TARGET value specified in the most recent SET subcommand. eTrust CA-ACF2 considers this the current default setting until you issue another SET subcommand.
- **Default**—The value specified as the default target in the OPTIONS record. ACF subcommands are processed at these nodes unless you specify the TARGET parameter.

TARGET Parameter Syntax

To set the target, you can specify the TARGET parameter with the SET, INSERT, CHANGE, DELETE, RECKEY and LIST subcommands. The syntax for the TARGET parameter is the same for all of the subcommands:

```
TARGET(null|=|?| nodemask1,...,nodemask100)
```

The values are described in the following:

null TARGET()
Processes commands only on the HOME node specified in the CPF OPTIONS record.

= TARGET(=)
Processes commands only on the HOME node specified in the CPF OPTIONS record.

TARGET(?)
Processes commands to the default nodes listed in the CPF OPTIONS record. This can be used to revert to the default node lists specified in the CPF OPTIONS record after issuing a SET TARGET subcommand.

nodemask1,...,nodemask100
TARGET(nodemask1,...,nodemask100) processes at these specific nodes. These node masks can be individual nodes or they can be masked with asterisks. Up to 100 nodes are allowed in this list. eTrust CA-Top Secret nodes are allowed in this list, however, they will only be used if Extended Password Sync is active.

You MUST have the CMD-PROP privilege in your logonid to use the TARGET parameter to override the DFTCMD and DFTPSW parameters of the CPF OPTIONS record. SECURITY or ACCOUNT privileges are not sufficient to override the default target list.

ACF Subcommands

You can specify the TARGET parameter with any of the following ACF subcommands:

- CHANGE
- DELETE
- INSERT
- LIST
- RECKEY
- SET

The TARGET parameter can be placed anywhere in the command line after the subcommand.

Note: The TARGET parameter is only valid when command propagation or Extended Password Synchronization is active. If Command Propagation or Extended Password Synchronization are not active, any command containing the TARGET parameter will be rejected.

Examples

The following are examples of each subcommand:

set target(CPU1)
Replaces the current default node list with node CPU1. Future commands will be propagated to the single node CPU1and will not be executed on the home node. This target list is in effect until another SET TARGET command is entered. This target value can be temporarily overridden by placing TARGET on individual commands.

set target()
Replaces the current default node list with the home node only and subsequent commands will only execute on the home node.

set target(?)
Replaces the current default node list with the nodes specified in the CPF OPTIONS record. This command resets the target node list to the values that were in effect before any SET TARGET commands.

insert using(user1) user2 name(user1) password(user1) target(cpu2)
Inserts a logonid on the database in use on CPU2. This target is used only for this command.

change user1 target(cpu3) security account
Changes the USER1 logonid on the CPU3 database. This target is used only for this command.

list like(aaa-) target(cpu4)
Lists all logonids at the CPU4 node that begin with AAA. This target is used only for this command.

delete user1 target(*******)
Deletes USER1 at all defined nodes. This target is used only for this command.

LIST Command and CPF

The eTrust CA-ACF2 CPF is used as a means to synchronize logonids and infostorage commands throughout the network. Imagine a network with five nodes. A user needs to look at a logonid and gets five identical versions returned. Since these records are identical in most cases, the LIST command has a special property. It is not propagated to remote nodes unless the TARGET keyword is explicitly added to the command. This reduces the amount of redundant information passed through the CPF network.

Abbreviating TARGET

Standard eTrust CA-ACF2 parsing mechanisms let keywords be abbreviated to the minimum number of letters required to differentiate two keywords. Since the TARGET parameter is allowed on multiple subcommands and can be placed anywhere in the command line, the smallest abbreviation of TARGET might differ depending on the subcommand and record type. For example, the SET subcommand has three parameters that begin with T, TERSE, TRIVIA, and TARGET. In this case, TARGET can be abbreviated as TA. Logonids have two fields that begin with TA: TAPE-LBL and TAPE-BLP. Because of these fields, when you insert or change logonids, the smallest abbreviation of TARGET is TAR. When listing or deleting logonids, the smallest abbreviation is T because the logonid fields cannot be used in the command.

There are similar examples in infostorage records. To be safe in all cases and to ease confusion, use TAR as the minimum abbreviation for the TARGET parameter.

Subcommands Affecting Multiple Records

eTrust CA-ACF2 subcommands can contain one or more parameters that cause multiple records to be modified by a single subcommand. These parameters are LIKE(logonid mask), UID(uidmask), and IF(filterlist). When a subcommand is issued that contains any of these parameters, matching records are selected and the requested modifications are performed. If Command Propagation or Extended Password Synchronization are active, these subcommands may be propagated to remote nodes for processing. It is important to remember that the request is propagated, not the actual changes that took place on the home node.

For example, the following subcommand is issued on CPU1:

```
Change like(empl*) suspend
```

If the home node logonid database contains 5 logonids that match empl* (empl1 – empl5), all 5 are changed. When this subcommand is propagated, it is possible that the logonid mask will not match the same logonids as the home node. If CPU2 has empl1- empl9 in the database, all nine logonids will be changed.

If the following subcommand is issued on CPU1:

```
Change like(-) if(security) nosuspend
```

The logonids changed on CPU1 may be completely different than the logonids changed on CPU2. When a subcommand contains masking parameters, the results may be inconsistent on all the nodes that are propagated to. In addition, logonid masking, UID strings and IF filters are not supported in eTrust CA-Top Secret. CHANGE commands that contain the LIKE, IF or UID parameters are not propagated to eTrust CA-Top Secret nodes when Extended Password Sync is active.

For these reasons, it is recommended that administrators take extra care when using the LIKE, UID and IF parameters when Command Propagation or Extended Password Synchronization are active.

CPF to eTrust CA-Top Secret Nodes

eTrust CA-ACF2 supports the propagation of system entry password changes to and from eTrust CA-Top Secret nodes when Password Synchronization is active. It is only necessary to add a NODEDEF with TSSNODE specified for the eTrust CA-Top Secret node and add the node name to the DFTPSW field in the CPF OPTIONS record.

Administrative password changes and other password related field changes via the ACF CHANGE subcommand or ACALT change requests are also eligible for propagation to eTrust CA-Top Secret nodes when Extended Password Sync is active. A valid NODEDEF with TSSNODE specified may be added to the DFTPSW node list or may be specified in the TARGET parameter of an eTrust CA-ACF2 subcommand or the ACATARG field of the ACALT parameter list. Only CHANGE subcommands are inspected for password related fields (PASSWORD, PSWD-VIO, PSWD-EXP, and SUSPEND). If any masking parameters are found, the request is not propagated to eTrust CA-Top Secret nodes.

The RESET console command can also be propagated to remote eTrust CA-Top Secret nodes. When the RESET console command is executed on eTrust CA-ACF2 nodes, the PSWD-VIO count for the specified user is decremented by one. eTrust CA-Top Secret does not maintain a PSWD-VIO field, therefore when the RESET command is executed on an eTrust CA-Top Secret system, the password violation suspend attribute is removed for the specified user. It is possible that the execution of the RESET command can cause eTrust CA-ACF2 and eTrust CA-Top Secret databases to be out of sync. It is recommended that sites use care when propagating RESET console commands to eTrust CA-Top Secret nodes.

eTrust CA-CA-ACF2 to eTrust CA-Top Secret Password Sync processing is not available in all releases of both products. Contact CA Technical Support if you require more information about the levels of eTrust CA-ACF2 or eTrust CA-Top Secret required.

System Entry Password Changes

When a user logs on to an eTrust CA-ACF2-protected system, they can change their password at the nodes specified in the CPF OPTIONS DFTPSW field. The DFTPSW field can contain eTrust CA-ACF2 and eTrust Top Secret nodes and up to 100 nodes can be specified. PSWDSYNC must be specified in the CPF OPTIONS record on both the sending and receiving nodes. The default is NOPSWDSYNC.

A password change is propagated after it has completed successfully on the home node.

Consider the following logon example:

```
logon user01

ACF82004 ACF2, ENTER PASSWORD
    asssword/newpassword
ACF82020 ACF2, REENTER NEW PASSWORD FOR VERIFICATION -
newpassword
ACF82000 ACF2, LOGON IN PROGRESS
ACF01129 PASSWORD SUCCESSFULLY ALTERED
```

This is the normal procedure for changing a password at logon. If CPF is active at each of the nodes where the user is defined and the two nodes can communicate (the INCMD and OUTCMD fields are specified to enable communication), the password change takes effect at each node. The user receives no indication that CPF processing was involved. The changes take effect automatically.

Extended Password Synchronization Processing

Extended Password Synchronization is an option that allows sites to ensure that all password changes and other password related changes to a logonid are propagated to all nodes specified in the DFTPSW field of the CPF OPTIONS record. Extended Password Synchronization requires that Password Synchronization is active. Command Propagation can be active or inactive. All nodes that are to send Extended Password Synchronization requests must have EXTDCPF and PASSWORD specified in the CPF OPTIONS record. All nodes that are to receive Extended Password Synchronization requests must at least have PASSWORD specified in the CPF OPTIONS record.

When Extended Password Synchronization is active, any changes to password related logonid fields are propagated to all DFTPSW nodes that are not already receiving the change via Command Propagation. All administrative changes are inspected for modifications to the following fields: PASSWORD, PSWD-VIO, PSWD-EXP, and SUSPEND. Only these fields are eligible for propagation via Extended Password Synchronization and only changes to existing logonids are inspected.

For example, node CHI contains a CPF OPTIONS record that specifies PSWDSYNC, NOCOMMAND, EXTDCPF, DFTCMD(CHI NYC) and DFTPSW(NYC LAS). The following command is issued by a security administrator:

```
change user01 password(newpw) name(newname)
```

The password change request will be propagated to nodes NYC and LAS but the name change will only occur on the home node, CHI. Since Command Propagation is not active only password related changes are propagated.

If the OPTIONS record is changed to specify COMMAND, and the same command is issued, the entire command would be executed on the home node and node NYC. Node LAS would receive only the password change request.

Extended Command Propagation Processing

Extended Command Propagation is an option that allows sites to propagate ACALT API requests to all nodes specified in the DFTCMD field of the CPF OPTIONS record or to the override node list, if specified. Extended Command Propagation requires that Command Propagation is active, but Password Sync can be active or inactive. All nodes that are to send Extended Command Propagation requests must have EXTDCPF and COMMAND specified in the CPF OPTIONS record. All nodes that are to receive Extended Command Propagation request must at least have COMMAND specified in the CPF OPTIONS record.

The ACALT parameter list is used to insert, delete, modify, and return logonid information via the ACFSVC TYPE=A SVC call. ACALT request propagation processing is similar to ACF command propagation. If Extended Command Propagation is active, the home node must be specified in the DFTCMD node list or the override node list for the request to be processed on the home node.

To override the default node lists the ACATARG field must contain the address of the target override list. If ACATARG contains the address of a valid node list, the ACALT request is propagated to all eTrust CA-ACF2 nodes in the list.

Note: If Extended Password sync is active but Command Propagation is not active, nodes specified in the override node list pointed to by the ACATARG field will receive password related changes and the entire ACALT request will execute on the home node.

If Extended Command Propagation is active, the ACALT request is propagated to the DFTCMD nodes or any eTrust CA-ACF2 nodes specified in the target override node list. If Extended Password Sync is also active, any password related changes are propagated to DFTPSW nodes or any eTrust CA-Top Secret nodes specified in the target override node list.

If Extended Command propagation and Extended Password Synchronization are active, all ACALT requests are eligible to be propagated. The ACANCPF flag in the ACALT parameter list can be set to indicate that this individual ACALT request should not be propagated. All ACALT requests are sent asynchronously even if CMDWAIT is specified. A successful or unsuccessful message is returned from remote nodes.

For additional information about the ACALT parameter list, see the *System Programmers* guide.

RESET and ACTRM Propagation

The eTrust CA-ACF2 RESET console command and ACTRM request can also be propagated to other nodes. RESET and ACTRM can be propagated via Command Propagation or Password Synchronization or both. The CPF OPTIONS record fields RSTNLST and TRMNLST are used to specify which default node lists should be used when propagating RESET and ACTRM. Valid values for these fields are DFTCMD, DFTPSW, BOTH or NONE. When BOTH is specified, the DFTPSW and DFTCMD lists are merged and duplicate entries are removed.

If you wish to propagate these requests, the node list that you specify must correspond to the active task(s). For example, if you wish to propagate RESET requests and only Password Sync is active, you should specify RSTNLST(DFTPSW) in the CPF OPTIONS record. If you specify RSTNLST(DFTCMD), the RESET request will not be propagated, and if you specify RSTNLST(BOTH) only the DFTPSW nodes will receive the request.

When the RESET console command is executed on the home node or remote ACF2 nodes, the PSWD-VIO count for the specified user is decremented by one. Top Secret does not maintain a PSWD-VIO field, therefore when the RESET command is propagated to remote Top Secret nodes, the password violation suspend attribute is removed for the specified user. Because of the differences in this processing, it is possible that the execution of the RESET command can cause the databases to be out of sync. It is recommended that sites use care when propagating RESET console commands to Top Secret nodes.

GATEWAY and UNINODE Processing

When a NODEDEF record has the GATEWAY attribute, GATEWAY processing rules apply. When processing a request from a GATEWAY node, the request is passed to all other defined nodes. When processing from a non-GATEWAY node, the request is passed to all defined GATEWAY nodes.

The GATEWAY attribute can be used to reduce the amount of records required to define a CPF network. In a six-node CPF network, you must define five NODEDEF records on each node (and six journal files if you use journal files).

However, using a GATEWAY node, you would reduce the number of NODEDEF records required to two on four nodes, and three on two nodes.

CA-Common Services nodes usually have only a fraction of the mainframe users defined. CA-Common Services nodes also only process password changes and user suspension requests. CPF sends all requests to all defined nodes regardless of whether the node can process the request. The UNINODE field of the

NODEDEF record provides the ability to distinguish between GATEWAY nodes and CA-Common Services GATEWAY nodes.

When a NODEDEF record has the GATEWAY attribute or the UNINODE attribute or both attributes, the normal GATEWAY processing rules apply. When processing a request from a GATEWAY or UNINODE node, the request is passed to all other nodes. When processing from a non-GATEWAY node, the request is passed to all GATEWAY and UNINODE nodes. To further restrict the traffic to CA-Common Services nodes, requests pertaining to loginids are only sent if the logonid being processed has the UNICNTR attribute set. Any request that is not sent due to these rules generates a journal record.

CPF Exit Processing

A CPF exit provides the ability to restrict the requests that are sent through CPF. This exit, CPFEXIT, is provided with the ACF mode, the command name, the record key, and a list of nodes where the request is to be sent. The exit has the ability to indicate that individual nodes in the list that is passed to it should not receive the request. The exit cannot override the target list by adding nodes. See the *Systems Programmer Guide* for details on the CPFEXIT.

CCI Generic Resources

CCI Generic Resources allow you to declare a group of CCI systems that are to be treated as a single logical entity. The use of CCI Generic Resources should be considered when sharing eTrust CA-ACF2 databases in a CPF environment. CCI Generic Resources allows you to assign a logical name to a group of CCI nodes (SYSID). When CPF sends an update to a CCI Generic Resource, the update is sent to one and only one of the nodes defined to the logical group. This feature avoids the problem of multiple updates to an eTrust CA-ACF2 database and ensures that updates are performed in a timely manner by sending updates to an active system in the Generic Resource logical group.

Each system in the logical group must include a CCI SYSPLEX definition that contains the same value, for example, SYSPLEX (CCIONE). Additionally, NODEDEF records are required for all systems within the logical group. A CPF NODEDEF record must also be defined for the generic resource and that node can be used as the target of a CPF update.

The following command is issued from a system that is not in CCIONE.

```
CHANGE userid TSO GROUP(newgroup) TARGET(CCIONE)
```

Note: The home node must always be defined as a CCI SYSID. CPF should never send to a generic resource that includes the home node.

CPF Journaling

CPF journal processing can be used to record all requests that are processed by CPF. You can specify two journal files on the CPF OPTIONS record, JRNLSEND and JRNLRECV. The file specified for JRNLSEND records all requests being sent to remote nodes and the responses received for these requests. The file specified for JRNLRECV records all requests received from remote nodes and the responses sent back to these remote nodes.

Journaling is activated by specifying JOURNAL in the CPF OPTIONS record. You can also turn the journal process on and off while CPF is active by issuing the following eTrust CA-ACF2 modify commands:

```
F ACF2,CPF(JOURNAL)
```

Or

```
F ACF2,CPF(NOJOURNAL)
```

By default, journaling is disabled (NOJOURNAL) and is not required by CPF. Sites may choose to enable journaling to create a record of CPF activity or to assist in configuring CPF.

Journal files must be created using the INITJNL job in SAMPJCL. The size of the journal files can be specified in the INITJNL job. Journaling continues at the top of the file when the file is full.

The following is an example of the entries that will be found in a journal file after a single request is processed:

```
2004162 164036 SYS1     ACFUSR1   000000002  MODE:  LID       TYPE:      SYSID/DIVISION:
2004162 164036 SYS1     ACFUSR1              change acfusr2 password(        )
2004162 164036 SYS2     ACFUSR1   000000002  eTrust CA-ACF2/CPF COMMAND PROCESSED SUCCESSFULLY
```

The following example indicates an error condition:

```
2004162 164036 SYS1     ACFUSR1   000000003  MODE:  LID       TYPE:      SYSID/DIVISION:
2004162 164036 SYS2     ACFUSR1              change acfusr2 password (        )
2004162 164036 SYS2     ACFUSR1   000000003  MODE:  LID       TYPE:      SYSID/DIVISION:
2004162 164036 SYS2     ACFUSR1              change acfusr2 password(         )
```

The character after the userid indicates that an error occurred while processing this request:

#
: A CCI error occurred and request was returned to originating node.

$
: Request sent to originating node by CCI because of a configuration error.

*
: Request purged by CCI LOGGER. Destination was not available to receive request.

Using the ISPF Panels

To process CPF records using the ISPF panels, select option 11 CPF from the eTrust CA-ACF2 Security ISPF Option Selection Menu. The eTrust CA-ACF2 Security Command Propagation Facility Services panel is displayed:

```
-------- eTrust CA-ACF2 Security COMMAND PROPAGATION FACILITY SERVICES ---------
OPTION ===>

  1 INSERT  - INITIALLY DEFINE CPF RECORDS

  2 CHANGE  - CHANGE EXISTING CPF RECORDS

  3 LIST    - DISPLAY CPF RECORDS AND PARAMETERS

  4 DELETE  - DELETE AN ENTIRE CPF RECORD

  5 MODE    - SHOW DEFAULT CPF TARGET NODES IN EFFECT

  6 FIELDS  - SHOW CPF RECORD FIELD NAMES

  7 TARGETS - SET CPF TARGET NODES, DEFAULTS IN USE
```

If you want to process an ACF command at a node other than the default node or the current target setting, you must select the TARGET option and specify the target nodes before you select INSERT, CHANGE, LIST, or DELETE.

Creating CPF Records

Select option 1 INSERT from the eTrust CA-ACF2 Security Command Propagation Facility Services panel to create a CPF record. The Insert A CPF Record panel is displayed:

```
---------------- eTrust CA-ACF2 Security INSERT A CPF RECORD -------------------
COMMAND ===>

  INSERT

  CHANGE TYPE ===> ADD          (ADD)

  SYSID       ===>              SYSTEM ID FOR CPF RECORD

  USING SYSID ===>              OPTIONAL SYSTEM ID FOR COPY OF EXISTING VALUES

  USING RECID ===>              OPTIONAL PROTOTYPE RECORD NAME

  RECID       ===>              (NODEDEF, OPTIONS)

  SPECIFICATION OF RECID WILL RESULT IN DISPLAY OF APPROPRIATE PANEL
  FOR OPTION SPECIFIED
```

Panel Field Descriptions

CHANGE TYPE
Specifying ADD for bit- or single-value fields has no effect. These changes apply only to multi-value fields.

- ADD
  adds the values specified to the default values.

SYSID
Specify the system ID for the CPF record that you want to create.

USING SYSID
Specify a system ID of another CPF record that you want to use to create a CPF record on the current system.

USING RECID
Specify the name of a record that you want to use as a model to create this record.

RECID
Specify the record type you want to insert: NODEDEF creates a NODEDEF record; OPTIONS creates an OPTIONS record. This field determines the panel that is displayed next.

Changing CPF Records

To change a CPF record, select option 2 CHANGE from the eTrust CA-ACF2 Security Command Propagation Facility Services panel. The Change A CPF Record panel is displayed:

```
----------------- eTrust CA-ACF2 Security CHANGE A CPF RECORD --------------
COMMAND ===>

  CHANGE

  CHANGE TYPE ===> ADD           (ADD, DEL, REP)

  SYSID       ===>               SYSTEM ID FOR CPF RECORD

  MASK SYSID  ===>               MASKED SYSTEM ID FOR CPF RECORD

  MASK RECID  ===>               RECID MASKED VALUE

  RECID       ===>              (NODEDEF, OPTIONS)

  SPECIFICATION OF RECID WILL RESULT IN DISPLAY OF APPROPRIATE PANEL
  FOR OPTION SPECIFIED
```

Panel Field Descriptions

CHANGE TYPE
Specify one of the following types of changes to the record:

- ADD
  adds the values specified to the default values.
- DEL
  deletes the values from the default values.
- REP
  replaces the default value with the value specified.

SYSID
Specify the system ID for the CPF record that you want to change.

MASK SYSID
Specify a system ID mask to change CPF records for a group of systems.

MASK RECID
Specify a record name mask to change a group of CPF records.

RECID
Specify the record type that you want to change: NODEDEF modifies a NODEDEF record; OPTIONS modifies an OPTIONS record. This field determines the panel that is displayed next.

Displaying CPF Records

To display a CPF record, select option 3 LIST from the eTrust CA-ACF2 Security Command Propagation Facility Services panel. The List A CPF Record panel is displayed:

```
------------------- eTrust CA-ACF2 Security LIST A CPF RECORD -----------------
COMMAND ===>
  LIST
  SYSID      ===>            SYSTEM ID FOR CPF RECORD
  MASK SYSID ===>            MASKED SYSTEM ID FOR CPF RECORD
  QUALIFIER  ===>            CPF RECORD QUALIFIER
  MASK RECID ===>            RECID MASKED VALUE
  RECID      ===>            (NODEDEF, OPTIONS)
```

Panel Field Descriptions

SYSID
Specify the system ID for the CPF record that you want to list. If you do not specify a system ID, this field defaults to the current SYSID of the system under which you are running.

MASK SYSID
Specify a system ID mask to list CPF records for a group of systems.

QUALIFIER
Specify the qualifier for the NODEDEF record that you want to view. There can be only one OPTIONS record for each SYSID.

MASK RECID
Specify a record name mask to list a group of CPF records.

RECID
Specify NODEDEF or OPTIONS.

Deleting CPF Records

To delete a CPF record, select option 4 DELETE from the eTrust CA-ACF2 Security Command Propagation Facility Services panel. The Delete A CPF Record panel is displayed:

```
----------------- eTrust CA-ACF2 Security DELETE A CPF RECORD ------------------
COMMAND ===>

  DELETE

  SYSID      ===>           SYSTEM ID FOR CPF RECORD

  MASK SYSID ===>           MASKED SYSTEM ID FOR CPF RECORD

  QUALIFIER  ===>           CPF RECORD QUALIFIER

  MASK RECID ===>           RECID MASKED VALUE

  RECID      ===>          (NODEDEF, OPTIONS)
```

Panel Field Descriptions

SYSID
Specify the system ID for the CPF record that you want to delete. If you do not specify a system ID, this field defaults to the current SYSID of the system under which you are running.

MASK SYSID
Specify a system ID mask to delete CPF records for a group of systems.

QUALIFIER
Specify the qualifier for the NODEDEF record that you want to delete. There can be only one OPTIONS record for each SYSID.

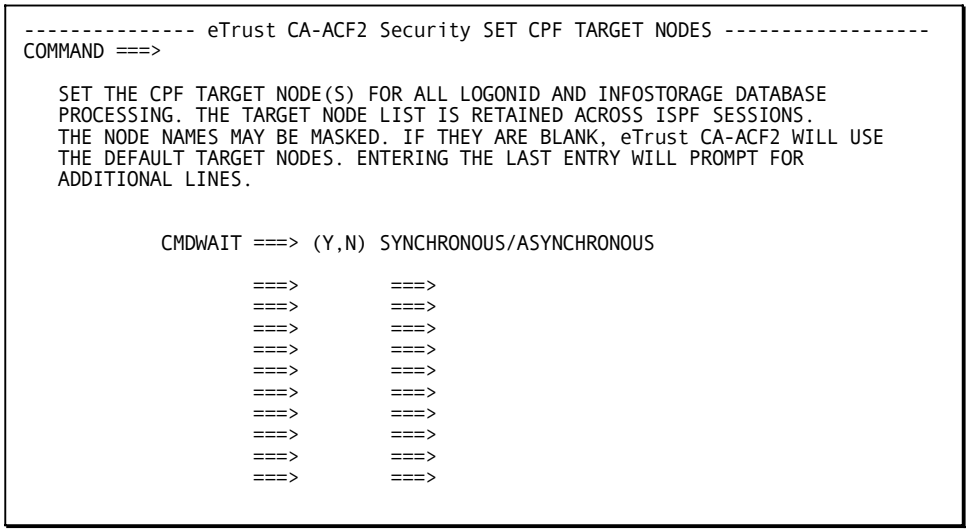
MASK RECID
Specify a record name mask to delete a group of CPF records.

QUALIFIER
Specify a one to nine-character qualifier name.

RECID
Specify NODEDEF or OPTIONS.

Setting Target Nodes

Select option 7 TARGETS from the eTrust CA-ACF2 Security Command Propagation Facility Services panel to set the target nodes where you want the commands to take effect. The Set CPF Target Nodes panel is displayed:



Enter up to 100 target node values. Characters of the nodename can be masked with an asterisk (*). The CPF address space must be active on each node and you must have defined the nodes in CPF NODEDEF and OPTIONS records for the commands to take effect.

Displaying CPF Options

Select option 5 MODE from the eTrust CA-ACF2 Command Propagation Facility Services panel to display the SHOW subcommands for CPF. See SHOW MODE and SHOW CPF Subcommands later in this chapter for more information.

Displaying Field Names for a CPF Record

Select option 6 FIELDS from the eTrust CA-ACF2 Command Propagation Facility Services panel to display the fields of a CPF record. The fields for the various CPF records are described earlier in this chapter. See the descriptions of the OPTIONS and NODEDEF records in particular.

Using the ACF Command

Since CPF records are another type of structured infostorage record, you can find details on the syntax of the ACF command as it applies to structured infostorage records in the chapter entitled, "Maintaining Structured Infostorage Records." Examples of CPF records and how to implement CPF are described earlier in this chapter.

Refreshing CPF Records

To refresh CPF records, use the following eTrust CA-ACF2 modify commands.

OPTIONS Records

To refresh only the OPTIONS record:

```
F ACF2,REFRESH(OPTIONS),TYPE(CPF)
```

Note: Although REFRESH of the CPF OPTIONS record activates options on the currently running system, it does not automatically start or stop the command propagation task (COMMAND field) or the password synchronization task (PSWDSYNC field). You can change the task status by stopping and restarting CPF using the following commands:

```
F ACF2,CPF(STOP)
F ACF2,CPF(START)
```

NODEDEF Records

To refresh only the NODEDEF records:

```
F ACF2,REFRESH(NODEDEF),TYPE(CPF)
```

All CPF Records

To refresh all CPF records:

```
F ACF2,REFRESH(ALL),TYPE(CPF)
```

SHOW MODE and SHOW CPF Subcommands

You can use the SHOW MODE subcommand to display a list of the current target nodes. If no SET TARGET commands have been issued, SHOW MODE will list the nodes in the DFTCMD, DFTPSW or a combined node list depending on what CPF functions are active. If only Command Propagation is active, SHOW MODE will display nodes in the DFTCMD node list. If only Extended Password Sync is active, the DFTPSW nodes are displayed. If both are active, a combined list is displayed. If a SET TARGET(…) subcommand has been issued, this override list is displayed.

```
set control(CPF)
CPF
show mode
MODE: CONTROL, TYPE: CPF, SYSID: CHI, TARGET: NYC,LAS,WAS
```

You can also use the SHOW CPF subcommand to display information about the OPTIONS record and the CPF network as defined in NODEDEF records. Here is the information provided by the SHOW CPF subcommand:

```
show cpf

-- COMMAND PROPAGATION FACILITY --

CURRENT STATUS: ACTIVE

 CURRENT SYSID: CHI       JOURNAL: YES

 CURRENT HOME NODE: CHI   LOGDAYS: 30
 PASSWORD SYNC: YES       COMMAND: YES
 EXTENDED CPF: YES        CMDWAIT: NO
 UNDEFINED NODES: NO      JRNL QUICK START: NO

 DFTCMD: NYC
 DFTPSW: NYC LAS WAS
 JRNLRECV: CPF1.JRNLRECV
 JRNLSEND: CPF1.JRNLSEND
 RSTNLST: BOTH
 TRMNLST: BOTH

 -- NODE DEFINITIONS --
  NODE RECEIVE SEND GTWY UNI  TSS  VM  VM  VM  VM  VM  VM
  NAME  FROM  TO  NODE NODE NODE NODE LIDS RULE INFO LACCES 1ADAY
 ====== ==== ==== ==== ==== ==== ==== ==== ==== ==== ====== =====
 CHI    YES  YES   NO   NO   NO   NO   NO   NO   NO    NO    NO
 LAS    YES  YES   NO   NO   NO   NO   NO   NO   NO    NO    NO
 NYC    YES  YES   NO   NO   NO   NO   NO   NO   NO    NO    NO
 WAS    YES  YES   NO   NO   YES  NO   NO   NO   NO    NO    NO
```

Field Descriptions

CURRENT STATUS
Indicates whether CPF is active or inactive.

CURRENT SYSID
The SYSID used when CPF was started.

JOURNAL
Indicates whether CPF is using the journal files.

CURRENT HOME NODE
Indicates that the OPTIONS record on NYC lists NYC as the home node.

LOGDAYS
Indicates that undelivered commands stay on the CAICCI LOGGER database for 30 days.

PASSWORD SYNC
Indicates whether users can change their password on the current node and have the change propagated to remote nodes.

COMMAND
Indicates whether ACF subcommands entered at the current node can be propagated to remote nodes.

EXTENDED CPF Indicates whether Extended CPF is active on the current node.

UNDEFINED NODES
Indicates that this node does not process any request that does not have a NODEDEF record defined on this system.

CMDWAIT
Indicates that asynchronous processing is the default at this node. CPFWAIT must be specified on individual commands if you desire synchronous processing.

DFTCMD
These are the nodes that are sent the ACF command if the default list is not overridden.

DFTPSW
These are the nodes that are sent password synchronization requests when a password is changed at system entry.

JRNLRECV
This is the name of the journal data set that contains all inbound requests to this node and the response to the request.

JRNLSEND
This is the name of the journal data set that contains all outbound requests from this node and the response to the request.

NODE NAME
Indicates the names of NODEDEF records at CHI. These records have IDs of NODEDEF.CHI, NODEDEF.LAS, NODEDEF.NYC, and NODEDEF.WAS.

RECEIVE FROM
Indicates the setting of the INCMD field of the NODEDEF records.

SEND TO
Indicates the setting of the OUTCMD field of the NODEDEF records.

TSSNODE
Indicates if the node has been designated as an eTrust CA-Top Secret node.

GATEWAY NODE
Indicates if the node has been designated as a GATEWAY node. Any request from this node is forwarded to all other defined nodes. Any request from a non-GATEWAY node is forwarded to all nodes defined as GATEWAY nodes.

UNICENTER NODE
Indicates that this node has been designated as a CA-Common Services node. Requests from this node are forwarded to all other defined nodes. Requests from non-Unicenter nodes are forwarded to all nodes defined as UNINODE or GATEWAY.

VM NODE
Indicates that this is a VM node. This node is required for VM database synchronization and does not participate in normal CPF processing.

Chapter 14

Maintaining Global System Options Records

Global System Options Records Summary

Computer Associates selects record IDs for global system options (GSO) records. They are not site-definable or modifiable.

Each record has a unique set of fields. These predefined record IDs are listed in the following table, together with their basic functions:

Record ID	Function
APPLDEF	Defines the format of site-defined and other structured infostorage application records.
AUTHEXIT	Contains the vendor or site exit information that supports the secondary authentication facility.
AUTOERAS	Controls the automatic physical erasure of VSAM or non-VSAM data sets.
AUTOIDLX	Controls the automatic assignment of UID and GID values for PROFILE(USER),DIV(LINUX), and PROFILE(GROUP),DIV(LINUX), records.
AUTOIDOM	Controls the automatic assignment of UID and GID values for PROFILE(USER),DIV(OMVS) and PROFILE(GROUP),DIV(OMVS) records.
BACKUP	This record contains the CPU, command string information, and time when the automatic database backup utility is to occur. It can also control the space allocations for the backup work files.
BLPPGM	Specifies those programs that are authorized to use tape bypass label processing (BLP).
CACHESRV	Defines R_cacheserv cache names to eTrust CA-ACF2.

Record ID	Function
CRITMAP	Allows mapping of digital certificates to one of a number of eTrust CA-ACF2 logonids based on the system ID, application ID, or application-defined variables specified on the CRITMAP record.
CERTMAP	Allows mapping of multiple digital certificates to a single eTrust CA-ACF2 logonid.
CLASMAP	Translates an eight-character SAF resource class into a three-character eTrust CA-ACF2 resource type code, which lets you write resource rules to perform validation. CLASMAP also translates the resource type codes for eTrust CA-ACF2 calls or calls made to eTrust CA-ACF2 from Computer Associates International Standard Security Facility (CAISSF).
DELRSRC	Specifies a generalized resource or DB2 resource that is delegated in the system. Security administrators should only create delegated resources if an application explicitly requires it.
EIM	Provides support for IBM Enterprise Identity Mapping.
ETAUDIT	Implements event filter controls so only the selected eTrust CA-ACF2 security event notifications are transmitted to eTrust Audit.
EXITS	Specifies the module names of site-written eTrust CA-ACF2 exit routines.
INFODIR	Specifies the infostorage directories and rule sets that are to be made resident at eTrust CA-ACF2 initialization time.
LINKLST	Specifies one or more partitioned data sets that are considered part of the system link list (SYS1.LINKLIB) during data set access validation.
LINUX	Defines Linux machines to eTrust CA-ACF2.
LOGPGM	Specifies those programs for which all data set accesses are logged.
MAINT	Specifies the logonid, program, and library combinations used for system maintenance functions.
MLID	Specifies a logonid compression algorithm used in the MUSASS (Multiple-User, Single Address Space System) environment to reduce virtual storage requirements. Logon compression eliminates unused or unnecessary information from the resident portion of the logonid record.

Record ID	Function
MLSOPTS	Specifies Multilevel Security (MLS) global options available on a system.
MUSASS	Defines special processing to be performed by eTrust CA-ACF2 on behalf of a MUSASS (Multiple-User, Single Address Space System) to reduce eTrust CA-ACF2 storage requirements and CPU overhead.
NJE	Specifies eTrust CA-ACF2 validation options that apply to jobs submitted through a network job entry subsystem (JES2, JES3, RSCS).
OPTS	Specifies the global options available to the system.
PDS	Specifies partitioned data sets that will be protected at the member level.
PPGM	Specifices protected programs that can be executed only by privileged users.
PROXY	Specifies the default PROXY and EIM information.
PSWD	Specifies the user password controls.
REALM	Defines the characteristics of local and foreign Network Authentication and Privacy Services realms.
RESDIR	Specifies those resource rule directories that are to be made globally resident at eTrust CA-ACF2 initialization time.
RESRULE	Specifies those data set access rules that are to be made resident at eTrust CA-ACF2 initialization time.
RESVOLS	Specifies those DASD and mass storage volumes for which eTrust CA-ACF2 is to provide data-set-level protection.
RESWORD	Specifies the words or word prefixes that cannot be used in passwords.
RULEOPTS	Specifies the options pertinent to access and resource rule maintenance.
SAFDEF	Defines System Authorization Facility (SAF) calls that your site wants to process differently than the default eTrust CA-ACF2 process.
SECVOLS	Specifies those DASD and tape volumes for which eTrust CA-ACF2 is to provide volume-level protection.
SYNCOPTS	Defines the cache synchronization processing for a CPU running in a shared eTrust CA-ACF2 database environment.

Record ID	Function
SYSPLEX	Specifies options for eTrust CA-ACF2 use of command broadcast (XCS) and data sharing (XES) in the SYSPLEX environment.
STC	Assigns a logonid and optional groupid based on the started task ID.
TNGNODE	Specifies the CA-Common Services nodes that act as monitors for mainframe SNMP traps.
TSO	Specifies TSO/E system-wide options and default logon parameters.
TSOCRT	Specifies a screen-clear string used to obliterate the logon password on ASCII CRT devices.
TSOKEYS	Specifies site-supplied keywords that eTrust CA-ACF2 permits at TSO logon time.
TSOTWX	Specifies a cross-out mask used to obliterate the logon password on TWX devices.
TSO2741	Specifies a cross-out string used to obliterate the logon password on 2741 devices.
UNIXOPTS	Specifies global options pertinent to the UNIX System Services (OMVS) environment.
WARN	Specifies a warning message to be issued to the user when the system is in WARN mode and a violation is detected.

You can use ISPF panels and ACF commands to create and maintain GSO records. See Using the ISPF Panels and Using the ACF Command later in this chapter for more information.

You can create a privilege list for some fields of certain GSO records. A privilege list specifies the eTrust CA-ACF2 logonid privilege that a user must have to perform a particular function. See the next section for more information.

Detailed information on the GSO records is provided in the following sections.

SYSID Implications

GSO records are grouped in the InfoStorage database by SYSID. The SYSID determines which GSO records will be selected to be used by eTrust CA-ACF2 on a particular system. The SYSID can be specified as a startup parameter (for example, S ACF2,PARM='SYSID(PRD1)', or if no SYSID is specified on the START command, the SMFID will be used.

All GSO records are read from the InfoStorage database and are initially sorted, within ID, by their SYSID, most specific first and least specific last. For certain record Ids, only one record can be selected. eTrust CA-ACF2 will select the record with the SYSID that most specifically matches the SYSID for this iteration of eTrust CA-ACF2. These record Ids include: AUTOERAS, AUTOIDOM, BACKUP, CACHESRV, EIM, ETAUDIT, EXITS, INFODIR, LOGPGM, MLSOPTS, OPTS, PPGM, PROXY, PSWD, RESDIR, RESRULE, RESVOLS, RESWORD, RULEOPTS, SECVOLS, SYNCOPTS, SYSPLEX, TSO, TSOCRT, TSOKEYS, TSOTWX, TSO2741, UNIXOPTS, and WARN.

For the remaining record Ids, eTrust CA-ACF2 will select all records whose SYSID matches the specified SYSID, whether the match is specific or masked. The selected records are then sorted, within ID, by criteria based on the contents of various record fields. These record Ids are: APPLDEF, AUTOIDLX, AUTHEXIT, BLPPGM, CERTMAP, CRITMAP, CLASMAP, DELRSRC, LINKLST, LINUX, MAINT, MLID, MUSASS, NJE, PDS, REALM, SAFDEF, STC, and TNGNODE.

Structured Infostorage Record Privilege List

You can create a privilege list for some fields of certain GSO records. A privilege list specifies the eTrust CA-ACF2 logonid privilege that a user must have to perform a particular function.

For example, you can specify a privilege list for the DECOMP field of the GSO RULEOPTS record:

```
DECOMP(CONSULT)
```

Logonids with the SECURITY or AUDIT privilege, unless scoped, can **always** decompile access rules and resource rules. This example indicates that a user with the CONSULT field in his logonid is also able to decompile access or resource rules.

You can specify ALL in place of a privilege list. ALL means that any user defined to eTrust CA-ACF2 (SECURITY, ACCOUNT, AUDIT, LEADER, CONSULT, or USER) can use the function defined by the GSO record field.

As an example, you can specify the SELAUTH field of the GSO APPLDEF record as:

```
SELAUTH(ALL)
```

This specifies that any user with any standard eTrust CA-ACF2 privilege field in their logonid can use the SET and LIST subcommands for the structured infostorage application defined by the GSO APPLDEF record.

Structured Infostorage Record Definitions (APPLDEF)

The APPLDEF record lets you define your own structured infostorage records when your site has unique needs that are not fulfilled by the standard structured infostorage records, such as GSO records and TSO records. eTrust CA-ACF2 uses the APPLDEF record to validate access to your structured infostorage records.

The APPLDEF record defines the key structure for structured infostorage records, and identifies the record structure block (RSB) used to define the fields and where that RSB resides. A record structure block is a module that tells eTrust CA-ACF2 how to format and validate a structured infostorage record. It defines field lengths, validation routines, and other field characteristics.

Some popular uses for structured infostorage records are:

- Operator identification (OID) card support. See Appendix A, "Operator Identification Card Support."
- Extended user authentication (EUA) support. See the "Special Usage Considerations" chapter in the *Systems Programmer Guide*.
- VTAM common sign-on product support. See the "Special Usage Considerations" chapter in the *Systems Programmer Guide*.

For more in-depth descriptions of structured and unstructured infostorage records, see Creating Structured Infostorage Records in the "Special Usage Considerations" chapter in the *Systems Programmer Guide*.

The APPLDEF record is optional. If your site plans to use its own structured infostorage applications, you must create one APPLDEF record for each application. Only one APPLDEF record is required for each class and type. If more than one GSO APPLDEF record is retained for the same class and type of resource, the last record read determines which record structure block eTrust CA-ACF2 uses.

Record ID	Fields
APPLDEF*qual*	APPLDIV(*divmask*) APPLDLEN(*maxdivisionidlength*) BLANKS | NOBLANKS CLASS(*classcode* | *classname*) COMPILE | NOCOMPILE DFTDRTN(*defaultdivisionroutine*) EXTCOMPNOEXTCOMP RECID(*RSBmodule* | *recidmask*,...,*RSBmodule* | *recidmask*) RECIDLEN(*maxrecidlength*) RSBLIB(*rsblib*) SELAUTH(*privilegelist*) TYPE(*typecode* | *typename*)

Field Descriptions

APPLDIV(*divmask*)
Specifies the division used to group a set of related infostorage records. You can mask this field to refer to several different records in one structured infostorage application. For example, you can create an APPLDEF record that applies to all releases of IMS by specifying APPLDIV(-). The division mask is the third field of the record key. You can use it to access these records when you specify the SET subcommand.

If you do not specify a value for APPLDIV, you cannot specify DIVISION for any of the ACF subcommands.

APPLDLEN(*maxdivisionidlength*)
Specifies the maximum length of an application division value. The application division is specified with the APPLDIV field.

BLANKS | NOBLANKS
Specifies whether embedded blanks can be used in the record IDs for this application.

CLASS(*classcode* | *classname*)
Specifies the class name and code to be associated with the storage class of this structured infostorage application. The classcode is the actual numeric character used in the class field of the record key. (Alphabetic characters are reserved for use by eTrust CA-ACF2.) The classname is the name a user indicates on the SET subcommand of the ACF command to access these records.

COMPILE | NOCOMPILE
A Computer Associates-reserved control which designates whether the records stored under the infostorage class and type defined by this APPLDEF definition are not standard eTrust CA-ACF2 strucured infostorage records, but rather are special internal-format compiled records. The default, NOCOMPILE, means that the records are standard eTrust CA-ACF2 structured infostorage records. This default should be used in all cases unless otherwise instructed by Computer Associates in the way of product documenation, sample product installation jobs, or via other means of communication.

DFTDRTN(*defaultdivisionroutine*)
Indicates the default division routine to invoke if you omit division when you use the SET subcommand to access the records. The default is ACF00DFT, the eTrust CA-ACF2 standard default division routine.

EXTCOMP|NOEXTCOMP

A Computer Associates-reserved control which designates whether the records stored under the infostorage class and type defined by this APPLDEF definition are not standard eTrust CA-ACF2 structured infostorage records, but rather are special internal-format compiled records. The default value, NEXTCOMP, means that the records are standard eTrust CA-ACF2 structured infostorage records. This default should be used in all cases unless otherwise instructed by Computer Associates in the way of product documenation, sample product installation jobs, or via other means of communication.

RECID(*RSBmodule*|*recidmask*,...,*RSBmodule*|*recidmask*)

Indicates the record IDs for use with this structured infostorage application and the record structure block (RSB) to be associated with those record IDs. The recidmask is the name of the record ID that is used in the record key of these records. The recidmask is completely maskable, so several or all record IDs can use the same RSB. The RSB defines the fields of the records. (Creating an RSB is described in the "Special Usage Considerations" chapter of the *Systems Programmer Guide*.) RSBmodule is the name of the RSB that defines the fields for the associated record IDs of this structured infostorage application.

RECIDLEN(*maxrecidlength*)

Indicates the maximum length of the record ID. The record ID is specified with the RECID field. The default value is zero.

RSBLIB(*rsblib*)

Indicates the library into which the RSB has been link-edited. The value specified for RSBLIB should be a fully qualified data set name. eTrust CA-ACF2 searches the Link Pack Area (LPA) for the RSB. If the RSB is not found in the LPA, eTrust CA-ACF2 loads the RSB from the RSBLIB. The default value is null, which indicates to eTrust CA-ACF2 that the RSB can be found in the LPA.

SELAUTH(*privilegelist*)

Specifies the eTrust CA-ACF2 privilege users must have to use the SET and LIST subcommands for the structured infostorage application defined by this APPLDEF record. The SET subcommand lets users access the structured infostorage application, and the LIST subcommand lets them display the field values of the structured infostorage record. Specify any of the eTrust CA-ACF2 privileges in the logonid record (SECURITY, ACCOUNT, AUDIT, CONSULT, LEADER, and USER) for the SELAUTH value. You can specify more than one privilege.

If you specify ALL, a user with any of the standard eTrust CA-ACF2 logonid privilege fields can use the SET and LIST subcommands for the defined application. Because the SECURITY privilege permits full access authority, the specification of SECURITY in this field lets anyone with the SECURITY privilege defined in his or her logonid record issue other ACF subcommands besides SET and LIST; they can issue INSERT, CHANGE, and DELETE subcommands as well. Any of the other access privileges, such as AUDIT, allow the use of the SET and LIST subcommands. If no value is specified, the default is SECURITY, AUDIT, ACCOUNT, CONSULT, and LEADER.

When you tell eTrust CA-ACF2 users to bypass INFOLIST validation processing for the structured infostorage record defined by this record (via the ACRSB macro), it checks the value you specify in the SELAUTH field only to determine the authority required to access the record. Specify this field with care.

For information about INFOLIST bypass processing, see the "Special Usage Considerations" chapter in the *Systems Programmer Guide*.

TYPE(*typecode*|*typename*)
Specifies the type name and code to be associated with the storage class of this structured infostorage application. The type code is the actual three characters used as the type field of the record key. The type name is the name a user indicates on the SET subcommand of the ACF command to access these records.

Creating Multiple GSO APPLDEF Records

If you need more than one APPLDEF record, you can append a qualifier to the record name in the format APPLDEF*qual* to generate a unique record ID (for example, APPLDEF001 or APPLDEF.RSBA). This optional qualifier can be up to nine characters, and must immediately follow the characters APPLDEF. If you use a period (.) as part of the qualifier string for the record name, eTrust CA-ACF2 counts it as one of the nine available characters.

Displaying GSO APPLDEF Record Information

SHOW APPLDEF displays the applications defined to the system.

```
-- INSTALLATION DEFINED STRUCTURED INFOSTORAGE APPLICATIONS --

CLASS (SHORT / LONG): I / IDENTITY
TYPE (SHORT / LONG): AUT / AUTHSUP
SELECTION AUTHORIZATION: SECURITY, ACCOUNT, AUDIT, CONSULT, LEADER
DEFAULT DIVISION ROUTINE: ACF00DFT
ACTIVE DIVISION: AUTHSUP3
ASSOCIATED RSB / RECORD ID: ACF2RSB1 / ********

CLASS (SHORT / LONG): I / IDENTITY
TYPE (SHORT / LONG): AUT / AUTHSUP
SELECTION AUTHORIZATION: SECURITY, ACCOUNT
DEFAULT DIVISION ROUTINE: ACF00DFT
ACTIVE DIVISION: AUTHSUP5
ASSOCIATED RSB / RECORD ID: ACF2RSB2 / ********
```

If you have not defined any GSO APPLDEF records, the SHOW APPLDEF subcommand displays the title:

```
--NO APPLICATION DEFINITIONS EXIST ON THIS SYSTEM--
```

Extended User Authentication Exit (AUTHEXIT)

The AUTHEXIT record defines the name of an extended user authentication exit to be invoked during TSO logon. This exit processes logonids with the corresponding LIDFIELD attribute. You can define as many as eight extended user authentication exits, one per AUTHEXIT record.

AUTHEXIT records for operator identification (OID) card support are described in Appendix A, "Operator Identification Card Support." More information about extended user authentication support is found in the "Special Usage Considerations" chapter of the *Systems Programmer Guide*.

Record ID	Fields
AUTHEXIT*qual*	INFOSTG | NOINFOSTG LIDFIELD(*attributename*) PROCPGM(*processingprogramname*)

Field Descriptions

INFOSTG | NOINFOSTG
Indicates whether the extended authentication exit stores information in the Infostorage database. You define information about the record format and type in a corresponding GSO APPLDEF record. INFOSTG is the default. The actual data required by the authentication program is stored in the eTrust CA-ACF2 Infostorage database. You can create these records under the IDENTITY(AUT) setting. See the "Maintaining Identity Records" chapter for more information.

If your site uses extended user authentication, you can specify a total of eight different AUTHEXIT records. If you select the INFOSTG option, you must also define a corresponding GSO APPLDEF record for each AUTHEXIT record.

LIDFIELD(*attributename*)
Specifies the logonid attribute that triggers the extended user authentication exit named in the associated PROCPGM field. This attribute name can be up to eight characters long. It must be defined in an @CFDE macro in the ACFFDR.

PROCPGM(*processingprogramname*)
Specifies the name of the extended user authentication exit. This exit program is invoked during TSO logon validation if the logonid contains the appropriate LIDFIELD attribute. This name can be up to eight characters long.

Creating Multiple GSO AUTHEXIT Records

If you need more than one AUTHEXIT record, you can append a qualifier to the record name in the format AUTHEXIT*qual* to generate a unique record ID (for example, AUTHEXIT001 or AUTHEXIT.CARD). This optional qualifier can be up to eight characters, and must immediately follow the characters AUTHEXIT. If you use a period (.) as part of the qualifier string for the record name, eTrust CA-ACF2 counts it as one of the eight available characters.

Displaying GSO AUTHEXIT Records

The SHOW ACTIVE and SHOW ACF2 subcommands display the active GSO AUTHEXIT records in the site.

```
--AUTHENTICATION EXITS ON THIS SYSTEM: LIDFLD/PROCESS PROGRAM/INFOSTG

  AUTHSUP2/ACFEAXIT/INFOSTG
```

The SHOW FIELDS subcommand displays the logonid attribute names defined by the site.

Automatic Erase Feature (AUTOERAS)

The AUTOERAS record specifies whether you want eTrust CA-ACF2 to physically erase non-VSAM or VSAM data sets when you delete them. If you specify NON-VSAM or VSAM, eTrust CA-ACF2 erases the data sets before it releases the space.

Record ID	Fields
AUTOERAS	NON-VSAM | NONON-VSAM VSAM | NOVSAM VOLS(*volser,volmask1,...,volmask255*)

Field Descriptions

NON-VSAM | NONON-VSAM
Specifies whether eTrust CA-ACF2 automatically erases non-VSAM data sets before it releases the space for future use. The non-VSAM erase is invoked through JCL disposition processing, dynamic unallocation (SVC99), a system utility (IEHPROGM), or a user program. The default is NONON-VSAM, which deactivates automatic erase for non-VSAM data sets. Read the VOLS description for further information.

VSAM | NOVSAM
Specifies whether eTrust CA-ACF2 automatically erases VSAM data spaces. The VSAM erase is invoked during IDCAMS delete processing. If VSAM is specified, all VSAM data spaces are automatically erased during IDCAMS delete processing. The default of NOVSAM deactivates the VSAM automatic erase feature.

VOLS(*volser,volmask1,...,volmask255*)
Identifies a set of DASD volumes. eTrust CA-ACF2 automatically erases a non-VSAM data set if the deleted data set resides on one of the volumes. VOLS applies only to non-VSAM data sets. You can specify a maximum of 255 volume serials and volume serial masks.

You can specify this parameter as a list of one to six-character volume serial numbers or as masked patterns. For example, if you specify NON-VSAM VOLS(PROD01,PROD06), only non-VSAM data sets that reside on the DASD volumes PROD01 and PROD06 are automatically erased. Data sets that reside on other volumes (PROD03 or PROD04, for example) are not automatically erased.

The following chart illustrates the various protection mechanisms and the effect on the data types shown:

Data Type	JCL	SVC99	eTrust CA-ACF2 Utilities	Automatic Erase
VSAM	I	I	U	S
Non-VSAM	I	I	U(ACF2)	S
Temporary Non-VIO	I	I	U(ACF2)	S
VIO	S	S	S	S

Where:

- **I**—impossible
- **U**—user action required
- **U(ACF2)**—possible with user-invoked eTrust CA-ACF2 utilities
- **S**—action automatically taken by system

Both the VSAM and non-VSAM automatic erase functions are dynamically implanted by eTrust CA-ACF2 at system-provided exit points. They function through batch or online and are not dependent on the use of an indexed VTOC.

Displaying GSO AUTOERAS Record Information

The SHOW STATE subcommand displays the options and values defined in the GSO AUTOERAS record. SHOW ACTIVE indicates which dynamically implanted intercepts have gained control.

Automatic UID/GID Assignment Options (AUTOIDLX)

The AUTOIDLX record defines options for the automatic assignment of UID and GID values for PROFILE(USER), DIV(LINUX), and PROFILE(GROUP) DIV(LINUX) records. The AUTOIDLX record is never propagated to other nodes in a CPF environment.

Note: Automatic UID and GID value assignment does not guarantee that the assigned value will be unique. The value in UIDNEXT or GIDNEXT, as appropriate, is used for the automatic assignment. This occurs whether or not another record exists with the same value for UID or GID. If uniqueness is desired, use the SHOW LINUX command to determine a starting and ending value range that is not in use.

Record ID	Fields
AUTOIDLX*qual*	ASSIGNU | NOASSIGNU ASSIGNG | NOASSIGNG UIDSTART(*nnn*) UIDEND(*nnn*) UIDNEXT(*nnn*) GIDSTART(*nnn*) GIDEND(*nnn*) GIDNEXT(*nnn*)

Field Descriptions

qual

Specifies the LINUX machine this record is for. Valid qualifiers are LINUX short machine names (maximum eight-bytes). If you use a period (.) as part of the qualifier string for the record name, eTrust CA-ACF2 counts it as one of the eight available characters. If *qual* is specified, this record is used to auto-assign UID and/or GID values for requests that relate to the Linux machine that matches *qual*. If *qual* is not specified, this record is the default record used to auto-assign UID and/or GID values.

ASSIGNU|NOASSIGNU

ASSIGNU indicates that the LINUXUID can be auto-assigned when the AUTOUIDL keyword is specified or implied when inserting or changing a LINUX User Profile data record.

If NOASSIGNU is specified, the AUTOUIDL keyword is considered an error. The default is ASSIGNU. For more information on AUTOUIDL, see User Profile Records in the "Maintaining Profile Records" chapter.

ASSIGNG|NOASSIGNG

ASSIGNG indicates that the GID can be auto-assigned when the AUTOGIDL keyword is specified or implied when inserting or changing a LINUX Group Profile data record.

If NOASSIGNG is specified, the AUTOGIDL keyword is considered an error. The default is ASSIGNG. For more information on AUTOGIDL, see Group Profile Records in the "Maintaining Profile Records" chapter.

UIDSTART

Start of UID value range for auto-assignment. A number between 500 and 2,147,483,647 is valid, the default is 500.

UIDEND

End of UID value range for auto-assignment. A number between 501 and 2,147,483,647 is valid, the default is 2,147,483,647. UIDEND must be larger than UIDSTART.

UIDNEXT

Next available UID value. This field is maintained internally by eTrust CA-ACF2 and cannot be modified by the user directly.

UIDNEXT is set to -1 when all values have been used and UIDEND has been set to 2,147,483,647. When UIDNEXT has been set to –1, change the AUTOID record specifying new UIDSTART and UIDEND values. This will reset the UIDNEXT value to equal the new UIDSTART value.

UIDNEXT is set to the next value when the UIDEND value has been automatically assigned. For example, if UIDEND is 5000 and UID(5000) has just be automatically assigned, then UIDNEXT will be set to 5001. When UIDNEXT is higher than UIDEND, no automatic assignment can take place. In this case, UIDEND can be set to a higher number, or UIDSTART and UIDEND can be set to some other range of values.

GIDSTART

Start of GID value range for auto-assignment. A number between 100 and 2,147,483,647 is valid, the default is 100.

GIDEND

End of GID value range for auto-assignment. A number between 101 and 2,147,483,647 is valid, the default is 2,147,483,647. GIDEND must be larger than GIDSTART.

GIDNEXT

Next available GID value. This field is maintained internally by eTrust CA-ACF2 and cannot be modified by the user directly.

GIDNEXT is set to -1 when all values have been used and GIDEND has been set to 2,147,483,647.

When GIDNEXT has been set to –1, change the AUTOID record specifying new GIDSTART and GIDEND values. This will reset the GIDNEXT value to equal the new GIDSTART value.

GIDNEXT is set to the next value when the GIDEND value has been automatically assigned. For example, if GIDEND is 5000 and GID(5000) has just be automatically assigned, then GIDNEXT will be set to 5001.

When GIDNEXT is higher than GIDEND, no automatic assignment can take place. In this case, GIDEND can be set to a higher number, or GIDSTART and GIDEND can be set to some other range of values.

Refresh Command

Note, using the F ACF2,REFRESH(AUTOIDLX) command to refresh the AUTOID record will cause the AUTOIDLX records to be refreshed for the current SYSID or the SYSID as stated on the REFRESH command.

Automatic UID/GID Assignment Options (AUTOIDOM)

The AUTOIDOM record defines options for the automatic assignment of UID and GID values for PROFILE(USER),DIV(OMVS), and PROFILE(GROUP),DIV(OMVS) records. The AUTOIDOM record is never propagated to other nodes in a CPF environment.

Record ID	Fields
AUTOIDOM	ASSIGNU | NOASSIGNU ASSIGNG | NOASSIGNG UIDSTART(*nnn*) UIDEND(*nnn*) UIDNEXT(*nnn*) GIDSTART(*nnn*) GIDEND(*nnn*) GIDNEXT(*nnn*)

Field Descriptions

ASSIGNU|NOASSIGNU
ASSIGNU indicates that the UID can be auto-assigned when the AUTOUID keyword is specified or implied when inserting or changing an OMVS User Profile data record.

If NOASSIGNU is specified, the AUTOUID keyword is considered an error. The default is ASSIGNU. For more information AUTOUID, see User Profile Records in the "Maintaining Profile Records" chapter.

ASSIGNG|NOASSIGNG
ASSIGNG indicates that the GID can be auto-assigned when the AUTOGID keyword is specified or implied when inserting or changing an OMVS Group Profile data record.

If NOASSIGNG is specified, the AUTOGID keyword is considered an error. The default is ASSIGNG. For more information on AUTOGID, see Group Profile Records in the "Maintaining Profile Records" chapter.

UIDSTART
Start of UID value range for auto-assignment. A number between 1 and 2,147,483,647 is valid; the default is 1.

UIDEND
End of UID value range for auto-assignment. A number between 1 and 2,147,483,647 is valid; the default is 2,147,483,647. UIDEND must be larger than UIDSTART.

UIDNEXT
Next available UID value, or -1 when all values have been used. This field is maintained internally by eTrust CA-ACF2 and cannot be modified by the user directly. When UIDNEXT has been set to –1, change the AUTOIDOM record specifying new UIDSTART and UIDEND values. This will reset the UIDNEXT value to equal the new UIDSTART value.

GIDSTART
Start of GID value range for auto-assignment. A number between 1 and 2,147,483,647 is valid; the default is 1.

GIDEND
End of GID value range for auto-assignment. A number between 1 and 2,147,483,647 is valid; the default is 2,147483,647. GIDEND must be larger than GIDSTART.

GIDNEXT
Next available GID value, or -1 when all values have been used. This field is maintained internally by eTrust CA-ACF2 and cannot be modified by the user directly. When GIDNEXT has been set to –1, change the AUTOIDOM record specifying new GIDSTART and GIDEND values. This will reset the GIDNEXT value to equal the new GIDSTART value.

Refresh Command

Note, using the F ACF2,REFRESH(AUTOIDOM) command to refresh the AUTOIDOM record will cause the AUTOIDOM record to be refreshed for the current SYSID or the SYSID as stated on the REFRESH command.

Automatic Backup Options (BACKUP)

The BACKUP record specifies the eTrust CA-ACF2 automatic backup procedures for the Logonid, Rule, and Infostorage databases. This record specifies a command that eTrust CA-ACF2 issues internally upon successful completion of backup processing. It can also dynamically allocate the eTrust CA-ACF2 backup work files if they were not preallocated. See the *Getting Started* guide for more information.

Record ID	Fields
BACKUP	BUFNO(1 | *nn*) CPUID(*smfid*) PRISPACE(5 | *nnn*) SECSPACE(5 | *nnn*) STRING(*string*) TIME(00:01 | *hh:mm*) SYSUT1 | NOSYSUT1 WORKVOL(*volser*) WORKUNIT(VIO | *devicetype*) #UNITS(1 | nn)

Field Descriptions

BUFNO(*nn*)

Designates the number of buffers that will be specified on the SYSUT1 and BACKUP file DCBs at OPEN time. If nothing is specified the system provides 5 buffers. The BUFNO value can be up to 25. If there are plans to increase the BUFNO value, the REGION parameter on the ACF2 PROC must be increased.

CPUID(*smfid*)
Specifies the SMF ID of the CPU designated to take the automatic backups in a multi-CPU environment. If you specify this field, eTrust CA-ACF2 compares it with the actual MVS system SMF ID. eTrust CA-ACF2 bypasses the automatic backup if the two do not match. Operators can take backups at any time from any CPU. You should designate a single CPU in a multi-system configuration as the sole automatic backup processor. Masking cannot be used in the CPUID( ) field value. ACF2 will interpret the dash and the asterisk as literal values and not as masking characters.

PRISPACE(5 | *nnn*)
Specifies the amount of primary work space to be allocated for eTrust CA-ACF2 backup processing. The default value is five. The units are expressed in cylinders. This field does not display if not entered.

SECSPACE(5 | *nnn*)
Specifies the amount of secondary workspace to be allocated for eTrust CA-ACF2 backup processing. The default value is five. Units are expressed in cylinders. This field does not display if not entered.

STRING(*string*)
Specifies a text string that you want eTrust CA-ACF2 to issue when it completes its backup. This text is usually an MVS START console command used to perform additional site-required processing. As part of the eTrust CA-ACF2 database recovery facility, a procedure named ACFBKUP is placed into SYS1.PROCLIB during the installation process. You can use ACFBKUP or a similar facility to REPRO the primary sequential backup data sets into the alternate VSAM clusters.

If you do not specify a string, eTrust CA-ACF2 does not issue a console command.

SYSUT1 | NOSYSUT1
Indicates whether the copy to the interim SYSUT1 file will be bypassed. Normally the VSAM file is first copied to the SYSUT1 file before being copied to the backup file.

TIME(*hh:mm* | 00:01)
Specifies the time of day (24-hour format) when the backup is initiated. The default is 00:01 AM. If you specify TIME(00:00), eTrust CA-ACF2 does not perform a backup.

WORKVOL(*volser*)
Specifies the volser of the volume where the backup work files are allocated. There is no default value for the field. This field does not display if not entered.

WORKUNIT(VIO | *devicetype*)
Indicates the device type on which eTrust CA-ACF2 is to dynamically allocate its work files for backup processing. Device names are VIO, SYSDA, or DISK (VIO is the default). You can also use a name of your own choice. This field does not display if not entered.

#UNITS(1 | nn)
Indicates number of units associated with the back-up data set. Will only be used if WORKVOL has not been specified. The default value of #UNITS is 1.

BACKUP Processing

As eTrust CA-ACF2 backup processing progresses from cluster to cluster, it displays the number of records copied from each database. You must meet the following requirements before you can use ACFBKUP:

- Modify the JCL to reflect the data set names for both the primary sequential backup files and the alternate VSAM clusters at your site.
- Initialize the alternate VSAM clusters. Specify the STRING field as STRING(S ACFBKUP).
- The backup work files, such as SYSUT1, can be specified in the eTrust CA-ACF2 proc to be allocated at eTrust CA-ACF2 startup, or, they can be dynamically allocated when BACKUP commences using the PRISPACE, SECSPACE, and WORKVOL/WORKUNIT fields of the BACKUP record. If the work files are dynamically allocated, their storage is released as soon as backup completes. Otherwise, the allocated storage is retained until eTrust CA-ACF2 terminates.

To deactivate the automatic backup, change the GSO BACKUP record to specify TIME(00:00). You can specify the NOBACKUP at eTrust CA-ACF2 startup time to deactivate the automatic backup facility for an individual CPU in a multi-CPU complex. Alternatively, you can specify the CPUID field to designate the backup CPU. You can also start backup processing by issuing the console command:

```
F ACF2,BACKUP
```

Displaying GSO BACKUP Record Information

The SHOW SYSTEMS and SHOW ACF2 subcommands display the values specified for fields in the BACKUP record.

Tape Bypass Label Access Option (BLPPGM)

The BLPPGM record defines the programs and associated libraries that are authorized to use tape bypass label processing (BLP). When you specify this record, eTrust CA-ACF2 grants a specified program from the designated library BLP access even if the logonid executing the program does not have the TAPE-BLP or TAPE-LBL privileges in the logonid record.

Record ID	Fields
BLPPGM*qual*	LIBRARY(*library*) PGM(*pgm1,...,pgm256*)

Field Descriptions

LIBRARY(*library*)

Defines the fully qualified name of the library where the programs you specify in the PGM parameter reside. However, if the library is also specified in the GSO LINKLST record, the library name that you specify in the GSO BLPPGM record should be SYS1.LINKLIB. For example, if PROD.LOADLIB is specified in the LINKLST record, SYS1.LINKLIB should be specified in the BLPPGM record. Based on the entry in the LINKLST record, eTrust CA-ACF2 validates accesses from PROD.LOADLIB as coming from SYS1.LINKLIB.

If you do not specify the LIBRARY field or you specify a null value for LIBRARY, that is, LIBRARY(), eTrust CA-ACF2 rejects the BLPPGM records during refresh processing and does not use them for validation.

PGM(*pgm1,...,pgm256*)

Defines up to 256 program names.

Creating Multiple GSO BLPPGM Records

If you need more than one BLPPGM record, append a qualifier to the record name in the format BLPPGM*qual* to generate a unique record ID (for example, BLPPGM001 or BLPPGM.LOADLIB). This optional qualifier can be up to ten characters, and must immediately follow the characters BLPPGM. If you use a period (.) as part of the qualifier string for the record name, eTrust CA-ACF2 counts it as one of the ten characters.

Displaying Programs and Libraries Authorized for Tape Bypass Label Access

The SHOW PROGRAMS (or SHOW PGMS) and the SHOW ACF2 subcommands of the ACF command display all programs and library combinations that are authorized for tape bypass label access.

R_cacheserv Cache Names (CACHESRV)

The CACHESRV record defines the cache names that can be stored on a file on behalf of the R_cacheserv SAF Callable Service. Caches stored on a file are called "hardened". For detailed information on the R_cacheserv SAF Callable Service, see the IBM manual *z/OS Security Server RACF Callable Services.*

Note: See the *Getting Started Guide* for important information about defining the file that contains the hardened caches.

Record ID	Fields
CACHESRV	ACTIVE|NOACTIVE NAMES(*cache names)*

Field Descriptions

ACTIVE|NOACTIVE
Specifies whether cache hardening to a file is active or not. The default is ACTIVE.

NAMES(cache names)
Specifies a list of cache names that may be hardened to a file by the R_cacheserv SAF Callable Service. Cache names must be exactly 6 characters long, must start with the capital letter R, and contain only the following characters: A-Z, 0-9, @, #, and $. Blanks are not allowed.

Certificate Name Filtering Options (CERTMAP)

Certificate Name Filtering allows the mapping of multiple digital certificates to a single eTrust CA-ACF2 logonid. Optionally, a single digital certificate can be mapped to one of a number of eTrust CA-ACF2 logonids based on the system ID, application ID, or application defined variables.

Record ID	Fields
CERTMAP.*recid*	CRITERIA(*criteria-name-template*) DSN(*data-set-name*) IDNFILTR(*issuer's-dist-name-filter*) LABEL(*32-byte-label*) MULTIID | NOMULTIID SDNFILTR(*subject's-dist-name-filter*) TRUST | NOTRUST USERID(*userid-to-map-to*)

Field Descriptions

CRITERIA(*criteria-name-template*)

When specified with the MULTIID field, CRITERIA indicates a dynamic user ID mapping. The user ID associated with this mapping profile is based not only on the issuer's distinguished name and the subject's distinguished name found in the certificate, but also on additional criteria. CRITERIA specifies the criteria in the form of one or more variable names, separated by freeform text. These variable names begin with an ampersand (&) and are separated by periods. The freeform text should identify the variables contained in the template:

variable-name1=&name1.variable-name2=&name2

For example, if the application ID and system ID are to be considered in determining the user ID associated with this mapping, the CRITERIA parameter should be specified as follows:

```
CRITERIA(APPLID=&APPLID.SYSID=&SYSID)
```

The eTrust CA-ACF2-defined criteria are the application ID (APPLID) and the system ID (SYSID). When a user presents a certificate to the system for identification, the identity of the application (as well as the system the user is trying to access) becomes part of the criteria. The application passes its identity to eTrust CA-ACF2, and eTrust CA-ACF2 determines the system identifier. The system ID is the 4-character value specified for the SID parameter of the SMFPRMxx member of SYS1.PARMLIB. This value is substituted for &SYSID in the criteria.

Once the substitution is made, the fully expanded criteria field is used to find a matching profile defined in the CRITMAP GSO records. For example, APPLID=BANKU.SYSID=SYSA is the definition if the application being accessed is BANKU on the SYSA system. You should create a CRITMAP GSO record for this profile. The logonid to be associated with these certificates must be specified as the USERID. The APPLID and SYSID values of the CRITMAP record can be masked so that a record containing APPLID(BANKU) SYSID(*) allows the certificates to be used on any system, rather than just system SYSA. While masking characters can be used in the CRITMAP parameters, they should not be specified in the CRITERIA keyword on the CERTMAP GSO record.

Criteria names other than APPLID and SYSID are allowed, but are effective in certificate name filtering if the application supplies these criteria names and their associated values to eTrust CA-ACF2 when the user attempts to access the application using a certificate. SYSID is determined by eTrust CA-ACF2, but APPLID must be specified with the initACEE callable service. Criteria names, such as APPLID and SYSID, should only be specified if the application instructs you do so.

A maximum of 255 characters can be entered when specifying the CRITERIA keyword. The values can be entered in any case, but are made upper case by eTrust CA-ACF2 because they must match upper case names in the CRITMAP record to be effective. When specifying the criteria value, note that the maximum length of the APPLVAR field on the CRITMAP record is also 254.

The CRITERIA keyword can only be set for MULTIID.

DSN(*data-set-name*)

When IDNFILTR, SDNFILTR, or both are specified along with DSN on the CERTMAP record, the value in the filter fields must correspond to a starting point within the respective distinguished name found in the certificate contained in the data set. You should specify enough of the name to precisely identify the starting point for the filter. For example, if the certificate in the data set has the following subject:

```
CN=John Jones.OU=Accounts Payable.O=My Company.L=internet
```

and you want all certificates for anyone in:

```
Accounts Payable
```

to be selected by this filter, you must specify:

```
SDNFILTR('OU=Acc')
```

Without the data set containing the certificate, you must enter the following to produce the same result:

```
SDNFILTR(OU=Accounts Payable.O=My Company.L=internet')
```

When a starting point value is specified for a certificate contained in a data set, there cannot be more than 255 characters between the starting point and ending point of the subject's name in the certificate.

IDNFILTR('*issuer's-distinguished-name-filter*')
Specifies the "significant" portion of the issuer's distinguished name that is used as a filter when associating a logonid with a certificate. If this field contains blanks, the entire filter must be encased in quotes; otherwise quotes are not needed.

When the CERTMAP record is inserted or changed without the DSN parameter, you must specify the entire portion of the distinguished name to be used as the filter.

The format of the *issuer's-distinguished-name-filter* variable is the same as the subject's-distinguished-name-filter variable in the SDNFILTR field. See the SDNFILTR field description for a more complete description of the values that are permitted in the filter parameters.

A maximum of 255 characters can be entered for IDNFILTR.

IDNFILTR is optional if SDNFILTR is specified. If IDNFILTR is not specified, only the subject's name is used as a filter. If IDNFILTR is specified and only a portion of the issuer's name is used as the filter, SDNFILTR must not be specified.

If both IDNFILTER and SDNFILTR are specified, the IDNFILTR value does not need to begin with a valid prefix from the previous list. This allows the use of certificates from a certificate authority that chooses to include non-standard data in the issuer's distinguished name.

LABEL(*label-name*)
Up to 32 characters can be specified for label-name. It can contain embedded blanks and mixed-case characters. The LABEL field can be used as a record identifier when changing or deleting CERTMAP records. For example, when changing the CERTMAP.INTERNET record, which has a LABEL of A1, to TRUST status, one of the following commands would work.

```
Change certmap.internet trust
```

```
Change certmap label(A1) trust
```

Note: If you must change the actual value of the LABEL field itself, use the keyword NEWLABEL. For example, to change the label value from A1 to A2, enter the following command:

```
Change certmap.internet newlabel(A2)
```

MULTIID | NOMULTIID
MULTIID tells eTrust CA-ACF2 to use the values specified for CRITERIA. If MULTIID is specified, then CRITERIA must be specified. If MULTIID is not specified, CRITERIA is ignored and USERID must be specified. The default is MULTIID.

recid
Specifies the unique one to eight-character name of the record ID.

SDNFILTR('*subject's-distinguished-name-filter*')

Specifies the "significant" portion of the subject's distinguished name. This is the part of the name that is used as a filter when associating a logonid with a certificate. If this field contains blanks, the entire filter must be encased in quotes, otherwise quotes are not needed.

When the CERTMAP record is inserted or changed without the DSN parameter, you must specify the entire portion of the distinguished name to be used as the filter.

The format of the *subject's-distinguished-name-filter* variable is similar to the output displayed when a certificate is displayed with the LIST command. It is an X.509 distinguished name in an address type format:

```
component.component.component.component...
```

Or, more specifically:

```
Qualifier1=node1.qualifier2=node2....qualifiern=noden
```

For example:

```
SDNFILTR('CN=John Jones.OU=Accounts Payable.O=My Company.L=internet')
```

The value specified for SDNFILTR must begin with a prefix found in the following list, followed by an equal sign (X'7E'). Components should be separated by a period (X'4B'). The case, blanks, and punctuation displayed when the digital certificate information is listed must be maintained in the SDNFILTR. Since digital certificates only contain characters available in the ASCII character set, the same characters should be used for the SDNFILTR value. Valid prefixes are:

```
COUNTRY             Specified as C=
STATE/PROVINCE      Specified as ST=
LOCALITY            Specified as L=
ORGANIZATION        Specified as O=
ORGANIZATIONAL UNIT Specified as OU=
TITLE               Specified as T=
COMMON NAME         Specified as CN=
STREET Address      Specified as STREET=
POSTAL CODE (Zip)   Specified as PC=
SERIALNUMBER        Specified as SERIALNUMBER=
```

A maximum of 255 characters can be entered for SDNFILTR.

SDNFILTR is optional if IDNFILTR is specified. If SDNFILTR is not specified, only the issuer's name is used as a filter. SDNFILTR must not be specified with IDNFILTR unless the value of IDNFILTR results in the entire issuer's name being used in the filter. Note that subject's name can be partial but cannot be used in a filter that contains only a partial issuer's name.

Use of the SERIALNUMBER prefix is only appropriate when certificates contain the serialNumber attribute within the subject-distinguished name. The serialNumber attribute in the subject-distinguished name is an infrequently used attribute, which may contain a value such as a router serial number. The subject distinguished name serialNumber attribute is unrelated to the certificate serial number (certSerialNumber). The SERIALNUMBER prefix cannot be used for filtering based on the certificate serial number.

TRUST|NOTRUST
When TRUST or NOTRUST is specified it indicates whether this mapping can be used to associate a logonid to a certificate presented by a user accessing the system. If neither is specified, the default is NOTRUST.

USERID(logonid)
Specifies the logonid of the user that is used when a certificate matches this record.

SAF Resource Classes (CLASMAP)

The CLASMAP record translates eight-character resource classes into three-byte eTrust CA-ACF2 resource type codes. The three-character resource type code lets you write resource rules to validate security calls for the specified classes. eTrust CA-ACF2 checks the CLASMAP record for this type code for all SAF, #SECUR, CAISSF, HLI and user SVCA calls that set ACG8RTYP and ACG8CRTF.

For a brief description of the strategy eTrust CA-ACF2 uses in searching CLASMAP records to determine the type code for a resource call, see Components of the eTrust CA-ACF2 SAF Interface in the "Understanding SAF" chapter. For a list of IBM-supplied SAF resource classes, see Appendix B, "IBM-Supplied Resource Classes."

Record ID	Fields
CLASMAP*qual*	ENTITYLN(0|*entitylength*) LOG|NOLOG MIXED|NOMIXED MUSID(*musassid*|********) POSIT(*positvalue*) PROFINT|NOPROFINT RESOURCE(*class*) RSRCTYPE(*typecode*)

Field Descriptions

ENTITYLN(0|*entitylength*)
Specifies the entity length of the specified SAF class. If ENTITYLN is zero, CA-ACF2 will check for a matching internal CLASMAP and assign the length from the internal CLASMAP. If no matching internal CLASMAP exists, then CA-ACF2 will assign a length of 39, the IBM default.

LOG|NOLOG
Specifies whether ACF2 will override the LOG parameter on a matching RACROUTE AUTH call and treat it as LOG=ASIS. This is a way of logging to SMF a violation that is not normally logged because the RACROUTE AUTH call specified LOG=NONE or LOG=NOFAIL or LOG=NOSTAT. NOLOG is the default. Note that LOG|NOLOG in the CLASMAP does not affect RACROUTE AUTH calls that are logged. NOLOG will not prevent loggings.

MIXED|NOMIXED
Indicates whether eTrust CA-ACF2 accepts all input of resource names as mixed case. When MIXED is chosen, the inserted CLASMAP must be made active via the ACF2 REFRESH command before any subsequent administration commands can be issued for this resource class.

MUSID(*musassid*|******)**
Identifies the MUSASS to which the CLASMAP record applies. This lets several MUSASSes that share the same resource class use a different type code. Normal eTrust CA-ACF2 resource name masking conventions apply.

POSIT(*positvalue*)
Specifies the bit value that will be checked in a bit table to determine whether a class is active in cases when a RACROUTE call is not issued. Valid values are: 19-56 and 128-527.

PROFINT|NOPROFINT
Specifies whether the profile interpreter should be invoked for the profile record associated with this class. NOPROFINT is the default.

RESOURCE(*class*)
Specifies the explicit eight-character resource class from the CLASS keyword on the RACROUTE macro. RESOURCE can also define the resource class defined to eTrust CA-ACF2 by CAISSF. Normal eTrust CA-ACF2 resource name masking conventions apply.

RSRCTYPE(*typecode*)
Specifies the explicit three-character resource type code associated with the class. If you define a RESOURCE but do not define a RSRCTYPE, eTrust CA-ACF2 uses the first three characters of the RESOURCE as the RSRCTYPE. Use this type code to write resource rules to perform validation. This value cannot be a mask. If you want to mask the name of the resource in your resource rule key, add this type code to the GSO RESDIR or INFODIR record and perform a rebuild. For more details, see the chapter entitled, "Maintaining Resource Rules."

Creating Multiple GSO CLASMAP Records

If you need more than one CLASMAP record, append a qualifier to the record name in the format CLASMAP*qual* to generate a unique record ID (for example, CLASMAPVMAN or CLASMAP.DATASET). This optional qualifier can be up to nine characters, and must immediately follow the characters CLASMAP. If you use a period (.) as part of the qualifier string for the record name, eTrust CA-ACF2 counts it as one of the nine characters.

Using CLASMAP Records to Validate SAF RACROUTE Calls

SAF RACROUTE calls for the DATASET, TAPEVOL, and DASDVOL classes are not translated into eTrust CA-ACF2 resource validations. For these 3 classes, a SAF call for REQUEST=AUTH is translated into a dataset validation. No CLASMAP is required. A SAF call for REQUEST=FASTAUTH or REQUEST=LIST is ignored.

SAF RACROUTE calls for any other class are translated into eTrust CA-ACF2 resource validations and a CLASMAP is recommended. For REQUEST=FASTAUTH the resource validation is done without the issuance of a SVC call. Type codes must be placed in a resource rule dierectory and the rules must be made resident for FASTAUTH calls to process correctly. For REQUEST=LIST a directory build is initiated. The type codes for these resources must be defined in a GSO RESDIR or INFODIR record.

Displaying GSO CLASMAP Definitions

The SHOW CLASMAP subcommand of the ACF command displays the internal definitions (eTrust CA-ACF2-defined) and external definitions (site-defined) of SAF calls that are being translated in one merged table. There are no GSO CLASMPA records for the internal CLASMAPs. Note in the following example that an external CLASMAP will appear in the output before an internal CLASMAP with the same resource class. Assuming the MUSASS ID is the same, the first occurance of a resource class is the one that will be used.

```
show clasmap

-- MERGED CLASMAP DEFINITIONS --

 MUSASS     RESOURCE    TYPE    ENTITY
   ID         CLASS     CODE    LENGTH     PROFINT     LOG   MIXED    EXTERNAL
 =======    ========    ===    ======     =======     ===   =====    ========
 ********   AC#CMD       SAF       8
 ********   ACAPPL       ACA      39
 ********   ACCBPROC     ACC      39
 ********   ACCTNUM      SAF      39
 ********   ACDIALOG     ACD      39
 ********   ACICSPCT     SAF      13
 ********   ACLIST       ACL      39
 ********   ACMSG        ACM      39
 ********   ACPANEL      ACP      39
```

```
********  ACREPORT     ACR     39
********  ACSQL        ACS     39
********  AIMS         SAF     8
********  AMARY        MAR     39                    LOG  MIX     EXT
********  APPCLU       ALU     35      PROF
********  APPCPORT     SAF     8
********  APPCSERV     SAF     73
********  APPCSI       SAF     26
***
```

Certificate Name Filtering Criteria Mapping (CRITMAP)

Digital certificates can be mapped to one of a number of eTrust CA-ACF2 logonids based on the system ID, application ID, or application-defined variables specified in the CRITMAP record. These records are used with the CRITERIA parameter of the CERTMAP GSO records.

Record ID	Fields
CRITMAP.*recid*	APPLID(*application-name*) APPLVAR(*site-variable-list*) SYSTEMID(*sysid*) USERID(*userid-to-map-to*)

Field Descriptions

recid
Specifies the unique one to eight-character name of the record ID.

APPLID(*application-name*)
Specifies an application ID specified by the application.

APPLVAR(*site-variable-list*)
Specifies a list of application-defined variables. For example, (BOBSAPPL=BANKAPP.LEVEL=HIGH). The application-defined variables are passed to eTrust CA-ACF2 on the initACEE callable service along with the certificate identifying the user. APPLVAR has a maximum length of 254.

SYSTEMID(*sysid*)
Specifies a system ID. The system ID is obtained from the 4 character SID parameter of the SMFPRMxx member of SYS1.PARMLIB.

USERID(*userid-to-map-to*)
Specifies the eTrust CA-ACF2 logonid assigned when matching this CRITMAP record

Example 1

We want all users whose certificate contains an issuer's distinguished name ending in L=Internet to have their logonid selected based on the application that they are accessing. Users accessing the WEBBANK application will be assigned the WEBBANK logonid. Users accessing the WEBINS application will be assigned the INSUREU logonid.

```
Insert CERTMAP.APPLS IDNFILTR(L=Internet) CRITERIA(APPLID=&APPLID) TRUST

Insert CRITMAP.BANK APPLID(WEBBANK) USERID(WEBBANK)
Insert CRITMAP.INS  APPLID(WEBINS)  USERID(INSUREU)
```

Example 2

We want all users whose certificate contains an issuer's distinguished name ending in L=Internet to have their logonid selected based on the current SYSID and application variable called LEVEL. If the application specified that LEVEL=HIGH and it is running on system A, we'll assign logonid TOPDOG. If the application specified that LEVEL=LOW on system A, we'll assign logonid WORKER.

If the request is running on system B, all users should be assigned the logonid named GENERAL.

```
Insert CERTMAP.APPL2 IDNFILTR(L=Internet) CRITERIA(SYSID=&SYSID.LEVEL=&LEVEL)
TRUST

Insert CRITMAP.SYSA1 SYSID(SYSA) APPLVAR(LEVEL=HIGH) USER(TOPDOG)
Insert CRITMAP.SYSA2 SYSID(SYSA) APPLVAR(LEVEL=LOW) USER(WORKER)
Insert CRITMAP.SYSB  SYSID(SYSB) APPLVAR(LEVEL=*) USER(GENERAL)
```

Delegated Resources (DELRSRC)

The DELRSRC record defines one or more resources that are delegated in the system. Any resource class that is available for SAF FASTAUTH processing can have resources which are designated as delegated. However, security administrators should only create delegated resources if an application explicitly requires it. eTrust CA-ACF2 resource validation is affected based on the resources you define in this record.

WARNING! Improper creation of delegated resources could adversely affect some applications.

Delegated resources you define in the system can be displayed with the SHOW DELRSRC command. You can create multiple DELRSRC records.

Record ID	Fields
DELRSRC*qual*	RDESC(R | D) RNAME(*resource-name* | *resource-mask*) RSYS(*DB2-sysid* | *DB2-sysid-mask*) RTYPE(*resource-type*)

Field Descriptions

RDESC(R | D)
Specifies a one-byte character, which is D (DB2 resource) or R (generalized resource). This field cannot be masked. The default is R.

RNAME(*resource-name* | *resource-mask*)
Specifies the 1- to 256-byte name of the delegated generalized resource or the 1- to 252-byte name of the delegated DB2 resource. This field may be masked. Use of the dash (-) character as a mask is limited to the last character you specify on the RNAME value. Only a trailing dash is treated as a mask, and will mask out to the full length of the RNAME field. Any embedded dash is not considered to be a mask character. RNAME is a required field.

RSYS(*DB2-sysid* | *DB2-sysid-mask*)
Specifies the four-byte DB2 resource sysid. This field must be specified if the RDESC field value is D. This field can be masked.

RTYPE(*resource-type*)
Specifies the three-character type code of the resource. This field is required and cannot be masked. The three-character resource type is used to uppercase the resource name at insert/change time when its associated 8-character resource class in the CLASMAP table is not mixed-case. **Note:** Do not use an asterisk (*) as a character in the RTYPE field. For example:

```
RTYPE(**C)
```

or

```
$KEY(aaa) RTYPE(**C)
```

eTrust CA-ACF2 reads asterisks in the RTYPE field as the asterisk character instead of a mask when interpreting resource rules.

Creating a GSO DELRSRC Record

To create a GSO DELRSRC record, you must have the SECURITY privilege in your logonid.

The following command, which inserts a GSO DELRSRC record, designates as delegated a resource in the FACILITY class called RESOURCE.NAME:

```
 ACF
set control(gso)
 CONTROL
insert delrsrc.fac01 rdesc(r) rtype(fac) rname(resource.name)
```

The following command, which inserts a GSO DELRSRC record, designates as delegated a DB2 resource in the DB2TABLE class on system SYS1 called RESOURCE.TABLE:

```
insert delrsrc.db201 rdesc(d) rtype(tbl) rsys(sys1) rname(resource.table)
```

Creating Multiple GSO DELRSRC Records

If you need more than one DELRSRC record, add a qualifier to the record name in the format DELRSRC*qual* to generate a unique record ID (for example, DELRSRCAPPL1 or DELRSRC.APPL1). This optional qualifier can be up to nine characters, and must immediately follow the characters DELRSRC. **Note:** If you use a period (.) as part of the qualifier string for the record name, eTrust CA-ACF2 counts it as one of the nine characters.

Activating GSO DELRSRC Records

To activate the GSO DELRSRC records, issue the following command:

```
f acf2,refresh(delrsrc),class(c),type(gso)
```

Displaying Delegated Resources (GSO DELRSRC records)

The SHOW DELRSRC, SHOW ACF2, and SHOW ALL subcommands of the ACF command display the eTrust CA-ACF2 delegated resources defined to the system in GSO DELRSRC records. The following is an example of the SHOW DELRSRC subcommand and its output:

```
show delrsrc
-- DELEGATED RESOURCES –

  Type   Sysid  Resource
  -----  -----  ------------------------------------------------------------
  D-TBL  BBMS   USER99.RACROUTE-
  R-FAC         USER99.RACROUTE.TESTDATA
```

EIM GSO Records (EIM)

The eTrust CA-ACF2 EIM defaults record provides support for IBM Enterprise Identity Mapping (EIM). LDAP bind information is extracted from eTrust CA-ACF2 for connection to an LDAP server. The EIM GSO record record contains default bind information for all EIM applications.

Record ID	Fields
EIM	BINDDN(*binddn*) BINDPTOD(00/00/00-00:00) BINDPW(*bindpw*) DOMAINDN(*domaindn*) ENABLE | DISABLE KERBREG(*registry name*) LDAPHOST(*ldaphost*) LOCALREG(*localregistry*) X509REG(*registry name*)

Field Descriptions

BINDDN
The distinguished name to use when authenticating to the LDAP server. The field can be up to 1023 characters in length.

BINDPTOD(00/00/00-00:00)
Specifies the date and time the BINDPW field was last changed. eTrust CA-ACF2 lists the date in the format mm/dd/yy, dd/mm/yy, or yy/mm/dd, depending on the DATE field of the GSO OPTS record. You cannot set this field. eTrust CA-ACF2 maintains and displays it. (Eight-byte binary field)

BINDPW
The password to use when authenticating to the LDAP server. The field can be up to 128 characters in length.

DOMAINDN
The distinguished name of an EIM domain. The field can be up to 1023 characters in length.

ENABLE | DISABLE
Specifies whether or not new connections may be established with the specified EIM domain. The default is ENABLE.

KERBREG(*registry name*)
A one to 255-character name identifying the KERBEROS registry associated with this EIM or PROXY definition. Any value entered will be changed to upper cased. To eliminate a previously specified KERBREG value on a CHANGE statement, specify KERBREG().

LDAPHOST(*ldapurl*)

Specifies the URL of the LDAP server that the z/OS LDAP Server will contact when acting as a proxy on behalf of a requester. An LDAP URL has a format such as ldap://123.45.6:389 or ldaps://123.45.6:636, where ldaps indicates that an SSL connection is desired for a higher level of security. LDAP will also allow you to specify the host name portion of the URL using either the text form (LDAP.HOST.CA.COM) or the dotted decimal address (111.222.33.44). The port number is appended to the host name, separated by a colon ':'. See your LDAP server documentation for information on how to set up the LDAP server for SSL connections.

This value must be at least 10 bytes long and can be up to 1023 bytes long. It must start with either ldap:// or ldaps://. Any characters may be entered in the remaining portion of the URL, but you should ensure that the URL conforms to TCP/IP conventions. Normally, characters such as commas, blanks, parenthesis, semicolons and single quotes are not allowed in a host name. These characters will be accepted if the LDAPHOST is enclosed in single quotes.

eTrust CA-ACF2 does not validate that the contents of the URL are valid.

LOCALREG

The name of the local registry. The field can be up to 255 characters in length.

X509REG(*registry name*)

A one to 255-character name identifying the X509 registry associated with this EIM or PROXY definition. Any value entered will be changed to upper case. To eliminate a previously specified X509REG value on a CHANGE statement, specify X509REG().

eTrust CA-ACF2 Event Filter Control for eTrust Audit (ETAUDIT)

The GSO ETAUDIT record filters the eTrust CA-ACF2 security event notifications so that only the selected security events are communicated to eTrust Audit (a security event monitoring product). There are 31 security event types that can be communicated to eTrust Audit. When all the events are selected, the volume of security event notifications may be significant and system performance may be impacted.

Note: This record will not be activated unless the GSO OPTS record field, ETAUDIT, is specified. See GSO OPTS record field descriptions later in this chapter for more information. The following list describes the ETAUDIT record fields:

Record ID	Fields
ETAUDIT	**Security System Events**: START | NOSTART STOP | NOSTOP MODIFY | NOMODIFY **System Access Events**: SIGNON | NOSIGNON SIGNOFF | NOSIGNOFF SIGNVIO | NOSIGNVIO **Object Access Events**: DATALOG | NODATALOG DATAVIO | NODATAVIO RSRCLOG | NORSRCLOG RSRCVIO | NORSRCVIO SECLVIO | NOSECLVIO **Administrative Events**: ADMINFO | NOADMINFO ADMINFOV | NOADMINFOV ADMLID | NOADMLID ADMLIDV | NOADMLIDV ADMRULE | NOADMRULE ADMRULEV | NOADMRULEV ADMRSRC | NOADMRSRC ADMRSRCV | NOADMRSRCV **z/OS Unix System Services Events**: ACCESS | NOACCESS AUDIT | NOAUDIT CHAUDIT | NOCHAUDIT CHMOD | NOCHMOD CHOWN | NOCHOWN DELUSP | NODELUSP INITACEE | NOINITACEE INITUSP | NOINITUSP SETEGID | NOSETEGID SETEUID | NOSETEUID SETFACL | NOSETFACL SETGID | NOSETGID SETUID}NOSETUID

Field Descriptions

Security System Events:

START | NOSTART
Specifies the start of the eTrust CA-ACF2 security product (S ACF2). The default is NOSTART.

STOP | NOSTOP
Specifies the stopping of the eTrust CA-ACF2 security product (P ACF2). The default is NOSTOP.

MODIFY | NOMODIFY
Specifies any modifications to the eTrust CA-ACF2 execution environment (F ACF2). The default is NOMODIFY.

System Access Events:

SIGNON | NOSIGNON
Specifies eTrust CA-ACF2 system signons. These events occur when users successfully sign on to the eTrust CA-ACF2 security system. The default is NOSIGNON.

SIGNOFF | NOSIGNOFF
Specifies eTrust CA-ACF2 system signoffs. These events occur when users successfully sign off of the eTrust CA-ACF2 security system. The default is NOSIGNOFF.

SIGNVIO | NOSIGNVIO
Specifies eTrust CA-ACF2 system signon violations. These events occur when an attempt to sign on to the eTrust CA-ACF2 security system fails, for example, when an incorrect password is entered. The default is NOSIGNVIO.

Object Access Events:

DATALOG | NODATALOG
Specifies data set loggings. The default is NODATALOG.

DATAVIO | NODATAVIO
Specifies data set violations. The default is NODATAVIO.

RSRCLOG | NORSRCLOG
Specifies resource loggings. The default is NORSRLOG.

RSRCVIO | NORSRCVIO
Specifies resource violations. The default is NORSRCVIO.

SECLVIO | NOSECLVIO
Specifies security label (SECLABEL) violations. The default is NOSECLVIO.

Administrative Events:

ADMINFO | NOADMINFO
Specifies administrative changes to InfoStorage records, such as cross-reference, scope, control, profile, etc. The default is NOADMINFO.

ADMINFOV | NOADMINFOV
Specifies attempts to make administrative changes to InfoStorage records, such as cross-reference, scope, shift, control, profile, etc, that failed because the user was not authorized to make the change. The default is NOADMINFOV.

ADMLID | NOADMLID
Specifies administrative changes to Logonid records. The default is NOADMLID .

ADMLIDV | NOADMLIDV
Specifies attempts to make administrative changes to Logonid records that failed because the user was not authorized to make the change. The default is NOADMLIDV .

ADMRULE | NOADMRULE
Specifies administrative changes to data set access rule records. The default is NOADMRULE .

ADMRULEV | NOADMRULEV
Specifies attempts to make administrative changes to data set access rule records that failed because the user was not authorized to make the change. The default is NOADMRULEV.

ADMRSRC | NOADMRSRC
Specifies administrative changes to resource rule records. The default is NOADMRSRC.

ADMRSRCV | NOADMRSRCV
Specifies attempts to make administrative changes to resource rule records that failed because the user was not authorized to make the change. The default is NOADMRSRCV.

z/OS Unix System Services Events:

ACCESS | NOACCESS
Specifies the results of the z/OS Unix System Services callable service, ck_access, to eTrust Audit. The default is NOACCESS.

AUDIT | NOAUDIT
Specifies the results of the z/OS Unix System Services callable service, R_audit, to eTrust Audit. The default is NOAUDIT.

CHAUDIT | NOCHAUDIT
Specifies the results of the z/OS Unix System Services callable service, R_chaudit, to eTrust Audit. The default is NOCHAUDIT.

CHMOD | NOCHMOD
Specifies the results of the z/OS Unix System Services callable service, R_chmod, to eTrust Audit. The default is NOCHMOD.

CHOWN | NOCHOWN
Specifies the results of the z/OS Unix System Services callable service, R_chown, to eTrust Audit. The default is NOCHOWN.

DELUSP | NODELUSP
Specifies the results of the z/OS Unix System Services callable service, deleteUSP, to eTrust Audit. The default is NODELUSP.

INITACEE | NOINITACEE
Specifies the results of the z/OS Unix System Services callable service, initACEE, to eTrust Audit. The default is NOINITACEE.

INITUSP | NOINITUSP
Specifies the results of the z/OS Unix System Services callable service, initUSP, to eTrust Audit. The default is NOINITUSP.

SETEGID | NOSETEGID
Specifies the results of the z/OS Unix System Services callable service, R_setegid, to eTrust Audit. The default is NOSETEGID.

SETEUID | NOSETEUID
Specifies the results of the z/OS Unix System Services callable service, R_seteuid, to eTrust Audit. The default is NOSETEUID.

SETFACL | NOSETFACL
Specifies the results of the z/OS Unix System Services callable service, R_setfacl, to eTrust Audit. The default is NOSETFACL.

SETGID | NOSETGID
Specifies the results of the z/OS Unix System Services callable service, R_setgid, to eTrust Audit. The default is NOSETGID.

SETUID | NOSETUID
Specifies the results of the z/OS Unix System Services callable service, R_setuid, to eTrust Audit. The default is NOSETUID.

Example

The following is an example of the ACF command to insert an ETAUDIT record:

```
ACF
 SET CONTROL(GSO)
 INSERT ETAUDIT SIGNVIO DATAVIO RSRCVIO

 CPU1 / ETAUDIT LAST CHANGED BY TESTID1 ON 03/29/04-17:09
                NOACCESS NOADMINFO NOADMINFOV NOADMLID NOADMLIDV
                NOADMRSRC NOADMRSRCV NOADMRULE NOADMRULEV NOAUDIT
                NOCHAUDIT NOCHMOD NOCHOWN NODATALOG DATAVIO NODELUSP
                NOINITACEE NOINITUSP NOMODIFY NORSRCLOG RSRCVIO NOSETEGID
                NOSETEUID NOSETFACL NOSETGID NOSETUID NOSIGNOFF NOSIGNON
                SIGNVIO NOSTART NOSTOP
```

Event IDs

Each class of security events has been assigned an event ID. The event IDs are communicated to eTrust Audit in the Event-ID field and can be used to sort or filter security events for eTrust Audit viewing purposes.

Event	Event ID
Start of eTrust CA-ACF2	501
Stop of eTrust CA-ACF2	502
Modify of eTrust CA-ACF2	503
Sign off	512
Sign on	511
Sign on violation	ACPMFWHY (message ID)
Data set logging	522
Data set violation	521
Resource logging	532
Resource violation	531
InfoStorage administration	547
InfoStorage administration violation	548
Logonid administration	541
Logonid administration violation	542
Resource rule administration	545
Resource rule administration violation	546
Rule administration	543
Rule administration violation	544
Security label violation	551
USS ck_access	606
USS deleteUSP	602
USS initACEE	638
USS initUSP	601
USS R_audit	619
USS R_chaudit	617
USS R_chmod	616

Event	Event ID
USS R_chown	615
USS R_setegid	614
USS R_seteuid	612
USS R_setfacl	647
USS R_setgid	613
USS R_setuid	611

Displaying GSO ETAUDIT Record Information

You can display the control options that are in effect for communicating security events to eTrust Audit with the SHOW ETAUDIT and SHOW ACF2 subcommands of the ACF command.

eTrust CA-ACF2 Exit Specifications (EXITS)

The EXITS record specifies the module name for each user-written eTrust CA-ACF2 exit. See the "User Exits" chapter in the *Systems Programmer Guide* for complete information concerning each exit. For the HFSEXIT, see the Security Cookbook, "Controlling Access to the Hierarchical File System" chapter, for more information.

Note: Specifying a value of NONE as a module in any of the EXITS fields results in the value being ignored and generates message ACF7A540 at eTrust CA-ACF2 startup.

Link all exits into CAI.CAILPA. LPA modules must be reentrant. To activate the new exit, perform an IPL. The *Systems Programmer Guide* provides more information about these exits.

Note: If running a release of the operating system that supports the SETPROG LPA command, you can use this instead of an IPL if you execute the following:

1. In the case of a new exit not already specified in the GSO EXITS record:

```
SETPROG LPA,ADD,MODNAME=(abcdefg),DSNAME=cailpa
CHANGE EXITS exitname(abcdefg)
F ACF2,REFRESH(EXITS)
```

2. In the case of needing to load a new copy of the exit code in which the module name is already specified in the GSO EXITS record:

```
SETPROG LPA,ADD,MODNAME=(abcdefg),DSNAME=cailpa
F ACF2,REFRESH(EXITS)
```

   Where exitname is the ACF2 GSO EXITS record name (for example, RSCXIT1, DSNPOST) and abcdefg is the site's exit module name.

The following table lists the record ID and fields that apply for EXITS:

Record ID	Fields
EXITS	CPFEXIT(*module*) DSNGEN(*module*) DSNPOST(*module*) DYNPSWD(*module*) EXPPXIT(*module*) HFSEXIT(*module*) INFOPRE(*module*) INFOPST(*module*) LGNIXIT(*module*) LGNPARM(*module*) LGNPXIT(*module*) LGNTERM(*module*) LIDLOC(*module*) LIDMOD(*module*) LIDPOST(*module*) LIDPRE(*module*) NEWPXIT(*module*) PGMOVRD(*module*) RSCXIT1(*module*) RSCXIT2(*module*) RULEPRE(*module*) RULEPST(*module*) SEVPOST(*module*) SEVPRE(*module*) SRCXIT(*module*) STCXIT(*module*) SVCIXIT(*module*) VIOEXIT(*module*) VLDEXIT(*module*)

Field Descriptions

CPFEXIT(*module*)
CPF target override exit.

DSNGEN(*module*)
Pseudo data set name generator exit.

DSNPOST(*module*)
Data set postvalidation exit. If a DSNPOST exit is taken, then the VIOEXIT is not taken.

DYNPSWD(*module*)
Dynamic password token exit.

EXPPXIT(*module*)
Expired password exit.

HFSEXIT(*module*)
Hierarchical File System (HFS) security exit.

INFOPRE(*module*)
Infostorage database preprocessing exit.

INFOPST(*module*)
Infostorage database postprocessing exit.

LGNIXIT(*module*)
TSO logon prevalidation exit.

LGNPARM(*module*)
TSO logon parameters exit.

LGNPXIT(*module*)
TSO logon postvalidation exit.

LGNTERM(*module*)
TSO terminal identification exit.

LIDLOC(*module*)
Logonid search sequence modification exit.

LIDMOD(*module*)
Logonid record Modification exit.

LIDPOST(*module*)
Logonid database postprocessing exit.

LIDPRE(*module*)
Logonid database preprocessing exit.

NEWPXIT(*module*)
New password exit.

PGMOVRD(*module*)
Program override exit.

RSCXIT1(*module*)
Resource prevalidation exit.

RSCXIT2(*module*)
Resource postvalidation exit.

RULEPRE(*module*)
Rule database preprocessing exit.

RULEPST(*module*)
Rule database postprocessing exit.

SEVPOST(*module*)
System entry validation postprocessing exit.

SEVPRE(*module*)
System entry validation preprocessing exit.

SRCXIT(*module*)
Source name modification exit.

STCXIT(*module*)
Started task validation exit.

SVCIXIT(*module*)
eTrust CA-ACF2 supervisor call initialization exit.

VIOEXIT(*module*)
Data set and program violation exit. This exit is not taken if a DSNPOST exit is specified.

VLDEXIT(*module*)
Data set and program prevalidation exit.

Displaying GSO EXIT Record Information

You can display all exits, active or inactive, with the SHOW ACTIVE and SHOW ACF2 subcommands of the ACF command.

Infostorage Rule Directories (INFODIR)

The INFODIR record indicates which infostorage rule directories are built and made globally resident at eTrust CA-ACF2 initialization. The INFODIR record also indicates whether the rule sets associated with those directories are made resident and whether the residency is global (in common storage) or local (in an address space). The INFODIR record is similar in function to the RESDIR record. While the RESDIR record enables you to specify a code and a type, the INFODIR record lets you specify a code, class, and type. This capability lets you use different class codes for your applications. We recommend that you migrate RESDIR records to INFODIR records.

Note: Resident directories can also be created by a RACROUTE REQUEST=LIST and either GLOBAL=YES or SUBPOOL=(241). When this is done, eTrust CA-ACF2 will read the approprate rules into storage. This directory cannot be delted and remains until the next IPL.

Record ID	Fields
INFODIR	TYPES(*code-classtype1,...,code-classtype256*)

Field Descriptions

TYPES(*code-classtype1,...,code-classtype256*)
Code indicates whether the rule set associated with the directory is made resident, and whether that residency is global or local. The three possible codes are:

- **T**—rule sets are transient and never are made resident.
- **D**—rule sets are made resident locally in the user's address space when access to a resource is attempted rather than being made resident in global storage at eTrust CA-ACF2 initialization time. Locally resident rules are not affected by the F ACF2,REBUILD command. To clear locally resident rules, use the ACF2 SETNORUL command or the address space must come down and back up (or the user must sign off and sign on again.) The rules are reloaded when access to the resource is attempted.
- **R**—rule sets are made resident in global storage at eTrust CA-ACF2 initialization time.

We recommend specifying **R** for the code; specifying **T** or **D** could affect performance since additional I/O to the infostorage database is required for transient or on-demand processing.

Class specifies the infostorage class of the rule set for which the directory is built. For eTrust CA-ACF2 for DB2 rule sets, the storage class is **D**; for resource rule sets, **R**; for record-level protection records, **F**; for profile records, **P**.

Type specifies the pertinent rule set for which a directory is built and made globally resident at eTrust CA-ACF2 initialization. The *type* is the same three-character type code used in the $TYPE control statement of the rule set.

For example, to create an INFODIR record for DB2 resources, use the following:

```
TYPES(R-DSYS,D-DDBS,T-DBPL)
```

Where:

R-DSYS specifies that a rule directory is built and loaded into common storage for eTrust CA-ACF2 for DB2 rules of $TYPE(SYS), and the rules themselves are made globally resident.

D-DDBS specifies that a rule directory is built and loaded into common storage for eTrust CA-ACF2 for DB2 rules of $TYPE(DBS) and the rules themselves are made resident in an address space upon request.

T-DBPL specifies that a rule directory is built and loaded into common storage for eTrust CA-ACF2 for DB2 rules of $TYPE(BPL), but the rules themselves are never made resident in common storage nor in an address space, but are loaded and unloaded into an address space on a transient basis.

Regardless of whether a given type of rule set is resident or transient, a directory is built for each type of rule set specified in the INFODIR record. The directory itself is always made globally resident.

Resident directories are rebuilt at each IPL, at each restart of eTrust CA-ACF2, or when the console operator enters the following command:

```
F ACF2,REBUILD(type),CLASS(class)
```

To use a mask (asterisks) in the $KEY or $SYSID control statement of a resourc rule, you must make the directory resident for the given type of rule set.

Storage utilization is improved when common rules are made globally resident. For example, when 100 users access the same resource, 100 copies of that rule are in LSQA unless the rule is made resident.

Displaying Resources Specified in the GSO INFODIR Record

The SHOW RESIDENT and SHOW ACF2 subcommands of the ACF command display the types of resources as specified in the INFODIR record.

Logical Extension of the System Link List (LINKLST)

The LINKLST record defines one or more program libraries that eTrust CA-ACF2 considers part of the system link list (SYS1.LINKLIB). Many sites use program pathing controls (for example, PROGRAM and LIB access rule options) to define which programs are allowed access to a data set. LINKLST provides sites with added flexibility for the eTrust CA-ACF2 program pathing facility.

A LINKLST record is required. If one is not specified, eTrust CA-ACF2 will define a default record. Ensure all specified libraries are catalogued before you start eTrust CA-ACF2.

Record ID	Fields
LINKLST*qual*	LIBRARY(SYS1.LINKLIB | *libraryname1,...,libraryname64*)

Field Descriptions

LIBRARY(SYS1.LINKLIB | *libraryname1,...,libraryname64*)
Specifies up to 64 fully qualified partitioned data set (library) names. Each library name can be up to 44 characters long. The default is LIBRARY(SYS1.LINKLIB).

LINKLST Processing

You cannot specify more than 64 program library names in an individual LINKLST record, but you can use multiple records to expand the number of program libraries that eTrust CA-ACF2 recognizes as part of the system link list to 500. You can append a qualifier to the LINKLST record name to generate a unique record ID (for example, LINKLSTABC or LINKLST.ABC). This optional qualifier can be up to nine characters, and must immediately follow the characters LINKLST. If you use a period (.) as part of the qualifier string for the record name, eTrust CA-ACF2 counts it as one of the nine characters.

If you specify the same program library in more than one LINKLST record, eTrust CA-ACF2 processes it, but issues a warning message.

eTrust CA-ACF2 regards any library specified in the LINKLST record as part of SYS1.LINKLIB. If you specified PROD.LOADLIB in the LINKLST record, eTrust CA-ACF2 would validate using the name SYS1.LINKLIB. This means that if you specify a library in the LINKLST record and in the MAINT record, eTrust CA-ACF2 would use the value in the LINKLST record. Returning to our example, if PROD.LOADLIB is specified in both the LINKLST and MAINT records, eTrust CA-ACF2 would validate access by MAINT programs using SYS1.LINKLIB, deny access, and create a violation record. To remedy the violation, if PROD.LOADLIB is specified in the LINKLST record, then SYS1.LINKLIB should be specified in the MAINT record.

Displaying GSO LINKLST Record Information

The SHOW LINKLST and SHOW ACF2 subcommands of the ACF command display all LINKLST option specifications.

Linux Machine Definitions (LINUX)

The LINUX record defines LINUX machines to eTrust CA-ACF2.

Record ID	Fields
LINUX*qual*	ACTIVE | NOACTIVE IPADDR (*IP Address*) MACHNAME *(Linux machine name*)

Field Descriptions

ACTIVE | NOACTIVE
Specifies whether the Linux machine is active or not. ACTIVE is the default.

IPADDR(*IP Address*)
The IP address of the Linux machine. This is a required field.

MACHNAME(*Linux machine name)*
Specifies the LINUX machine name. This field can be from 1-246 characters long and is case sensitive. Valid characters are A-Z, a-z, 0-9, - (dash) and . (period). This is a required field.

Displaying LINUX Machine Definitions

The SHOW LINUX subcommand of the ACF command displays the LINUX machine definitions currently in use.

Data Set Access Logging Options (LOGPGM)

The LOGPGM record defines a set of programs that your site wants to protect. eTrust CA-ACF2 creates an SMF logging record each time a program listed on the GSO LOGPGM record is executed with a JCL EXEC PGM= statement. To specify the names of programs you want to audit without defining a GSO PPGM record, see the section at the end of this chapter that defines the relationship among the GSO LOGPGM, PPGM, and MAINT records.

eTrust CA-ACF2 provides a default LOGPGM record. You can change or add names to the list.

Record ID	Fields
LOGPGM	PGMS(AMASPZAP,IMASPZAP|*pgm1,...,pgm256*)

Field Descriptions

PGMS(AMASPZAP,IMASPZAP|*pgm1,...,pgm256*)
Specifies up to 256 program names. Programs AMASPZAP and IMASPZAP are the defaults.

Displaying Programs Defined in the GSO LOGPGM Record

The SHOW PROGRAMS (or SHOW PGMS) and the SHOW ACF2 subcommands of the ACF command display all program names defined in the LOGPGM record.

System Maintenance Options (MAINT)

The MAINT record specifies the program, library, and logonid that make up a maintenance environment. Disk compression and archiving are examples of standard system maintenance functions. These maintenance programs must reside in the specific library and be executed on behalf of the specified logonid.

When eTrust CA-ACF2 encounters this environment, it:

- Bypasses rule validation only if the library, LID, and program match the MAINT record
- Creates **no SMF logging records**
- Does not queue data set access rule sets in the address space

Any logonid you specify in the MAINT record must have the NON-CNCL or MAINT privilege. If the logonid does not have the MAINT or NON-CNCL privilege, eTrust CA-ACF2 does not examine the maintenance program list. See the section at the end of this chapter that describes the relationship among the GSO MAINT, LOGPGM, and PPGM records.

Record ID	Fields
MAINT*qual*	LIBRARY(*library*) LID(*logonid*) PGM(*pgm1,...,pgm256*)

Field Descriptions

LIBRARY(*library*)
Defines a fully qualified library data set name in which the maintenance programs reside. eTrust CA-ACF2 checks all active libraries when matching the MAINT record as it does for program pathing. Define only one library per MAINT record.

However, if the library is also specified in the GSO LINKLST record, the library name that you specify in the MAINT record should be SYS1.LINKLIB. For example, if PROD.LOADLIB is specified in the LINKLST record, then SYS1.LINKLIB should be specified in the MAINT record. Based on the entry in the LINKLST record, eTrust CA-ACF2 validates accesses from PROD.LOADLIB as coming from SYS1.LINKLIB.

LID(*logonid*)
Specifies the logonid of an authorized maintenance user. Define only one logonid per MAINT record.

PGM(*pgm1,...,pgm256*)
Individual MAINT records are limited to 256 programs per MAINT record; however, the maximum number of programs you can specify in the cumulative total of MAINT records that exist is 1024. Any programs you create over the 1024 maximum are ignored by eTrust CA-ACF2.

Creating Multiple GSO MAINT Records

If you need more than one MAINT record, append a qualifier to the record name to generate a unique record ID (for example, MAINTABC or MAINT.ABC). This optional qualifier can be up to 11 characters, and must immediately follow the characters MAINT. If you use a period (.) as part of the qualifier string for the record name, eTrust CA-ACF2 counts it as one of the 11 characters.

Displaying Programs Defined in the GSO MAINT Record

The SHOW PROGRAMS (or SHOW PGMS) and SHOW ACF2 subcommands of the ACF command display all program names defined by the MAINT record.

Mini-Logonid Record (MLID)

The MLID record specifies a logonid compression algorithm used in the MUSASS (Multiple-User, Single Address Space System) environment to reduce virtual storage requirements. Logonid compression eliminates unused or unnecessary information from the resident portion of the logonid record.

When MLID GSO records are REFRESHed, MUSASSes affected by changes to the MLID record must be taken down if the MUSASSes are to take advantage of the changes. Changes take effect the next time the affected MUSASSes are restarted.

Note: Be aware that GSO MLID records can only be used in conjunction with GSO MUSASS records, and not with @MUSASS macros in the ACFFDR. GSO MUSASS records can be used in conjunction with either @MLID macros in the ACFFDR or GSO MLID records.

When built, the mini-logonid includes two sections; the MUSASS-dependent fields and the eTrust CA-ACF2 required fields. The MUSASS mini-logonid does not need to include those fields named in the eTrust CA-ACF2 @MLID definition because they are automatically included in the mini-logonid.

See the Expand or Contract a Mini-Logonid Macro (ACELID) in the "Interfacing with eTrust CA-ACF2" and the "MUSASS Environments" chapters of the *Systems Programmer Guide* for more information about MUSASS and mini-logonid concepts.

Record ID	Fields
MLID.*qual*	LENGTH(0|length) MLIDFLDS(*lidfield(offset), lidfield(offset) ,_, lidfield(offset)*)

Field Descriptions

MLID.qual

Specifies a unique name for this MLID record. More importantly, the qualifier (.qual) must match the name of the MLID definition, as specified in the GSO MUSASS MLID field. A period (.) must be the first character of the qualifier string for the MLID record. eTrust CA-ACF2 will use the characters after the period as the MLID name. The MLID name can be 1 to 8 characters long.

LENGTH(0|length)

Specifies the total length of the mini-logonid record. The LENGTH must be specified in hexidecimals, for example: LENGTH(1FC) is valid. If LENGTH is not specified or LENGTH(0) is specified, the length of the mini-logonid section will default to the highest offset specified in the MLIDFLDS operand, plus the length of the last lidfield. If the LENGTH is smaller than the highest offset plus the length of the corresponding lidfield, then the LENGTH field will be ignored. The LENGTH cannot exceed 400 hexidecimal, which is the length of the full logonid. Valid values are 0 to 400 hexidecimal. The default is 0.

MLIDFLDS(*lidfield(offset), lidfield(offset) ,_, lidfield(offset)*)

A list of field names (lidfield) from the full Logonid with their corresponding offset into the mini-lid mapping. Offsets are specified in hexidecimal. All logonid field names referenced in the MLID record must have a corresponding @CFDE macro specified in the ACFCFDE macro or the USERCFDE macro. The lidfield(offset) pairs must be separated by a blank space or a comma. Up to 256 lidfield(offset) pairs can be specified.

For the purposes of the MLID record, logonid fields are divided into two basic types: those fields with values and those fields that are designated as bit fields. When specifying a value field, indicate the external name of the logonid field and its starting offset in the MLID's mapping DSECT. When specifying a bit field, refer to the external name of any bit field defined in the byte and the offset of the bit's byte in the MLIDs mapping DSECT. Any one of the eight bit fields in any byte can be used to identify the byte making all eight of the bit fields accessible via the same MLID record.

Example MLID Record

The following example shows the steps involved in an @MLID macro to a GSO MLID record. The default CICS MLID macro shipped with eTrust CA-ACF2 is used in this example.

The CICS @MLID macro appears as follows:

```
@MLID CICS,MLACICS,MLACICSL,      NAME, START, LENGTH
          (LIDCOPCL,MLACOPCL),      OPERATOR CLASS
          (LIDCOPID,MLACOPID),      OPERATOR ID
          (LIDCOPKY,MLACOPKY),      OPERATOR SECURITY KEY
          (LIDCOPRL,MLACOPRL),      RESOURCE LEVEL KEY
          (LIDCOPPR,MLACOPPR),      OPERATOR PRIORITY
          (LIDIDLE,MLACIDLE),       MAX IDLE TIME IN MINUTES
          (LIDM2FLG,MLACAUTH),      CICS SIGNON AUTH BYTE
          (LIDCOPKX,MLACOPKX)       CICS OP SEC KEY EXTENDED MINUTES
```

The corresponding CICS MLID mapping DSECT:

```
HEXADECIMAL
OFFSET
------------
000000   MLACICS  DSECT  ,
000000   MLACOPCL DS     CL3        OPERATOR CLASS
000003   MLACOPID DS     CL3        OPERATOR ID
000006   MLACOPKY DS     XL3        OP SECURITY KEY
000009   MLACOPRL DS     XL3        RESOURCE LEVEL KEY
00000C   MLACOPPR DS     XL1        OPERATOR PRIORITY
00000D   MLACIDLE DS     XL1        MAX IDLE TIME IN MINUTES
00000E   MLACAUTH DS     XL1        CICS AUTHORIZATION BYTE
00000F   MLACOPKX DS     XL5        CICS OP SEC KEY EXTENDED
000014   MLACICSL DS     EQU        *-MLACICS LENGTH OF MILD
```

The following are the @CFDE macros for each of these fields:

```
@CFDE    CICS,LIDM2FLG,BIT,ALTER=SECURITY+ACCOUNT,LIST=ALL
               FLAGS=NULL,BITMAP=LIDM2CIC,GROUP=2
@CFDE  CICSCL,LIDCOPCL,HEX,ALTER=SECURITY+ACCOUNT,LIST=ALL,
               FLAGS=NULL,GROUP=7,TRIM=NO
@CFDE  CICSID,LIDCOPID,CHAR,LIST=ALL,ALTER=SECURITY+ACCOUNT,
               FLAGS=NULL,GROUP=7
@CFDE  CICSKEY,LIDCOPKY,HEX,ALTER=SECURITY+ACCOUNT,LIST=ALL,
               FLAGS=NULL,GROUP=7,TRIM=NO
@CFDE  CICSKEYX,LIDCOPKX,HEX,ALTER=SECURITY+ACCOUNT,LIST=ALL,
               FLAGS=NULL,GROUP=7,TRIM=NO
@CFDE  CICSPRI,LIDCOPPR,BINARY,ALTER=SECURITY+ACCOUNT,
               LIST=ALL,FLAGS=NULL,GROUP=7,VRTN2=14
@CFDE  CICSRSL,LIDCOPRL,HEX,ALTER=SECURITY+ACCOUNT,LIST=ALL,
               FLAGS=NULL,GROUP=7,TRIM=NO
@CFDE  IDLE,LIDIDLE,BINARY,LIST=ALL,ALTER=SECURITY+ACCOUNT,
               FLAGS=NULL+LIMIT,GROUP=7,VRTN2=14
```

Based on the this, the following is the required insert command to create a MLID record:

```
INSERT MLID.CICS LENGTH(14)
MLIDFLDS(CICSCL(0),CICSID(3),CICSKEY(6),CICSRSL(9),CICSPRI(C),IDLE(D),CICS(E),
CICSKEYX(F))
```

This command was created by using the following procedure:

- The qualifier for the record name uses the name of the @MLID macro. In this case, it is CICS so the record name is MLID.CICS.
- The LENGTH field is the total length of the combined logonid fields used in the MLID. In this case, the ending length equate entry, MLACICL, from the MLID DSECT indicates the hexadecimal notation is 14, so that value is placed in the LENGTH parameter.
- Each field used in the MLID record must be defined in the MLIDFLDS parameter as a field name and its offset in the MLID DSECT. The CICS @MLID macro specifies the fields in pairs. The first value is the internal logonid field name as specified in the lidrec DSECT and referenced in the @CFDE macro for the field. The second value is the MLID DSECT name for the field. Take each internal logonid field name and locate it's external field name in its @CFDE macro. For example using the first field in the CICS MLID mapping DSECT, LIDCOPCL, you would find this value in one of the @CFDE macros. In this case it is the second @CFDE macro in the list. Note it's external name of CICSCL. Next take the MLID DSECT name of MLACOPCL and locate it's position in the MLID DSECT, which in this case is 0. Specify the entry of CICSCL(0) in the MLIDFLDS. Continue with each field until all have been identified as shown in the example command. All of the fields used in this example are value fields with the exception of the LIDM2FLG,MLACAUTH pair.

  In a case such as this, select one of the bit field names (in this example CICS) and use it to reference the byte. As previously noted, even though a bit field name is used, the entire byte is actually referenced so all bit fields specified in that byte are available. This means that this one MLID can be used to specify any of the bit fields defined in the LIDM2FLG byte.

Displaying GSO MLID Information

SHOW MLID or SHOW ACF2 subcommands display MLID record information.

Multilevel Security Options (MLSOPTS)

The MLSOPTS record lets you set values that define system options for the MLS environment. For more information on how to implement MLS on a system, including setting system options and defining, assigning and using security classifications, see the *Multilevel Security Planning Guide.*

Record ID	Fields
MLSOPTS	MLACTIVE | NOMLACTIVE MLFSOBJ | NOMLFSOBJ MLIPCOBJ | NOMLIPCOBJ MLSECBYS | NOMLSECBYS MLWRITE | NOMLWRITE MODE(*mode* | QUIET)

Field Descriptions

MLACTIVE | NOMLACTIVE
Specifies whether MLS is active on the system. The default is NOMLACTIVE, MLS is not active.

MLFSOBJ | NOMLFSOBJ
Specifies whether security labels are required for UNIX directories and files. The default is NOMLFSOBJ, security labels are not required for UNIX directories and files.

When MLS is active and MLFSOBJ is set, the following will occur:

- If a UNIX file or directory does not have a security label, the requested access will fail, regardless of the MLS mode that has been established.
- If a UNIX file or directory has a security label, MAC label dominance checking will be performed to allow or deny the requested access, based on the MLS mode that has been established.

When MLS is active and NOMLFSOBJ is set, the following will occur:

- If a UNIX file or directory has the security label, SYSMULTI, the requested access will be allowed (if DAC validation also allows it), and MAC label dominance checking will be bypassed (since SYSMULTI is equivalent to any other security label).
- If a UNIX file or directory has a security label other than SYSMULTI, MAC label dominance checking will be performed to allow or deny the requested access, based on the MLS mode that has been established.

MLIPCOBJ | NOMLIPCOBJ

Specifies whether security labels are required for UNIX IPC objects. The default is NOMLIPCOBJ, security labels are not required for UNIX IPC objects.

When MLS is active and MLIPCOBJ is set, the following will occur:

- If a UNIX IPC object does not have a security label, the requested access will fail, regardless of the MLS mode that has been established.
- If a UNIX IPC object has a security label, MAC label dominance checking will be performed to allow or deny the requested access, based on the MLS mode that has been established.

When MLS is active and NOMLIPCOBJ is set, the following will occur:

- If a UNIX IPC object has the security label, SYSMULTI, the requested access will be allowed (if DAC validation also allows it), and MAC label dominance checking will be bypassed (since SYSMULTI is equivalent to any other security label).
- If a UNIX IPC object has a security label other than SYSMULTI, MAC label dominance checking will be performed to allow or deny the requested access, based on the MLS mode that has been established.

MLSECBYS | NOMLSECBYS

Specifies whether security labels can be restricted for use on specific systems as specified in the SYSTEMID field of the SECLABEL Profile Data Record. The default is NOMLSECBYS, defined security labels are not restricted for use on specific systems.

Note: The system-defined security labels, SYSLOW, SYSHIGH, SYSMULTI, and SYSNONE, are always available on all systems regardless of whether this option is on or off.

Important! After activating MLSECBYS, you must rebuild the security classifications table with the command 'F ACF2,MLS'.

MLWRITE|NOMLWRITE

Specifies whether writing data from a higher security label to a lower security label is allowed or prohibited in the system. The default is MLWRITE, write-down is allowed. When NOMLWRITE is set, write-down is prevented and new data is labeled automatically and internally by the system at the time it is created with the session security label of the user who first created the data. In addition, when NOMLWRITE is set, there may be some situations where a user may need to write-down. In these cases, a security administrator can authorize a user for "controlled write-down". Controlled write-down lets a user enter the system with the ability to write-down by default (UPDATE access to the IRR.WRITEDOWN.BYUSER resource in the FACILITY class) or issue the ACF MLWRITE subcommand to set, reset or query write-down privilege (READ access to the IRR.WRITEDOWN.BYUSER resource in the FACILITY class). Controlled write-down is also available in a UNIX System Services environment through the UNIX **writedown** command.

WARNING! *Before enabling NOMLWRITE, it is recommended that you successfully test the system in MLS LOG and WARN modes before implementing MLS mode. There may be some system performance degradation when NOMLWRITE is set.*

For more information about write-down and controlled write-down, including the MLWRITE and UNIX **writedown** commands, see the *Multilevel Security Planning Guide.*

MODE(QUIET|LOG|WARN|MLS)

Defines the mode for MLS operations. The mode determines what actions to take when a request to access a data set or resource is considered a violation according to MLS validation rules. Valid modes are:

QUIET

Validates security labels only at system entry. SMF logging is done for violations. This is the default.

LOG

Logs and permits MLS accesses to classified data sets and resources, including UNIX files, directories, and IPC objects that normally would violate MLS validation rules.

WARN

Logs and permits MLS accesses to classified data sets and resources, including UNIX files, directories, and IPC objects, that normally would violate MLS validation rules and sends a warning message to the user.

MLS

Prevents MLS accesses to classified data sets and resources, including UNIX files, directories, and IPC objects, based on MLS validation rules and logs violations.

Creating a GSO MLSOPTS Record

To create the GSO MLSOPTS record, you must have the SECURITY privilege in your logonid. Issue the following commands to create the record:

```
ACF
set control(gso)
 CONTROL
insert mlsopts
 XE41 / MLSOPTS LAST CHANGED BY M1ADMN ON 04/22/04-15:21
 NOMLACTIVE NOMLFSOBJ NOMLIPCOBJ NOMLSECBYS MLWRITE MODE(QUIET)
CONTROL
```

Viewing a GSO MLSOPTS Record

To view the GSO MLSOPTS record, you must have the SECURITY privilege in your logonid. Issue the following commands to view the record:

```
ACF
set control(gso)
 CONTROL
list mlsopts
 XE41 / MLSOPTS LAST CHANGED BY M1ADMN ON 04/22/04-15:21
 NOMLACTIVE NOMLFSOBJ NOMLIPCOBJ NOMLSECBYS MLWRITE MODE(QUIET)
CONTROL
```

Changing a GSO MLSOPTS Record

To change the GSO MLSOPTS record, you must have the SECURITY privilege in your logonid. Issue the following commands to change the record:

```
ACF
set CONTROL(GSO)
 CONTROL
change mlsopts mode(log)
 XE41 / MLSOPTS LAST CHANGED BY M1ADMN ON 04/22/04-15:22
 NOMLACTIVE NOMLFSOBJ NOMLIPCOBJ NOMLSECBYS MLWRITE MODE(LOG)
CONTROL
```

Deleting a GSO MLSOPTS Record

To delete the GSO MLSOPTS record, you must have the SECURITY privilege in your logonid. Issue the following commands to delete the record:

```
ACF
set CONTROL(GSO)
 CONTROL
delete mlsopts
 ACF6D073 XE41 / MLSOPTS RECORD DELETED
CONTROL
```

Activating a GSO MLSOPTS Record

To activate a new record or changes to it, issue the following command:

```
f acf2,refresh(mlsopts),class(c),type(gso)
```

Multiple-User, Single Address Space System (MUSASS)

The MUSASS record defines special processing to be performed by eTrust CA-ACF2 on behalf of a MUSASS (Multiple-User, Single Address Space System) to reduce eTrust CA-ACF2 storage requirements and CPU overhead. When a MUSASS starts up, eTrust CA-ACF2 will first look for a matching ID in the MUSASS GSO records, then in the ACFFDR @MUSASS macros.

Note: Be aware that GSO MLID records can only be used in conjunction with GSO MUSASS records, and not with @MUSASS macros in the ACFFDR. GSO MUSASS records can be used in conjunction with either @MLID macros in the ACFFDR or GSO MLID records.

See the Expand or Contract a Mini-Logonid Macro (ACELID) in the "Interfacing with eTrust CA-ACF2" chapter and the "MUSASS Environments" chapter of the *Systems Programmer Guide* for more information about MUSASS and mini-logonid concepts.

Record ID	Fields
MUSASS.*qual*	CACHE | NOCACHE CACHE#(5 | nnn) CVTNAME(name) CVTCOM | NOCVTCOM FASTPTH | NOFASTPTH MLID(name) MUSID(musassid | ********) WORKSP(0 | nnn) WORKLEN(0 | nnnnn)

Field Descriptions

CACHE | NOCACHE
Specifies whether a generalized resource cache should be built and maintained for each ACMCB associated with this MUSASS. The cache consists of an in-core table that contains information (resource rule key and eTrust CA-ACF2 validation information) from previous generalized resource validation calls.

The size of each cache is determined by CACHE#. Use of the cache reduces the number of eTrust CA-ACF2 generalized resource calls in many MUSASS environments. The eTrust CA-ACF2 TYPE=A SVC builds and maintains a cache for each ACMCB associated with this MUSASS. When the default, NOCACHE is specified, a cache is not built.

CACHE#(5|nnn)
Specifies the size of the cache for each ACMCB associated with this MUSASS when CACHE is specified. For example, the number of generalized resource information entries maintained for each ACMCB. The maximum value allowed is 255. The default is 5.

CVTNAME(name)
The name of an operand module to be loaded at job initialization time and copied to the work area defined by the WORKSP and WORKLEN. This module can define local processing options for the MUSASS (such as security mode, and resource classes). This module cannot contain any address contents (ADCONS) because it is relocated during job initialization.

CVTCOM|NOCVTCOM
Required with the CVTNAME, this field specifies whether the module loaded must reside in the system common area. If CVTCOM is specified, changes to the module do not take affect until the next system IPL (with MLPA/CLPA).

FASTPTH|NOFASTPTH
Specifies whether fast pathing support will be used by the MUSASS.

If FASPTH is specified, a work area is GETMAINed for use by the MUSASS when the MUSASS is initialized. The work area (key 0 (supervisor state storage) subpool 255) is used by the eTrust CA-ACF2 TYPE=A SVC. Whenever the TYPE=A SVC is called, the MUSASS ACUCB is checked to see if a work area is available. If available and not being used, the TYPE=A SVC uses it, thus eliminating a GETMAIN/FREEMAIN pair of instructions. The work area is freed when the MUSASS is terminated.

If NOFASTPTH is specified, a work area is not GETMAINed at MUSASS initialization and the TYPE=A SVC always issues a GETMAIN/FREEMAIN when called.

MLID(name)
The ID of a mini-logonid compression algorithm to be used for this MUSASS. The mini-logonid facility conserves space by omitting unnecessary portions of the logonid record from resident storage. This name must match the qualifier (1 to 8 characters) on a MLID GSO record or the ID name on an @MLID macro in the ACFFDR. When eTrust CA-ACF2 looks for a matching MLID, it looks first in the GSO MLID records, then in the ACFFDR @MLID table. If the MUSASS requires fields in addition to the eTrust CA-ACF2 standard mini-logonid, the name provided will be the MLID definition for the MUSASS.

If only the eTrust CA-ACF2 standard mini-logonid is required, then specify, MLID(ACF2). If MLID(name) is coded but no corresponding GSO MLID record or ACFFDR @MLID macro is found, the eTrust CA-ACF2 mini-logonid will be used when the MUSASS initializes.

If no mini-logonid support is desired, then do not code this operand.

Sample INSERT command:

```
INSERT MUSASS.MYMUSASS MUSID(MYMUS-) MLID(MYMLID1) CVTNAME(MYMUSCVT)
 WORKSP(255) WORKLN(1024) CVTCOM FASTPTH CACHE CACHE#(5)
```

MUSID(musassid|******)**

The MUSASS ID field must match the logonid of a job entering the system and that logonid must be defined with the MUSASS attribute for the MUSASS specifications to take effect.

The MUSASS ID distinguishes between the various MUSASSes that might be running. When a MUSASS enters the system, the logonid it is running under is compared to each MUSASS ID. If no match is found, eTrust CA-ACF2 assumes no special processing. When a match is found, the associated MUSASS record options control eTrust CA-ACF2 processing for the life of the job (MUSASS). eTrust CA-ACF2 supplies two default @MUSASS's, named CICSCVT and IMS, in the ACFFDR. Normal eTrust CA-ACF2 masking conventions apply.

WORKSP(0|nnn)

Specifies the storage subpool of an area to be allocated by eTrust CA-ACF2 at job initialization time. This information is used with the CVTNAME and must be defined if CVTNAME is specified. The module loaded in response to the CVTNAME is copied into this storage area. The length of the storage area must be defined in the WORKLEN field. Valid values are 0 to 255. The default is 0.

WORKLEN(0|nnnnn)

Specifies the length of the storage area to be allocated by eTrust CA-ACF2 at job initialization time. If this area is longer than the module specified in CVTNAME, it is padded with zeros. If WORKLEN(0) is specified, the length defaults to the total length of the module specified in CVTNAME. Valid values are 0 to 32,760. The default is 0.

Creating Multiple GSO MUSASS Records

If you need more than one MUSASS record, append a qualifier to the record name to generate a unique record ID. For example, MUSASSPRODA or MUSASS.PRODA. This qualifier can be up to 10 characters and must immediately foolow the charactes MUSASS. If you user a period as part of the qualifier string for the record name, eTrust CA-ACF2 counts it as one of the 10 characters.

Displaying GSO MUSASS Information

SHOW ACF2 or SHOW MUSASS subcommands display MUSASS record information.

Network Job Entry Validation Options (NJE)

The NJE record tells eTrust CA-ACF2 how to process passwords and logonid inheritance for jobs exchanged with other NJE nodes.

eTrust CA-ACF2 uses the NJE record to build a table that it references whenever a job enters or leaves the system through NJE. eTrust CA-ACF2 compares the name of the remote node specified in the NODEMASK fields in the table. If eTrust CA-ACF2 does not find a match, it uses a default record generated for validation. The internal table that eTrust CA-ACF2 builds contains room for 256 NJE records. If more NJE records are found, a message is issued and the subsequent records are bypassed.

When a job enters the system from a remote NJE node, eTrust CA-ACF2 accesses the NJE record where the NODEMASK matches the sender's node name and uses the DFTLID, VALIN, and INHERIT options from that entry. When a job is sent to a remote NJE node, eTrust CA-ACF2 accesses the NJE record where the NODEMASK matches the node name of the receiver and uses the ENCRYPT and VALOUT options of the entry.

Record ID	Fields
NJE*equal*	DFTLID(*defaultlid*) DFSYSOUT(*defaultlid*) ENCRYPT | NOENCRYPT INHERIT | NOINHERIT NODEMASK(*nodename*) VALIN(YES | ONLY) VALOUT | NOVALOUT

Field Descriptions

DFTLID(*defaultlid*)

Specifies the default logonid to be used for jobs that come from the remote node when no logonid can be associated with the job. (IBM restricts the NJE default logonid to a maximum of seven characters.) The DFTLID value you specify must be a valid, specific logonid (no masking characters) defined to eTrust CA-ACF2 at the home node. Also, you must set the RESTRICT field in the logonid record to indicate that the logonid does not require password verification. The default value for DFTLID is null

If eTrust CA-ACF2 needs to assign a default logonid but no DFTLID is coded in the NJE record, eTrust CA-ACF2 substitutes the default logonid specified in the DFTLID field of the GSO OPTS record. If no default logonid is specified in the OPTS record, the job fails validation.

DFSYSOUT(*defaultlid*)

Specifies the default sysout logonid used for sysout from a remote node when no logonid can be associated with the job. (IBM restricts the NJE default sysout logonid to a maximum of seven characters.) The DFSYSOUT value you specify must be a valid, specific logonid (no masking characters) defined to eTrust CA-ACF2 at the home node. Also, you must set the RESTRICT field in the logonid record to indicate that the logonid does not require password verification. The default value for DFSYSOUT is null.

If eTrust CA-ACF2 needs to assign a default sysout logonid but no DFSYSOUT is specified in the NJE record, eTrust CA-ACF2 substitutes the DFTLID logonid from the NJE record. If no DFTLID is specified, eTrust CA-ACF2 substitutes the default logonid specified in the DFTLID field of the GSO OPTS record. If no default logonid is specified in the OPTS record, the sysout owner is assigned as ++++++++.

ENCRYPT|NOENCRYPT

ENCRYPT specifies that the password provided on jobs sent to this node for validation are in a partially encrypted format using the XDES algorithm. NOENCRYPT specifies that the password is sent in clear-text format. The default value is ENCRYPT.

Nodes that use no security system or use a security package other than eTrust CA-ACF2, must be sent passwords in clear text format.

Note: If the job is sent by an /*XMIT statement, ENCRYPT does not apply to any password specified on the second job statement. In this case, the password is sent in clear text format.

INHERIT|NOINHERIT

INHERIT specifies that the local node accepts network job inheritance. That is, a job sent to this node inherits the logonid of the user who submitted the job. NOINHERIT specifies that the job requires an explicit logonid and password to be validated at this node. In this case, the job does not inherit the logonid of the submitter and if no logonid is provided in the JCL, the job is assigned the default logonid specified at the local node. The default value is INHERIT.

NODEMASK(*nodename*)

Specifies the node or nodes with which this record is associated. *Nodename* is the value specified in the NAME= field of the NODE(x) statement of the JES initialization parameters for the system. If you are unsure of the node NAME=, issue a $TNODE(nn) command from the operator console to find out this value. For example, if you have these statements in your JES initialization member, you can create one NJE record with NODEMASK(THEM) to cover N6, THEM, and ELSEWHER.

```
N5 NAME=OURSITE2
N6 NAME=THEM
 ....
DESTID(ELSEWHER) DEST=N6
```

Nodenames defined by JES DESTID statements require no NJE record or nodemask.

Nodename can be from one to eight characters, and can be a specific node name or a masked value (standard masking rules apply). If you do not specify the field, *nodename* defaults to a mask of dash (–), which means all node names match.

For example, the following mask identifies a single node:

```
NODEMASK(CHICAGO)
```

In the next example, the following mask matches any four-character node name in which the first three characters are CHI and the last character is any character other than null:

```
NODEMASK(CHI*)
```

The following mask matches any one to eight-character node name.

```
NODEMASK(-)
```

VALIN(YES|ONLY)

YES specifies that all jobs coming in from the remote node are validated. ONLY specifies that incoming jobs that have already been eTrust CA-ACF2-validated are not revalidated. The default value is YES

VALOUT | NOVALOUT

VALOUT specifies that all outgoing jobs to the remote node are validated. NOVALOUT specifies that no validation is performed for outbound jobs. The default value is NOVALOUT.

You should review the implications of job inheritance when selecting these options. If a job is submitted without a password and the INHERIT option is active, the job runs under the submitter's logonid.

Important! *If network job inheritance is permitted, different authorization (possibly including higher privileges than intended) can be granted to the job. This could be due to the local logonid having a much higher authority than the same logonid defined at the sending node. You can control this with the logonid record NO-INH field. This field prevents a logonid from being inherited. For more information about the NO-INH field, see "Maintaining Logonid Records."*

Creating Multiple GSO NJE Records

eTrust CA-ACF2 lets you specify multiple GSO NJE records so that you can select different options for different NJE nodes. If you need more than one NJE record, append a qualifier to the record name in the format NJE*qual* to generate a unique record ID (for example, NJER221 or NJE.JOB). This lets you code as many NJE records (per SYSID) as your operations require. This optional qualifier can be up to thirteen characters, and must immediately follow the characters NJE. If you use a period (.) as part of the qualifier string for the record name, eTrust CA-ACF2 counts it as one of the thirteen characters.

You must have at least one NJE record per GSO SYSID. If none exists, eTrust CA-ACF2 constructs a default record and inserts it into the database when you refresh eTrust CA-ACF2.

Displaying GSO NJE Record Information

The SHOW NJE and SHOW ACF2 subcommands display NJE record information.

eTrust CA-ACF2 Option Specifications (OPTS)

The OPTS record defines the global options available for use by the system.

Record ID	Fields
OPTS	ACCESS | NOACCESS BLPLOG | NOBLPLOG BYPSTATS | NOBYPSTATS CACHE | NOCACHE CMDREC | NOCMDREC CONSOLE(NOROLL | ROLL | NONE | WTP) CPF | NOCPF CPUTIME(LOCAL | GMT) DATE(MDY | DMY | YMD) DDB | NODDB DFTLID(*defaultlid*) DFTLNXG(*default_linux_group*) DFTLNXU(*default_linux_user*) DFTSTC(ACFSTCID | *lid*) ETAUDIT | NOETAUDIT INFOLIST(SECURITY,AUDIT | *privilegelist*) JOBCK | NOJOBCK KERBLVL(0 | 1) LDS | NOLDS MAXVIO(10 | *nnn*) MODE(ABORT | WARN | LOG | QUIET | RULE,*norule,no$mode*) NAMEHIDE | NONAMEHIDE NOTIFY | NONOTIFY PRIMARY(*ppp*) RPTSCOPE | NORPTSCOPE SECONDRY(*sss*) SHRDASD | NOSHRDASD STAMPSMF | NOSTAMPSMF STC | NOSTC SYSPLEX | NOSYSPLEX TAPEDSN | NOTAPEDSN TEMPDSN | NOTEMPDSN TNGMON | NOTNGMON UADS | NOUADS VTAMOPEN | NOVTAMOPEN WRNDAYS(1 | nnnn) XAPPLVLD | NOXAPPLVLD

Field Descriptions

ACCESS | NOACCESS
Specifies whether the ACCESS subcommand is enabled for processing. The default is NOACCESS.

Note: A refresh of the OPTS record and a F ACF2,NEWUID operator command to build the LID/UID cross-reference table are required to activate the ACCESS subcommand.

BLPLOG | NOBLPLOG
Specifies whether BLP accesses, when they are authorized through the TAPE-BLP or TAPE-LBL logonid record fields, or through the BLPPGM record, should produce a eTrust CA-ACF2 logging record. The default is NOBLPLOG.

BYPSTATS | NOBYPSTATS
Specifies whether to update the logonid last-access information for signon request. The SMF record is written when the logonid update is being bypassed. The command syntax is as follows:

```
ACF
SET C(GSO)
CHANGE OPTS BYPSTATS
```

CACHE | NOCACHE
Specifies whether the eTrust CA-ACF2 cache facility is to be activated. The default is NOCACHE. For information about the cache facility, see the "Maintaining Cache Records" chapter.

CMDREC | NOCMDREC
Specifies whether the TSO command statistics SMF records are written for all users. If you specify CMDREC, eTrust CA-ACF2 displays TSO command statistics records in the COMMAND SMF RECORDS= field in the SHOW TSO display. These statistics cannot be viewed with SHOW STATE, SHOW ACTIVE, or SHOW SYSTEMS.

If you specify NOCMDREC, command statistics records can be written for individual users using the TSO-TRC field in the logonid record.

CONSOLE(NOROLL | ROLL | NONE | WTP)
Specifies how eTrust CA-ACF2 messages for data set access loggings (ACF99900) and security violations (ACF99913) display.

- **NOROLL**
  Specifies that the messages are marked non-deletable, so they do not roll off the console screens in roll deleteable mode. (Roll deletable mode is set by the K S,DEL=RD console command). Messages still roll off console screens in roll or wrap mode. (Roll mode is set by the K S,DEL=R console command; wrap mode, by the K S,DEL=W command). NOROLL is the default.

- **ROLL**
  Specifies that the messages are not marked non-deletable. They roll off console screens in roll, roll deletable, or wrap mode. (No messages roll off screens in noroll mode, which is set by the K S,DEL=N console command).
- **NONE**
  Specifies that the ACF99900 and ACF99913 messages do not display.
- **WTP**
  Specifies that the messages are issued as write-to-programmer messages.

The ACF99900 message is normally displayed with routing code 9. ACF99900 messages are not issued for data set read accesses. The ACF99913 message is normally displayed with routing codes 1, 9, and 11.

When CONSOLE(WTP) is specified, both messages are displayed with routing code 11.

Routing code 1 is defined by IBM as indicating "operator action" messages, messages indicating a change in the system status. The master console is generally set to receive messages with routing code 1.

Routing code 9 is defined by IBM as indicating "system security" messages. The master console generally receives these messages, though some sites have a separate security console.

Routing code 11 is defined by IBM as indicating "programmer information" messages. These messages display in the job log for batch jobs and on the terminal for TSO sessions. They also display on consoles that are set to receive routing code 11, but most sites chose to reduce console clutter by excluding routing code 11.

Routing codes are described more fully in the IBM *z/OS MVS/ESA Routing and Descriptor Codes* manual.

CPF|NOCPF
Specifies whether the Command Propagation Facility (CPF) is to be active on this system. The default is NOCPF.

CPUTIME(LOCAL|GMT)
Specifies the time setting of the CPU. This field determines how eTrust CA-ACF2 calculates a user's time of access when zone record processing is performed. If you specify GMT (Greenwich Mean Time), eTrust CA-ACF2 bases all time zone calculations on the time-of-day (TOD) clock. In LOCAL setting, eTrust CA-ACF2 first adjusts the TOD clock by the value stored in the CVTTZ field of the communications vector table (CVT) and then bases all time-zone calculations on the adjusted TOD clock. The default is LOCAL.

DATE(MDY|DMY|YMD)
Indicates the date format to be used by eTrust CA-ACF2: month-day-year (MDY), day-month-year (DMY), or year-month-day (YMD). Date formats can be freely changed. The default is MDY. eTrust CA-ACF2 stores dates internally as packed-decimal Julian fields in the format CCYYDDDF: century, year, day of year, F sign. Year designations of 70-99 assume a date in the 20th century (1970-1999); year designations of 00-69 assume a date in the 21st century (2000-2069). This means that if you specify DMY in the GSO OPTS DATE field, 24/02/02 would represent February 24, 2002.

DDB|NODDB
Specifies whether distributed database support (DDB) is to be active on this system. The default is NODDB.

DFTLID(*defaultlid*)
Specifies the default logonid assigned to batch jobs that enter the system and do not have a logonid specified in their JCL. No default value is supplied for this option. A job that enters the system without a logonid or a global default logonid specified is rejected. If you update DFTLID, you must refresh the GSO OPTS record.

DFTLNXG(*default_linux_group*)
Specifies the name of the default Linux group profile record. It is used when a user accesses the LINUX/390 service and does not have a valid group defined in the Linux user profile record.

DFTLNXU(*default_linu_user*)
Specifies the name of the default Linux user profile record. It is used when a user accesses the LINUX/390 service and does not have a valid Linux user profile record.

DFTSTC(ACFSTCID|*lid*
Specifies the logonid assigned to a started task if the external procedure name (the member name specified in the START command) is not defined as a eTrust CA-ACF2 logonid. This is used only if STC is also specified. The default value is ACFSTCID. If you update DFTSTC, you must refresh the GSO OPTS record.

ETAUDIT|NOETAUDIT
Specifies whether notifications of security events are sent from this node to the eTrust Audit product. The default is NOETAUDIT. The ETAUDIT GSO record should also be inserted. This record specifies which security events are to be monitored.

If ETAUDIT is specified, some additional requirements must be met for the successful transmission of eTrust CA-ACF2 security events to eTrust Audit. The ENF SNMP monitor program and the eTrust Audit Client component must be installed at the appropriate levels, and the ENF SNMP monitor task must be running. Information about these requirements can be found on SupportConnect.com under the eTrust Audit product. The eTrust Audit policy must also be configured to receive event notifications from eTrust CA-ACF2. Information about eTrust Audit policy management can be found in the eTrust Audit product documentation.

Note: The GSO ETAUDIT record will not be activated unless the ETAUDIT OPTS record field is specified.

INFOLIST(SECURITY,AUDIT|*privilegelist*)
Specifies the logonid attributes you want a logonid to have to be able to list infostorage records. It does not convey the authority to compile and store resource or access rules. Logonids with the privileges listed in the INFOLIST field are not affected by scopes. As indicated here, by default all logonids with the SECURITY or AUDIT privilege can list infostorage records.

You can bypass INFOLIST processing during validations performed for user-created structured infostorage records. For information about INFOLIST bypass processing, see the Creating Structured Infostorage Records section in the "Special Usage Considerations" chapter in the *Systems Programmer Guide*.

JOBCK|NOJOBCK
Indicates whether eTrust CA-ACF2 should check for the JOB field in the logonid record of the users who submit batch jobs or the logonid under which the batch job runs. It does not affect foreground jobs, but does affect any background or batch jobs, and any jobs submitted from these sources. The default value is NOJOBCK. For more information about submitting batch jobs, see Submitting Batch Jobs in the "Controlling System Entry" chapter.

KERBLVL(0|1)
Specifies the level of encryption support. 0 indicates the DES kerberos key is generated for passwords. 1 indicates the DES, DES3 and DESD kerberos keys will be generated for passwords. This level enables the use of the DES, DES3, and DESD encryption settings on the GSO REALM and KERB USER PROFILE records.

Note: It is recommended that KERBLVL be modified from 0 to 1 only when the status of all connected security services are at this level of support. Once KERBLVL is modified, it is not recommended to revert back to KERBLVL(0) because this may result in invalid data. Following a refresh of the GSO OPTS record, a refresh of the GSO REALM record, a rebuild of user profile records, and a rebuild of OMVS tables are required to activate the new KERBLVL setting for the currently active REALM records.

LDS|NOLDS
Specifies whether LDAP Directory Serverics (LDS) is to be active on this system. The default is NOLDS.

MAXVIO(10|*nnn*)
Specifies the maximum number of security violations that can occur in a single job or TSO session before eTrust CA-ACF2 terminates the job. The default value is 10; the maximum value is 32767. **Note:** The number of security violations per single batch job or TSO session is incremented by both resource and data set violations. However, that number of violations is only checked against MAXVIO after a data set violation.

MODE(ABORT|WARN|LOG|QUIET|RULE,*norule,no$mode*)
Specifies the mode of the eTrust CA-ACF2 system as it relates to data set access. The MODE determines what actions eTrust CA-ACF2 takes when a request to access a data set is considered a violation.

Note: MODE does not affect resource rules; they are always in ABORT mode.

Valid modes are:

- **QUIET**—disables eTrust CA-ACF2 data set access rule validations. Logonid, source, and similar system access validations still take place.
- **LOG**—logs data set access violations, but lets access continue.
- **WARN**—logs data set access violations, issues warning messages, and lets access continue.
- **ABORT**—logs attempted violations, issues violation messages, and denies the access. This is the default value.
- **RULE,*norule,no$mode*—**an interim mode you can use to gradually migrate to full security. With this mode, the site specifies what conditions exist in the absence of access rules or $MODE control statements in the access rule. Depending upon the condition, the data set access is validated accordingly.

  Using RULE mode lets you store access rules gradually and allows access to the system while eTrust CA-ACF2 is up and running. The value of the $MODE control statement can be QUIET, LOG, WARN, or ABORT, as defined previously. The $MODE control statement has meaning only when the MODE(RULE,*norule,no$mode*) option is in effect and when eTrust CA-ACF2 determines that a violation has occurred. The two positional parameters are:

  - *norule*—specifies the action eTrust CA-ACF2 takes if no access rule set is found when RULE mode is in effect. The *norule* value can be QUIET, LOG, WARN, or ABORT, as defined previously.

- *no$mode*—specifies the action eTrust CA-ACF2 takes if no $MODE control statement is found in the applicable access rule set when MODE(RULE) data set access control is in effect. The *no-$mode* value can be QUIET, LOG, WARN, or ABORT as defined previously.

For example, a site selects MODE(RULE,WARN,ABORT). User ABC requests write access to data set TEST.DATA. If no TEST rule set exists, eTrust CA-ACF2 bases its decision on the norule value, which in this case is WARN. If a TEST rule set does exist, but does not grant user ABC write access, eTrust CA-ACF2 checks the $MODE control statement in the access rule and bases the allow or prevent decision on the $MODE value. If $MODE(LOG) was specified in the access rule set, then user ABC is permitted to write to TEST.DATA and eTrust CA-ACF2 creates a logging record. However, if no $MODE was specified in the rule set, the no$mode value (in this case, ABORT) is used.

RULE mode does not apply to programs listed in the BLPPGM, LOGPGM, and PPGM records. This means that when eTrust CA-ACF2 is in RULE mode and a program in a BLPPGM, LOGPGM, or PPGM record attempts to access a data set, eTrust CA-ACF2 ignores the MODE field and considers itself to be in ABORT mode.

Note: Always specify the REP parameter when you add or modify the MODE field of the OPTS record.

NAMEHIDE|NONAMEHIDE

Specifies whether the names of data sets, files, and directories are protected from disclosure to users who do not have at least READ access to the data. Rule validation, and, if MLS is active, security label checking, will be performed to allow or prevent the user from viewing the names of data sets in a catalog or on a VTOC, and files and directories in listing the contents of a UNIX directory. However, if a user requests to view the name of a specific data set, file, or directory, the names will appear but the user may not be able to access the data. Name hiding can be used on a system where MLS is not active. However, MLS must be active to support name hiding for UNIX files and directories. The default is NONAMEHIDE, do not hide the names of data sets, files and directories from users.

When name-hiding is active, users are prevented from seeing the names of data sets to which they do not have at least READ access in a catalog, unless the exact name of the data set is specified. To hide the names of data sets in a catalog, before activating the GSO OPTS name-hiding option, the security administrator must make sure the following SAFDEF record is active for SVC026.

```
show safdef

-- SYSTEM AUTHORIZATION FACILITY DEFINITIONS --

         RACROUTE REQUEST=AUTH,CLASS='DATASET'

SVC026   JOBNAME=********   USERID=********   PROGRAM=********   RB=SVC026
         RETCODE=4          SAFDEF=GSO        MODE=GLOBAL       SUBSYS=****
         FUNCRET=4          FUNCRSN=0
```

To create the record, issue the following commands:

```
set control(gso)
 CONTROL
insert safdef.svc026 id(svc026) rb(svc026) mode(global)
    racroute(request=auth,class=dataset)
```

To activate the record, issue the following command:

```
f acf2,refresh(safdef)
```

In addition, the security administrator must create access rules, as necessary, to allow users to access the catalog. The following rule applies to catalog names that begin with ICF.V:

```
set rule
 RULE
comp *
  $KEY(ICF)
    V-.- UID(*****USERA) READ(A) EXEC(A)
store
```

Note: When a user issues an =3.4 in ISPF to access data sets, and does not specify a volume serial number, catalog processing is performed.

When name-hiding is active, users are prevented from seeing the names of data sets to which they do not have at least READ access on a VTOC when directly reading the VTOC or VTOC index or using CCHHR for CVAF reading of a VTOC. To allow users to see the names of data sets, a security administrator must give users READ access to the **STGADMIN.IFG.READVTOC.*volser*** resource in the FACILITY class.

The following rule allows USERA to see data set names on a VTOC:

```
set resource(fac)
 RESOURCE
comp *
  $KEY(STGADMIN) TYPE(FAC)
   IFG.READVTOC.- UID(*****USERA) READ(A)
store
```

To activate the rule, issue the following command:

```
f acf2,rebuild(fac)
```

Note: When a user issues an =3.4 in ISPF to access data sets, and specifies a volume serial number, VTOC processing is performed.

Name hiding for Data Sets:

- If MLS is inactive, only access and resource rule validation will be performed to allow or prevent the user from viewing the names of data sets in a catalog or on a VTOC.
- If MLS is active, a MAC label dominance check will be done for each data set name in a catalog or on a VTOC to determine whether a user can see it. If the MAC check allows the access, access and resource rule validation is performed to ultimately allow or prevent the user from viewing the names of data sets in a catalog or on a VTOC.

Name hiding for UNIX Files and Directories:

- If MLS is active, a MAC label dominance check will be performed for each file and directory name to determine if it can be viewed by the user who issues the command to list the contents of a directory.
- If MLS is inactive, name hiding is not supported in eTrust CA-ACF2.

  Warning! *Name hiding is not available in an HFS file system; only in a zFS file system.*

  Note: Name hiding degrades the performance of a system. When name hiding is active in an MLS environment, system performance is further degraded. Do not activate name hiding if any system sharing the eTrust CA-ACF2 databases does not meet the minimum software requirements for MLS support. Use of the name hiding option should not cause problems on these systems, but it does not provide full protection on these systems. You must be operating at z/OS R1V5 or above to activate name hiding in an eTrust CA-ACF2 system. For more information about name-hiding in a MLS system, see the *Multilvel Security Planning Guide*.

NOTIFY|NONOTIFY

Indicates whether eTrust CA-ACF2 displays the date, time, and input source ID of a user's last system access. NOTIFY specifies that the message is displayed at logon time. If a password change prompt or other eTrust CA-ACF2 message must be displayed, the last access message is not displayed. The default is NONOTIFY.

PRIMARY(*ppp*)

The three-byte character field specifying the language code for the global (system-wide) primary language.

Although you can specify your own language codes, there are accepted standard values for most common languages. These accepted values are documented in many places, including the *z/OS MVS Programming: Assembler Services Reference* , *MVS Programming Assembler Services Guide*.

When a SAF EXTRACT call is issued to obtain the profile information for a user, if no primary language was specified for the user in a USER PROFILE LANGUAGE record, this global primary language is returned in the extracted primary language field.

If eTrust CA-ACF2 national language support was implemented (documented in the *Getting Started* guide), the global primary language is used as the primary language for all eTrust CA-ACF2 system messages, that is, eTrust CA-ACF2 issues the message in that language, if possible. The global primary language is also used as the primary language for eTrust CA-ACF2 user-directed messages if no primary language was specified for the user in a USER PROFILE LANGUAGE record. If no global primary language is specified, eTrust CA-ACF2 uses ENU (U.S. English) as a default.

RPTSCOPE|NORPTSCOPE

Indicates whether users are restricted to view selected eTrust CA-ACF2 journal SMF records through the report utilities based on their particular scope record and privileges. RPTSCOPE specifies that reports are limited, or scoped, to the data that the logonid is authorized to see. NORPTSCOPE is the default. See the *Reports and Utilities Guide* for a more detailed description of the effect of this option on eTrust CA-ACF2 reports.

SECONDRY(*sss*)

The three-byte character field specifying the language code for the global (system-wide) secondary language.

Although you can specify your own language codes, there are accepted standard values for most common languages. These accepted values are documented in many places, including the *MVS Programming: Assembler Services Reference* or, *MVS Programming: Assembler Services Guide*.

When a SAF EXTRACT call is issued to obtain the profile information for a user, if no secondary language was specified for the user in a USER PROFILE LANGUAGE record, this global secondary language is returned in the extracted secondary language field.

If eTrust CA-ACF2 national language support was implemented (documented in the *Getting Started* guide), the global secondary language is used as the secondary language for all eTrust CA-ACF2 system messages, that is, eTrust CA-ACF2 issues the message in that language, if possible, when the message could not be located or issued in the primary language. The global secondary language is also used as the secondary language for eTrust CA-ACF2 user-directed messages if no secondary language was specified for the user in a USER PROFILE LANGUAGE record. If no global secondary language is specified, eTrust CA-ACF2 uses ENU (U.S. English) as a default.

SHRDASD|NOSHRDASD

Specifies whether eTrust CA-ACF2 enforces shared DASD protocol to preserve the integrity of the VSAM clusters. Normally, eTrust CA-ACF2 shared DASD logic is activated by the shared indicator in the UCB. We recommend that you specify SHRDASD.

If your site is using cache synchronization, you must specify SHRDASD to let eTrust CA-ACF2 activate the synchronizers.

STAMPSMF|NOSTAMPSMF

Specifies whether eTrust CA-ACF2 places the logonid in the SMF User Identification Field at job initiation time. This allows all SMF records (not just eTrust CA-ACF2 SMF records) generated by the job to contain the logonid. Use care when choosing the SMF stamp option. Some sites and some other vendor-supplied packages use this field for a communication area. The default value is NOSTAMPSMF.

STC|NOSTC

Specifies whether eTrust CA-ACF2 validates data set and resource accesses by started tasks. No data set or resource validation is performed if you specify NOSTC. If you specify STC, eTrust CA-ACF2 uses the started task name as the logonid if it is present in the logonid database. If you specify STC and no logonid exists for the started task, eTrust CA-ACF2 uses the value in DFTSTC. The default value is NOSTC. Logonid records are not required for started tasks when NOSTC is in effect.

The following logonids must be inserted with the STC, NON-CNCL, and MUSASS attributes before the GSO STC option is turned on: CATALOG, SMF, SMS, DUMPSRV, and MMS.

SYSPLEX|NOSYSPLEX

Specifies whether the XES or XCF service is to be used for SYSPLEX processing. This field must be turned on for eTrust CA-ACF2 processing to start using SYSPLEX whether the GSO SYSPLEX record is defined or not. The default is NOSYSPLEX.

TAPEDSN|NOTAPEDSN

Specifies whether eTrust CA-ACF2 protects data sets on tape volumes that do not match those listed in SECVOLS at the data set level. (TAPEDSN should only be specified if a site is running a tape management system. Otherwise, users may be able to spoof a data set name obtaining unauthorized access to tape data sets since MVS only checks the last 17 characters of the data set name specified in the JCL with the 17 characters of the data set name stored on the tape label and eTrust CA-ACF2 will validate the entire data set name specified in the JCL.) See Execution Flow in the chapter "Maintaining Access Rules" for more information. The default value is NOTAPEDSN, signifying no eTrust CA-ACF2 (data set validation) protection for these tape volumes.

TEMPDSN|NOTEMPDSN

Specifies whether eTrust CA-ACF2 should enable temporary data set protection. Normally, temporary data sets can only be accessed by the job using them. However, in some cases, temporary data sets can be left allocated on a storage device after the job has terminated. eTrust CA-ACF2 rules cannot be easily written to protect temporary data sets because the high-level qualifier of such data sets is dynamic. NOTEMPDSN is the default.

TEMPDSN lets you protect temporary datasets allocated to DASD. When this option is enabled, only the job that created the temporary data set can access it. If eTrust CA-ACF2 detected a user accessing a temporary data set which the user does not own, normal data set access validation occurs. A user must have the NON-CNCL or similar privilege to scratch such data sets.

When implementing temporary data set protection, you should review your GSO OPTS MODE setting. A MODE of (RULE, QUIET, ABORT) specifies a mode of QUIET if no rule set exists. This effectively negates the temporary data set protection since it is unlikely that a rule exists.

eTrust CA-ACF2 enables temporary data set protection only at IPL time. When the setting is changed to TEMPDSN, eTrust CA-ACF2 ignores the change until the next IPL.

We recommend that eTrust CA-ACF2 be auto-started through the CAISEC00 member when using the temporary data set protection feature. This ensures that eTrust CA-ACF2 is able to properly record temporary data set processing for all jobs. See the eTrust CA-ACF2 System Initialization section in the *Getting Started* guide for details regarding automatic startup of eTrust CA-ACF2.

TNGMON|NOTNGMON
Specifies whether CA-Common Services trap messages are sent from this node. The CA-Common Services machine IP addresses are defined in GSO TNGNODE records. The default is NOTNGMON.

UADS|NOUADS
Specifies whether the TSO User Attribute Data Set (UADS) is used for TSO account, JCL procedures, and user profile control. NOUADS, the default, implies that all TSO logon information is extracted from the user's logonid record. UADS means that UADS is used. See the "System Requirements" chapter in the Getting Started guide for more information. Also, see Choosing the UADS or NOUADS Option in the "Controlling System Entry" chapter.

VTAMOPEN|NOVTAMOPEN
Specifies whether VTAM ACB OPENs are validated. This option lets you protect your site's VTAM APPLIDs. For more information about VTAM ACB OPEN validations, see the *Systems Programmer Guide*.

WRNDAYS(1|nnnn)
Specifies the number of days a warning is issued to the user before their logonid expires. On those days, the following warning message is displayed every time the user accesses the system:

ACF01139 YOUR LOGONID WILL EXPIRE ON *date*

Date indicates the date (mm/dd/yy) on which the logonid will expire.

The default WRNDAYS is one. Specify WRNDAYS(0) to disable the option.

XAPPLVLD|NOXAPPLVLD

Specifies whether eTrust CA-ACF2 will assign an APPL if no APPL is passed by the caller. On a SAF VERIFY/VERIFYX request, CA-ACF2 will assign an APPL or 'MVS' followed by the 4 character SMFID of the system. On a NON-SAF signon request, if no APPL is passed by the caller, CA=ACF2 will assign an APPL of 'TSO' followed by the 4 character SMFID of the system for TSO signon or 'MVS' followed by the 4 character SMFID of the system for any other signon. The default is NOXAPPLVLD.

Displaying GSO OPTS Record Information

The SHOW STATE subcommand of the ACF command displays the values specified in the OPTS record for the BLPLOG, CACHE, CPF, CPUTIME, DATE, DDB, DFTLID, DFTSTC, ETAUDIT, INFOLIST, JOBCK, LDS, MAXVIO, MODE, PRIMARY, RPTSCOPE, SECONDRY, STC, SYSPLEX, TAPEDSN, TEMPDSN, TNGMON, UADS, VTAMOPEN, WRNDAYS, and XAPPLVLD fields.

The SHOW ACTIVE subcommand of the ACF command displays whether the VTAMOPEN intercept has gained control.

The SHOW SYSTEMS subcommand displays the values specified for the CONSOLE, NOTIFY, SHRDASD, and STAMPSMF fields.

The SHOW ACF2 subcommand displays all of these values.

The SHOW TSO subcommand displays the value specified for the CMDREC field.

PDS Member-Level Protection List (PDS)

The PDS record specifies those partitioned data sets (PDS) for which eTrust CA-ACF2 provides member-level protection. When a member of a PDS defined here is accessed, eTrust CA-ACF2 performs a resource validation to determine if access should be allowed. Also see Appendix D, "Implementing Member-Level Protection."

Record ID	Fields
PDS*qual*	LIBRARY(*library*) RSRCTYPE(*typecode*) VOLUME(*vol1,…,vol256*)

Field Descriptions

LIBRARY(*library*)
Defines a fully qualified PDS library name for which member-level protection should be performed. This field cannot be masked. Define only one library name per PDS record.

RSRCTYPE(*typecode*)
Defines the three-character resource rule type code under which the member-level validation occurs. This field is eight characters to accommodate the special value PDSALLOW, which when used in place of a type code, indicates that the specified library is not to be under member-level control. If more than three characters are specified, the first three characters are used as the type code.

VOLUME(*vol1,...,vol256*)
Defines the volume serial number on which the library described previously must reside for member-level protection to be performed. This field cannot be masked. If volume is not specified, member-level protection is performed for the library no matter where it resides. Up to 256 volumes can be specified.

Creating Multiple GSO PDS Records

If you need more than one PDS record, append a qualifier to the record name to generate a unique record ID (for example, PDSABC or PDS.ABC). This optional qualifier can be up to 13 characters and must immediately follow the characters PDS. If you use a period as part of the qualifier string for the record name, eTrust CA-ACF2 counts it as one of the 13 characters.

Displaying Libraries with Member-Level Protection

The SHOW STATE and SHOW ACF2 subcommands of the ACF command display libraries currently under member-level protection control.

Protected Program List (PPGM)

The PPGM record specifies the programs that can be executed by privileged users. These programs can only be executed by unscoped users with the SECURITY privilege, by logonids with the NON-CNCL privilege, or by users with the PPGM field in their logonid records.

eTrust CA-ACF2 logs the execution of a program in this list at step initiation and at every data set open executed by the program. eTrust CA-ACF2 does not log data set opens performed by programs called by a program in the PPGM record. PPGM provides limited program protection. To protect all use of a program, use the CA SAF interface and resource rules. See the Relationship Among LOGPGM, MAINT, and PPGM section at the end of this chapter for more information.

eTrust CA-ACF2 provides a default PPGM record. You can change the values or supply others up to a total of 255 program masks.

Record ID	Fields
PPGM	PGM-MASK(IEHD-,FDR***,DRWD-,ICKDSF-,…,pgmmask255)

Field Descriptions

PGM-MASK(IEHD,FDR***,DRWD-,ICKDSF-**,...,pgmmask255)**
Specifies up to 255 program masks. Specify a one to eight-character program mask.

PPGM Record Considerations

The eTrust CA-ACF2 system is supplied with a PPGM record that specifies IEHD- (IBM IEHDASDR), FDR*** (Innovation Fast Dump/Restore), DRWD- (IBM program product), and ICKDSF- (also an IBM product). Programs can also be protected by moving them to a non-linklist library and writing appropriate access rules.

Additional program protection is available through the use of Note 6. For information, see Appendix B, "eTrust CA-ACF2 Notes," in the *Systems Programmer Guide*.

Displaying GSO PPGM Record Information

The SHOW PROGRAMS (or SHOW PGMS) and SHOW ACF2 subcommands of the ACF command display each program name specified in a PPGM record.

PROXY GSO Records (PROXY)

Specifies the default PROXY and EIM information. These fields are returned on an extract for the IRR.PROXY.DEFAULT Facility class profile.

Record ID	Fields
PROXY	BINDDN(*binddn*) BINDPTOD(00/00/00-00:00) BINDPW(*password*) DOMAINDN(*domainname*) ENABLE | DISABLE KERBREG(*registry name*) LOCALREG(*localregistry*) LDAPHOST(*ldapurl*) X509REG(*registry name*)

Field Descriptions

BINDDN(*binddn*)

Specifies the distinguished name (DN) which will be used in conjunction with the BIND password if the LDAP Server needs to supply an administrator or user identity to BIND with another LDAP Server. The *binddn* value can be 1-1023 characters in length. A DN is made up of attribute value pairs, separated by commas. For example:

```
'cn=Tom Brady,ou=Quarterback,o=New England,c=US'
```

```
'cn=Sammy Sosa,ou=Slugger,o=Chicago Cubs,c=US'
```

Note: The list of attribute pairs specified in the BINDDN field are considered a single value, therefore, it must be enclosed in single quotes. The single quotes will qualify the entire list of attribute value pairs as one value.

eTrust CA-ACF2 does not validate a valid BIND DN was entered.

BINDPTOD(00/00/00-00:00)

Specifies the date and time the BINDPW field was last changed. eTrust CA-ACF2 lists the date in the format mm/dd/yy, dd/mm/yy, or yy/mm/dd, depending on the DATE field of the GSO OPTS record. You cannot set this field. eTrust CA-ACF2 maintains and displays it. (Eight-byte binary field)

BINDPW(*binddn*)

Specifies the password for the DN defined in the BINDDN parameter. The BIND password can contain 1-128 characters. Both uppercase and lowercase are accepted and maintained in the case in which they are entered.

eTrust CA-ACF2 does not validate that a valid BIND password was entered.

DOMAINDN
The distinguished name of an EIM domain. The field can be up to 1023 characters in length.

ENABLE|DISABLE
Specifies whether new connections may be established with the specified EIM domain.

KERBREG(*registry name*)
Specifies a one to 255-character name identifying the KERBEROS registry associated with the EIM or PROXY definition. Any value entered will be uppercased. To eliminate a previously specified KERBREG value on a CHANGE statement, specify KERBREG().

LOCALREG
The name of the local eTrust CA-ACF2 registry. The field can be up to 255 characters in length.

LDAPHOST(*ldapurl*)
Specifies the URL of the LDAP server that the z/OS LDAP Server will contact when acting as a proxy on behalf of a requester. An LDAP URL has a format such as ldap://123.45.6:389 or ldaps://123.45.6:636, where ldaps indicates that an SSL connection is desired for a higher level of security. LDAP will also allow you to specify the host name portion of the URL using either the text form (LDAP.HOST.CA.COM) or the dotted decimal address (111.222.33.44). The port number is appended to the host name, separated by a colon ':'. See your LDAP server documentation for information on how to set up the LDAP server for SSL connections.

This value must be at least 10 bytes long and can be up to 1023 bytes long. It must start with either ldap:// or ldaps://. Any characters may be entered in the remaining portion of the URL, but you should ensure that the URL conforms to TCP/IP conventions. Normally, characters such as commas, blanks, parenthesis, semicolons and single quotes are not allowed in a host name. These characters will be accepted if the LDAPHOST is enclosed in single quotes.

eTrust CA-ACF2 does not validate that the contents of the URL are valid.

X509REG(*registry name*)
Specifies a one to 255-character name identifying the X509 registry associated with this EIM or PROXY definition. Any value entered will be uppercase. To eliminate a previously specified X509REG value on a CHANGE statement, specify X509REG().

Password Maintenance and Support (PSWD)

The PSWD record defines the various logonid password options and controls.

Record ID	Fields
PSWD	MAXTRY(1 | *nnn*) MINPSWD(1 | *n*) PASSLMT(2 | *nnn*) PSWDALPH | NOPSWDALPH PSWDALT | NOPSWDALT PSWDFRC | NOPSWDFRC PSWDHST | NOPSWDHST PSWDJES | NOPSWDJES PSWDLID | NOPSWDLID PSWDMAX(0 | _) PSWDMIN(0 | _) PSWDMIXD | NOPSWDMIXD PSWDNCH | NOPSWDNCH PSWDNMIC | NOPSWDNMIC PSWDNUM | NOPSWDNUM PSWDPAIR(0 | *n*) PSWDPLST() PSWDSIM(0 | *n*) PSWDSPLT | NOPSWDSPLT PSWDREQ | NOPSWDREQ PSWDRSV | NOPSWDRSV PSWDVFY | NOPSWDVFY PSWDXTR | NOPSWDXTR PSWDVOWL | NOPSWDVOWL PSWNAGE | NOPSWNAGE PSWXHIST | NOPSWXHIST PSWXHST#(0 | *nn)* WRNDAYS(1 | *nnn*)

Field Descriptions

MAXTRY(1 | *nnn*)

Specifies the maximum number of attempts, including the initial password entry, that are permitted before the terminal session is canceled. The default value is one. The maximum value is 255. If you specify MAXTRY(0), a user can still logon. **Note:** The display of the MAXTRY option in the show output appears under the PASSWORD OPTIONS IN EFFECT heading with a label of LOGON RETRY COUNT=.

MINPSWD(1|*n*)
Specifies the minimum number of characters required in a new password. When eTrust CA-ACF2 is first installed, you should set MINPSWD to one to permit conversion of the passwords currently in UADS. You can raise the minimum later. The old passwords continue to be valid until they are changed or expire. The default value is one. The maximum value is eight. **Note:** Changes to this parameter take effect at the next password change of the user.

PASSLMT(2|*nnn*)
Specifies the maximum number of invalid password attempts permitted in a single day before eTrust CA-ACF2 denies all accesses to the system by the logonid. For example, if the maximum number of invalid password attempts is two, eTrust CA-ACF2 denies all access attempts after the second invalid attempt. If you try to log on after the PASSLMT has been reached, you receive a message telling you that your logonid has been suspended. Security administrators and account managers can reset the invalid password count by entering the following operator command:

```
F ACF2,RESET(logonid)
```

They can also use the ISPF panels or the ACF CHANGE subcommand. The default value is two. The maximum value is 32767.

If you specify PASSLMT(0), all user logonids are suspended the next time they try to logon, regardless of whether they have any invalid access attempts.

PSWDALPH|NOPSWDALPH
Specifies whether eTrust CA-ACF2 requires at least one alphabetic (a-z or A-Z) character to be present in a new password. The default is NOPSWDALPH, which specifies that eTrust CA-ACF2 will not validate the new password for at least one alphabetic character. **Note:** Changes to this parameter take effect at the next password change of the user.

PSWDALT|NOPSWDALT
Specifies whether users can enter new passwords at system entry (logon) time. The default is PSWDALT, which permits password alteration. A user can also change his password by changing the PASSWORD field of his logonid record with the ACF CHANGE subcommand. To prevent such changes, you must also specify the PSWDNCH option.

PSWDFRC|NOPSWDFRC
Specifies whether a user is forced to change the password at the next logon whenever someone other than the user, such as a security administrator or account manager, changes the password. NOPSWDALT and PSWDFRC conflict and you should not use them together. If you set PSWDFRC, eTrust CA-ACF2 uses the PSWDALT option. The default is PSWDFRC.

PSWDHST|NOPSWDHST

Specifies whether users can enter new passwords that match previous passwords. eTrust CA-ACF2 remembers the user's last four passwords (the current password and the previous three). The default is NOPSWDHST, which specifies that password history is not checked.

Note: PSWDHST implements the support formerly in Note 12. If PSWDHST is used, remove Note 12.

PSWDJES|NOPSWDJES

Indicates whether invalid passwords used in batch jobs should be counted towards the invalid password limit (PASSLMT) in a single day (that is, update PSWD-VIO count). The default value is NOPSWDJES.

PSWDLID|NOPSWDLID

Specifies whether eTrust CA-ACF2 will check if a new password matches the logonid. PSWDLID specifies that new passwords will be checked and rejected if they match the logonid. Before the password is compared to the logonid, it is temporarily upper-cased. The default is NOPSWDLID, which specifies that passwords are not checked for logonid match. **Note:** Changes to this parameter take effect at the next password change of the user.

PSWDMAX(0|_)

Specifies the global value for the maximum number of days (based on the date specified in the PSWD-TOD field) permitted between password changes before the password expires. Any non-zero value in the LIDREC MAXDAYS field will override this value for validations. A zero in the LIDREC MAXDAYS field will also override this value if the LIDZMAX flag is also set in the LIDREC. Zero specifies that there is no global value set; the value in the LIDREC MAXDAYS field will be used for validations.

PSWDMIN(0|_)

Specifies the global value for the minimum number of days that must elapse before a user can change his password. Any non-zero value in the LIDREC MINDAYS field will override this value for validations. A zero in the LIDREC MINDAYS field will also override this value if the LIDZMIN flag is also set in the LIDREC. Zero specifies that there is no global value set, the value in the LIDREC MINDAYS field will be used for validations.

Note: If there are currently non-zero values in the MAXDAYS and MINDAYS fields for a LIDREC and you would now like the global value to apply, you must do the following:

```
change lidrec maxdays(0) mindays(0)
```

If there is currently a zero value for either field and you want the zero to apply you must set the LIDZMAX flag for the MAXDAYS zero value to apply and the LIDZMIN flag for the MINDAYS zero value to apply.

PSWDMIXD|NOPSWDMIXD

Specifies passwords are case sensitive. The default is NOPSWDMIXD. PSWDMIXD is a global setting for all Logonids.

Once PSWDMIXD is turned on, existing (current) passwords are not affected. That is, they can be entered in any combination of upper and lower case characters and they will always be uppercased before password validation is performed. Once a user has changed their password while PSWDMIXD is on, their password becomes case sensitive. If PSWDMIXD is turned off, their password remains case-sensitive until they set a new password while NOPSWDMIXD is set. There is a new field in the logonid record called PSWD-MIX to indicate that the current password is case-sensitive.

Note: Before setting PSWDMIXD on, read the section "Considerations for Mixed-Case Passwords" **carefully**.

Note: Changes to this parameter take effect at the next password change of the user.

PSWDNCH|NOPSWDNCH

Specifies whether users can change their passwords with the ACF CHANGE command. The default is NOPSWDNCH, which permits password changes through the ACF CHANGE command. This option does not affect administrators who are changing the passwords of other users. (It does affect administrators changing their own passwords.)

The purpose of the PSWDNCH option is to require users to change their passwords only at system entry. PSWDNCH can also be used with the NOPSWDALT option to require all password changes to be done by security administrators.

PSWDNMIC|NOPSWDNMIC

Specifies that eTrust CA-ACF2 requires at least one numeric (0-9) character in a new password. The default is NOPSWDNMIC, which specifices that eTrust CA-ACF2 will not validate the new password for at least one numeric character. **Note:** Changes to this parameter take effect at the next password change of the user.

PSWDNUM|NOPSWDNUM

Specifies whether eTrust CA-ACF2 will check if a new password is all numeric. PSWDNUM specifies that new passwords will be checked and rejected if they contain only numerics (digits 0-9). The default is NOPSWDNUM, which specifies that passwords are not checked for all numeric characters. **Note:** Changes to this parameter take effect at the next password change of the user.

PSWDPAIR(0 | *n***)**

Specifies the number of consecutively repeated characters allowed to be in a password. The default value for this field is null, which indicates that eTrust CA-ACF2 will not validate the new password for consecutively repeated pairs of characters. Valid values are 0-7. To set the PSWDPAIR field to null, issue the command: PSWDPAIR().

For example, when PSWDPAIR(1) is specified, a new password can specify up to one consecutively repeated pair of characters. So a valid new password could be 'RABBIT', 'NEEDED' or 'NEWPSWD'. However, eTrust CA-ACF2 will not allow 'RABBBIT' since 'BBB' is considered two consecutively repeated pairs of characters. **Note:** Changes to this parameter take effect at the next password change of the user.

PSWDPLST( )

Specifies that eTrust CA-ACF2 will allow new password to contain non-alphanumeric characters in addition to default password characters, which are alphanumeric (a-z, A-Z, 0-9) and national (@ # $). There are 12 non-alphanumeric characters it can specify. By default it specifies none.

The following are the supported non-alphanumeric characters:

Character name	Character	Notes
Asterisk	*	
Ampersand	&	
Carat	^	
Colon	:	
Equal sign	=	
Hyphen	-	
Exclamation Point	!	
Period	.	
Percent Sign	%	
Question Mark	?	
Underscore	_	
Vertical Line	|	

Example: When PSWDPLST(& * -) is specified, a valid password can contain ampersand(&), asterisk(*), and hyphen(-) characters. The following are examples of valid passwords:

```
'NEW#PWD', 'NEW*PS&D', '123#PSWD', 'NEWPSWD@' or '#NEW-PWD'
```

Note: Changes to this parameter take effect at the next password change of the user.

PSWDSIM(0|*n*)

Specifies whether password similarity checking is to be performed. Password similarity checking is done whenever a new password is entered and *n* is greater than zero. If there are *n* characters that match in the same positions of both passwords, then the old and new passwords are too similar. eTrust CA-ACF2 temporarily upper-cases the new and old passwords before checking for similarities. This is because one or both passwords could be a mixed-case password. The value of *n* can be any integer from 0 to 7. The default value is 0, which means no similarity checking will be done. **Note:** PSWDSIM similarity checking does not take effect until the user has changed their password at least once while PSWDSIM was set to a value greater than zero.

Examples of similar passwords for PSWDSIM(3):

- jpL123 is too similar to JplxYZ
- RePeAteD is too similar to dEfEaT00
- 12345Big is too similar to ABCDEBIG

PSWDSPLT|NOPSWDSPLT

Specifies whether a password contains a national or a user-defined character between the first and last position. The default is NOPSWDSPLT. **Note:** If a user-defined character list, PSWDPLST, is not specified, then the new password must contain a character from the national character set between the first and last positions.

Note: When PSWDSPLT is specified, a valid password cannot have a national or user-defined character in the first or last position.

Example: When PSWDSPLT and PSWDPLST(& %) are specified, then a valid password must contain either a national or user-defined (& %) character **between** the first and last position. The following are examples of valid passwords:

```
'BIG&RED', 'BIG%RED', 'BIG#RED', 'BIG$RED', '456%RED' or 'BIG@RED'
```

Example: When PSWDSPLT and PSWDPLST() are specified, then a valid password must contain a national character **between** the first and last position. The following are examples of valid passwords:

```
'NEW#PASS', 'N$EWPASS', '123#PASS' or 'NEWPAS@S'
```

Note: Changes to this parameter take effect at the next password change of the user.

PSWDREQ|NOPSWDREQ

Specifies whether a password is required for all logonids, except STC and RESTRICT logonids when a logonid is inserted or a change command is issued for a logonid to remove the STC or RESTRICT privilege. When PSWDREQ is set, logonids that are inserted or changed (to remove STC or RESTRICT), the password field will be checked for a valid value. The default is PSWDREQ.

Note: If PSWDREQ is set and a LOGONID is being inserted without a password or a CHANGE is done to remove STC or RESTRICT without a password, the following message will be issued:

ACF02037 KEYWORD PASSWORD IS REQUIRED

PSWDRSV|NOPSWDRSV

Specifies whether users can enter new passwords that begin with a reserved word prefix. The default is NOPSWDRSV, which specifies that passwords are not checked for reserved word prefixes. The list of reserved prefixes is specified in the GSO RESWORD infostorage record. See Reserved Word Prefix List (RESWORD) later in this chapter for a list of the default reserved prefixes.

Note: Changes to this parameter take effect at the next password change of the user.

PSWDVFY|NOPSWDVFY

Specifies whether a user is required to verify their new password. This is only required if the user is updating the current password under an ACF TSO session and does not specify the new password value on the CHANGE command line. The default is NOPSWDVFY.

PSWDXTR|NOPSWDXTR

Specifies whether authorized programs can extract passwords from logonids. Some program products let you submit a job at one node that processes on another node. Sometimes these products perform a SAF EXTRACT call and transmit the password across the network.

This can expose the password to those who might be monitoring the communications line. To avoid this exposure, eTrust CA-ACF2 does not return the actual password; instead, it returns a halfway-encrypted version of the password, using the same halfway encryption that is used when transmitting passwords in NJE jobs. It is not possible to reverse this encryption to produce the clear text password. (The term “halfway encryption” is used because this encryption is followed by a second encryption, also non-reversible, on the system that receives the password.)

When PSWDXTR is specified, the following events take place whenever the password is changed for a logonid:

- The PSWD-XTR bit is turned on in the logonid record
- The halfway-encrypted password is encrypted yet again and stored in the logonid record. (The purpose of this extra encryption is to keep those who have access to backups of the eTrust CA-ACF2 LOGONIDS VSAM cluster from being able to obtain the halfway-encrypted passwords.)

When PSWDXTR is turned on, an extractable halfway-encrypted password is not available for a logonid until the password is changed. To make extractable halfway-encrypted passwords available for all, an unscoped security administrator or account manager must take the following steps:

1. Set the PSWDXTR field in the GSO PSWD record.
2. Refresh the GSO PSWD record.
3. Expire the passwords for all logonids by issuing the CHANGE LIKE(-) PSWD-EXP command. This forces all users to change their passwords the next time they log on.

When NOPSWDXTR is specified, eTrust CA-ACF2 does not store halfway-encrypted passwords. The next time the password is changed for each logonid, the following occurs:

- The PSWD-XTR bit in the logonid record is turned off
- The halfway-encrypted password field in the logonid record is cleared.

To stop making halfway-encrypted passwords available for extraction, an unscoped security administrator or account manager must take the following steps:

1. Set NOPSWDXTR in the GSO PSWD record.
2. Refresh the GSO PSWD record.
3. Make the existing halfway-encrypted passwords unavailable by issuing a CHANGE LIKE(-) NOPSWD-XTR command.

This ensures that halfway-encrypted passwords are not saved the next time the password is changed for each logonid, and that the existing halfway-encrypted passwords cannot be extracted.

Note: Changes to this parameter take effect at the next password change of the user.

PSWDVOWL|NOPSWDVOWL
Specifies whether eTrust CA-ACF2 will validate if a new password can specify vowel (A, E, I, O, U, a, e, i, o, u) characters. If NOPSWDVOWL is specified, vowels are not allowed. eTrust CA-ACF2 stores the password as uppercase, unless the GSO PSWD record specifies PSWDMIXD. The default is PSWDVOWL. **Note:** Changes to this parameter take effect at the next password change of the user.

PSWNAGE|NOPSWNAGE

Specifies that if an administrator is changing a password on behalf of another user and that password will be immediately expired, the password history will not be updated to include the temporary password that the administrator assigns.

Note: If you are using the sample new password exit and would like to use this feature, see the sample exit for the bits to check and update your exit.

PSWXHIST|NOPSWXHIST

Specifies that Extended Password History is to be used. This is an extension of password history, which is specified via the PSWDHST field. With PSWDHST, four previous passwords are checked against the new password. With PSWXHIST, you can extend this support to check up to 64 previous passwords. The number of previous passwords is determined by the value in PSWXHST#. Note that PSWDHST still works the same when NOPSWXHIST is specified. The default is NOPSWXHIST, which specifies that extended password history is not active. When PSWXHIST is on, previous passwords are stored in a PROFILE(USER) DIV(PASSWORD) record in the INFOSTG database where the record name is the same as the Logonid.

Note: If you currently use NOTE 12 to maintain password history, you need to be aware of the following:

- If you use NOTE 12 unmodified, then PSWDHST and PSWXHST will work concurrently with NOTE 12. However, there is no reason to continue using NOTE 12 because you can set PSWDHST on to replace NOTE 12 immediately.
- If you have modified the password external name table at the bottom of the NOTE 12 source code in order to add additional user defined fields, then PSWDHST and PSWXHST will work concurrently with NOTE 12. You can use PSWXHIST to **eventually** replace NOTE 12 but you need to decide when NOTE 12 can safely be removed. To help make this decision, let's look at an example: Suppose you have three user defined old password and TOD stamp fields in addition to the four provided by the original NOTE 12 code. That means you maintain seven old passwords altogether. If you set the GSO PSWD record to PSWXHIST PSWXHST#(7) then seven old passwords will be maintained by eTrust CA-ACF2. Note that PSWXHST# could be set higher than 7 if desired. When a password is changed, NOTE 12 pushes down all the old passwords, updating the first four eTrust CA-ACF2 defined fields and the three user defined fields. PSWXHIST processing will leave the first four fields alone since they have already been updated by Note 12, and then it will save the last six passwords in the PROFILE(USER) DIV(PASSWORD) record for the Logonid. After the user's password has been changed three times, the NOTE 12 user defined fields and the PROFILE(USER) DIV(PASSWORD) record fields will be in sync. Once this has been accomplished for all Logonids with passwords, NOTE 12 can safely be removed.

PSWXHST#(0 | *nn*)
The number of previous passwords to be retained when PSWXHIST is specified. A value of 0 or a value between 5 and 64 is valid. Values 1, 2, 3 and 4 are not valid because the password history that is maintained via PSWDHST keeps the first four previous passwords. If less than five previous passwords need to be retained, then specify PSWDHST NOPSWXHST PSWXHST#(0). The default is 0.

WRNDAYS(1 | *nnn*)
Specifies the number of days a warning is issued to the user before his password expires. On those days, the following warning message is displayed every time the user tries to access the system:

```
ACF01134  YOUR PASSWORD WILL EXPIRE ON mm/dd/yy
```

The default WRNDAYS value is one.

Considerations for Mixed-Case Passwords

There are several important things to consider before setting PSWDMIXD on. In order to use PSWDMIXD, you need to be sure that all applications that perform password validation can support it. Some applications may upper-case the password before it is passed to eTrust CA-ACF2. If an application does not support mixed-case password validation and PSWDMIXD is on, then the users of the application must type their passwords in upper-case.

The following is a list of the types of password validation that are supported. **Note:** eTrust CA-ACF2 r8 or above is required. This list includes requirements that are necessary for a given type of password validation.

- TSO logon
- JES2 batch jobs with //*PASSWORD card or PASSWORD= on the JOB card
- JES3 batch jobs with //*PASSWORD card or PASSWORD= on the JOB card
- eTrust CA-ACF2 password prompt on console (ACF79341 message reply). On the reply the password must be entered in single quotes if it is a mixed-case password.
- eTrust CA-ACF2 CICS logon and password reverify (due to VERIFY keyword on a resource rule or an idle timeout prompt). Further requirements for CICS are documented in the "Administration of the CICS Interface" chapter of the *CICS Support Guide*.
- eTrust CA-ACF2 IMS password reverify is not supported due to native IMS restrictions. Password reverification may be performed due to idle timeout or because of resource rules that contain the keyword VERIFY. For this reason, it is not advisable to use mixed-case passwords if IMS is being used. If only sigon validation is needed, the /FOR SIGN signon method must be used when logging on with a mixed-case password. Use the sample SIGNFMT format definition from eTrust CA-ACF2 for IMS r8.

The following is a list of the types of password validation that are not support

- Console logon—Any password that is used for console logon must contain only upper-case characters.
- eTrust CA-ACF2 IMS logon using the /SIGN command.
- eTrust CA-ACF2 IMS password reverification.

CPF Considerations

If PSWDSYNC, CPF, or extended password sync is active, make sure all nodes within the CPF environment use either PSWDMIXD or NOPSWDMIXD. If PSWDMIXD is used, all nodes must be running eTrust CA-ACF2 r8 or later.

DDB Password Sync Considerations

If you are using DDB Password Sync, make sure all nodes within the DDB environment use either PSWDMIXD or NOPSWDMIXD. If PSWDMIXD is used, all nodes must be running eTrust CA-ACF2 r8 or later.

Shared Logonids Database

If you are sharing the Logonids database with other z/OS systems, then all the systems must use either PSWDMIXD or NOPSWDMIXD. If PSWDMIXD is used, all systems must be running eTrust CA-ACF2 r8 or later.

If you are using Database Synchronization to share Logonids with a z/VM system protected by eTrust CA-ACF2 Security for VM, you cannot use PSWDMIXD at this time.

Routing Batch jobs

When routing batch jobs that contain a mixed-case password in the JCL, the execution node must be running eTrust CA-ACF2 rel 8.0 or above with PSWDMIXD on.

Displaying Password Options

Display the password options in effect with the SHOW STATE or SHOW ACF2 subcommand.

When listing out the GSO PSWD record, PSWDPAIR will not display if the field is set to nulls.

Note: The display of the MAXTRY option in the show output appears under the PASSWORD OPTIONS IN EFFECT heading with a label of LOGON RETRY COUNT=.

REALM GSO Record (REALM)

The REALM GSO record defines the characteristics of local and foreign Network Authentication and Privacy Service realms. Local realms are defined with a qualifier of KERBDFLT. The local REALM GSO record defines the characteristics of the local realm. Foreign REALM GSO records define inter-realm trust relationships and should be entered in pairs. Similar relationships should be defined on the foreign realm as well.

Record ID	Fields
REALM*qualifier*	CURKEYV(*Kerberos key version*) DEFTKTLF(*ticketlife*) DES | NODES DES3 | NODES3 DESD | NODESD KERBPSWD(*kerbpswd*) MAXTKTLF(*ticketlife*) MINTKTLF(*ticketlife*) REALM(*realmname*)

Field Descriptions

Qualifier
This is a one to eight character suffix that is used to identify a specific REALM record. For local realms, this must be KERBDFLT. The suffix might be separated from the record id by a period.

CURKEYV(*Kerberos key version*)
This value represents the current Kerberos key version. This field is updated when the password for the realm changes. It cannot be modified using the ACF command.

DEFTKTLF(*ticketlife*)
Default ticket lifetime for the local realm in seconds. This field is only valid for the local realm. If it is defined, then both MAXTKTLF and MINTKTLF must also be defined. The range of values for this field is 1 to 2,147,483,647 seconds.

DES|NODES
Enables the DES encryption type setting to be defined for this REALM. This field is valid when KERBLVL(1) is active on the GSO OPTS record. The default is DES.

DES3|NODES3
Enables the DES3 encryption type setting to be defined for this REALM. This field is valid when KERBLVL(1) is active on the GSO OPTS record. The default is DES3.

DESD|NODESD
Enables the DESD encryption type setting to be defined for this REALM. This field is valid when KERBLVL(1) is active on the GSO OPTS record. The default is DESD.

KERBPSWD(*kerbpswd*)
Specifies the realm password. The password has a maximum length of 8-characters and is case sensitive. EBCDIC variant characters should not be used. A password is required in order for the local realm to grant ticket-granting tickets and a password must be associated with the definition of a foreign realm inter-realm trust relationship or the definition is incomplete.

This field is intended for administrators to be able to associate a Kerberos-password with the definition of a realm. It is not the same as a CA-ACF2 user password and is not constrained by the password rules or password interval values that might be established for CA-ACF2 user passwords.

MAXTKTLF(*ticketlife*)
Maximum ticket life for this realm in seconds. This field is only valid for the local realm. If it is defined then both DEFTKTLF and MINTKTLF must also be defined. The range of values for this field is 1 to 2,147,483,647 seconds.

MINTKTLF(*ticketlife*)
Minimum ticket life for this realm in seconds. This field is only valid for the local realm. If it is defined then both DEFTKTLF and MAXTKTLF must also be defined. The range of values for this field is 1 to 2,147,483,647 seconds.

REALM(*realmname*)
REALM defines the name of the realm. The local realm must be an unqualified name. For example, REALM1.CA.COM. When defining the name of the local realm, this field has a maximum length of 117 characters.

For a foreign realm, the field must contain a fully qualified realm name.

```
/…/realm1/KRBTGT/realm2
```

When defining a foreign realm, this field has a maximum length of 240 characters.

The name assigned to the local realm limits the length of the local principal names, since fully qualified principal names cannot exceed 240 characters.

```
/…/realm_name/principal_name
```

The local realm name defined to eTrust CA-ACF2 can contain any character except the / (X'61) character. It is highly recommended that you avoid using any EBCDIC variant characters to prevent problems with code pages. This field is folded into uppercase regardless of the case.

Because of the relationship between the realm name and local principal name where the length of a fully qualified name cannot exceed 240 characters, caution and planning must go into naming the local realm since the combined length is only checked by eTrust CA-ACF2 when a local principal is added or altered. Renaming the realm should be avoided. If the name of the realm does change, the user's keys will become unusable.

Display the GSO REALM Records

Display the Kerberos REALM definitions defined in the system with SHOW REALM or SHOW ALL.

REALM.SAFDFLT

The SAF realm is defined with a qualifier of SAFDFLT.

Specify a SAF realm if applications require a mechanism to distinguish between Kerberos and SAF authentication systems. Define the eTrust CA-ACF2 databases as a SAF realm by defining the REALM.SAFDFLT record. The only valid field you may specify for the REALM.SAFDFLT record is the REALM field. Use the REALM.SAFDFLT record and the REALM field to define a SAF realm name for the eTrust CA-ACF2 databases. For example you may define the following SAF realm record:

```
Insert REALM.SAFDFLT REALM('CA-ACF2_saf_realm')
```

The user cannot set the DES, DES3, or DESD encryption values for REALM.SAFDFLT records. However, whenever a REALM.SAFDFLT record is inserted, changed, deleted, or listed, the echoed values for the record will show characteristics NODES, NODES3, and NODESD.

Resident Resource Rule Directories (RESDIR)

The RESDIR record indicates which resource rule directories are built and made globally resident at eTrust CA-ACF2 initialization. The RESDIR record also indicates whether the resource rule sets associated with those directories are made resident and whether the residency is global (in common storage) or local (in an address space).

If you are using the RESDIR record, we recommend that you migrate to use of the INFODIR record. The INFODIR record provides greater flexibility by enabling you to specify a resource class in addition to the code and type. For more information, see the discussion about the GSO INFODIR record earlier in this chapter.

Record ID	Fields
RESDIR	TYPES(*code-type1,…,code-type256*)

Field Descriptions

TYPES(*code-type1,...,code-type256*)

Code indicates whether the resource rule set associated with the directory is made resident, and whether that residency is global or local. The three possible codes are:

- **T**—rule sets are transient and never are made resident.
- **D**—rule sets are made resident locally in the user's address space when access to a resource is attempted rather than being made resident in global storage at eTrust CA-ACF2 initialization time. Locally resident rules are not affected by the F ACF2,REBUILD command. To clear locally resident rules issue the F ACF2,SETNORUL(address space) command, or the address space must come down and back up (or the user must sign off and sign on again.) The rules are reloaded when access to the resource is attempted.
- **R**—rule sets are made resident in global storage at eTrust CA-ACF2 initialization time.

We recommend specifying **R** for the code; specifying **T** or **D** could affect performance since additional I/O to the infostorage database is required for transient or on-demand processing.

Type specifies the pertinent resource rule sets for which a directory is built and made globally resident at eTrust CA-ACF2 initialization. The *type* is the same three-character type code used in the resource rule $TYPE control statement.

For example, you can code the following RESDIR record:

```
TYPES(R-CKC,D-ITR,T-CFC)
```

Where:

- **R-CKC** specifies that a resource rule directory is built and loaded into common storage for resource rules of TYPE(CKC), and the rules themselves are made globally resident.
- **D-ITC** specifies that a resource rule directory is built and loaded into common storage for resource rules of TYPE(ITR) and the rules themselves are made resident in an address space upon request.
- **T-CFC** specifies that a resource rule directory is built and loaded into common storage for resource rules of TYPE(CFC), but the rules themselves are never made resident in common storage nor in an address space, but are loaded and unloaded into an address space on a transient basis.

Regardless if a given type of resource rule is resident on demand, resident in global storage, or transient, a directory is built for each type of resource rule specified in the RESDIR option and the directory itself is always made resident.

Resident directories are rebuilt at each IPL, at each restart of eTrust CA-ACF2, or when the console operator enters the following command:

```
F ACF2,REBUILD(type)
```

To use masks (asterisks or dashes in specifications) for resource names, you must make the directory for the given type resident.

There is no limitation on the number of $KEY entries (resource names) permitted in a directory.

Storage utilization is improved when common rules are made globally resident. For example, when 100 users access the same resource, 100 copies of that rule are in LSQA unless the rule is made resident.

Displaying Resource Types Specified in the GSO RESDIR Record

If your site wants to use resident directories for resource rules, you can specify a total of 256 resource rule types. The SHOW RESIDENT and SHOW ACF2 subcommands of the ACF command display the resource types as specified in the RESDIR record.

Resident Rule Index List (RESRULE)

The RESRULE record defines a set of high-level indexes that identify the access rule sets to be made resident in storage at eTrust CA-ACF2 initialization time. You can use this function to reduce the I/O operations required by eTrust CA-ACF2 to obtain heavily used indexes such as SYS1.

Record ID	Fields
RESRULE	INDEX(*index1,…,index255*)

Field Descriptions

INDEX(*index1,...,index255*)
You can use any number of RESRULE index fields up to a maximum of 255 high-level indexes.

Once you make an index resident, changes to its rule set do not take effect until the next IPL, the next restart of eTrust CA-ACF2, or until the console operator enters the following command:

```
F ACF2,RELOAD(index)
```

We recommend that you do not make rules resident until after you stabilize rule modifications.

Storage utilization is improved when common rules are made globally resident. For example, when 100 users access SYS1 data sets, then 100 copies of the SYS1 rule are in LSQA unless the SYS1 rule is made resident.

Displaying High-Level Indexes for Rule Sets Specified in the GSO RESRULE Record

The SHOW RESIDENT and SHOW ACF2 subcommands of the ACF command display the high-level index of each rule set specified in the RESRULE record.

Data Set Level Protection Volume List (RESVOLS)

The RESVOLS record defines DASD and mass storage volumes for which eTrust CA-ACF2 is to provide protection at the data set name level.

Record ID	Fields
RESVOLS	VOLMASK(-|*mask1,…,mask255*)

Field Descriptions

VOLMASK(-|*mask1,...,mask255*)
Specifies up to 255 volume serial masks up to six characters each. Two symbols can be used in RESVOLS and SECVOLS to signify masking, the asterisk (*) and the dash (-). A dash represents all valid volumes that begin with the specified characters that precede the dash or all volumes if the dash is used alone. An asterisk represents one or more masking or wild card characters that can be specified anywhere in the in RESVOLS and SECVOLS.

Since eTrust CA-ACF2 is shipped with a RESVOLVS value of VOLMASK(-), all DASD volumes are protected by default at the data set name level.

If you alter this default setting, you must specify each DASD volume in either RESVOLS or SECVOLS to ensure the data is secure.

eTrust CA-ACF2 provides a default RESVOLS record. You can specify up to 255 volume masks.

All catalog and uncatalog functions involving volumes specified in RESVOLS are treated as DASD regardless of where the data set actually resides. This means that you can catalog and uncatalog tape data sets.

Displaying GSO RESVOLS Record Information

If you specify a DASD volume in both RESVOLS and SECVOLS, eTrust CA-ACF2 ignores the SECVOLS entry. The SHOW STATE and SHOW ACF2 subcommands of the ACF command display the volume serial number of all volumes specified in the RESVOLS record.

Reserved Word Prefix List (RESWORD)

The RESWORD record defines the words or prefixes that are not allowed in the specification of a password. In the case of mixed-case passwords, the password is temporarily upper-case before being checked against the RESWORD prefixes.

Record ID	Fields
RESWORD	PREFIXES(APPL, APR, AUG, ASDF, BASIC, CADAM, DEC, DEMO, FEB, FOCUS, GAME, IBM, JAN, JUL, JUN, LOG, MAR, MAY, NET, NEW, NOV, OCT, PASS, ROS, SEP, SIGN, SYS, TEST, TSO, VALID, VTAM, XXX, 1234 | *prefix1,…,prefix255*)

Field Descriptions

PREFIXES

Specifies up to 256 password prefixes. Specify from one to eight characters. eTrust CA-ACF2 provides a default RESWORD record. You can change the values or supply others up to a total of 256. The list is sorted in collating sequence for ease of display.

Valid prefixes for the RESWORD list can include acronyms for the company or industry, names of software systems, terminal ID prefixes, and so forth.

Following is the default reserved word prefix list:

APPL	IBM	PASS
APR	JAN	ROS
AUG	JUL	SEP
ASDF	JUN	SIGN
BASIC	LOG	SYS
CADAM	MAR	TEST
DEC	MAY	TSO
DEMO	NET	VALID
FEB	NEW	VTAM
FOCUS	NOV	XXX
GAME	OCT	1234

Displaying Prefixes Defined in the GSO RESWORD Record

The SHOW RSVWORDS and SHOW ACF2 subcommands of the ACF command display all prefixes defined in the RESWORD record.

eTrust CA-ACF2 Rule Option Specifications (RULEOPTS)

The RULEOPTS record defines the options that determine how resource and access rules are used and maintained.

RULEOPTS is like any other GSO record in that if one is not found with a matching SYSID during the system IPL, a default record is built dynamically by eTrust CA-ACF2.

Record ID	Fields
RULEOPTS	CENTRAL | NOCENTRAL CHANGE | NOCHANGE COMPDYN | NOCOMPDYN DECOMP(SECURITY,AUDIT | *privilegelist*) $NOSORT \| NO$NOSORT RULELNG | NORULELNG VOLRULE | NOVOLRULE

Field Descriptions

CENTRAL | NOCENTRAL
Specifies whether the data owner has authority to store a set of access rules. By specifying CENTRAL, only security administrators and users authorized by the %CHANGE or %RCHANGE feature have this capability. You can combine the NO-STORE field of the logonid record and NOCENTRAL to give only selected users the ability to update their own rules. The default is NOCENTRAL. That is, all users are able to update the access rule sets that correspond to the data sets they own.

CHANGE | NOCHANGE
Specifies whether the rule features, %CHANGE and %RCHANGE, are recognized. If you specify NOCHANGE, eTrust CA-ACF2 ignores any %CHANGE or %RCHANGE control statement in a rule set when it determines whether a user has the authority to replace a rule. The default value is CHANGE, which activates %CHANGE and %RCHANGE authorization.

COMPDYN | NOCOMPDYN

Specifies whether to compile a rule set with the 32K compiler. With this option, eTrust CA-ACF2 defaults to use the 4K rule format compiler and will dynamically switch to use the 32K compiler if the 4K compiler fails due to an out-of-buffer condition. For example, the rule set is too large to fit in a 4K buffer. The default is NOCOMPDYN. **Note:** If COMPDYN is specified, then the $NORULELNG control statement is not needed to compile rule sets of varying size.

DECOMP(SECURITY,AUDIT | *privilegelist*)

Specifies the attributes a logonid needs in order to be able to decompile an access or resource rule. If the SECURITY or AUDIT privilege is not specified in the DECOMP field, logonids with these privileges can still list rules unless limited by a scope.

$NOSORT | NO$NOSORT

Specifies whether the rule set's $NOSORT control statement is processed. If you specify the $NOSORT option, eTrust CA-ACF2 recognizes the $NOSORT control statement during rule compilation. During rule compilations, the $NOSORT control statement suppresses the normal eTrust CA-ACF2 sorting of rules from most specific to most general. If you specify the default option of NO$NOSORT, eTrust CA-ACF2 ignores any $NOSORT control statements during rule compilation, and automatically sorts rule sets. We recommend that you use the default, NO$NOSORT.

RULELNG | NORULELNG

Specifies whether you want to use rules greater than 4K in length. NORULELONG, the default, indicates that rules are compiled and stored in the current format with a limit of 4K. RULELONG indicates a formatted access and resource rule record capable of expanding greater than 4K to a maximum of 32K. The Rules and Infostorage databases must be redefined to accommodate this greater length.

This option is disabled if the record size of these databases is not increased. Message ACF7A450 is displayed during eTrust CA-ACF2 startup or during a refresh of the RULEOPTS record if this is the case. Once this option is set, the Rules and Infostorage databases are no longer compatible with previous versions of eTrust CA-ACF2. See the DEFINE job in the SAMPJCL library for information on increasing the size of the records in the databases. **Note:** If the dynamic compile option (COMPDYN) is set in the GSO RULEOPTS record then the $NORULELNG control statement is not needed to compile rule sets of varying size.

VOLRULE | NOVOLRULE

Indicates the format used whenever eTrust CA-ACF2 creates a pseudo data set name for volume-level protected access validations (SECVOLS). All pseudo data set names, such as Z999999, will be named according to the setting of this option. If you specify VOLRULE, the pseudo data set name format is VOLUME.@*volser*. If you specify NOVOLRULE, the format is @*volser*.VOLUME.

Displaying GSO RULEOPTS Record Information

SHOW STATE displays information for the RULEOPTS record.

Environments for SAF Calls (SAFDEF)

The SAFDEF record defines the SAF environment and how you want eTrust CA-ACF2 to process a SAF call. eTrust CA-ACF2 provides internal SAFDEFs for SAF default protection. Both internal and external SAFDEFs display when you issue a SHOW SAFDEF command. For a list of SAF resource classes, see Appendix B, "IBM-Supplied Resource Classes."

You can use the SAFDEF record to override how eTrust CA-ACF2 processes SAF calls. eTrust CA-ACF2 performs validation based on the environment you define in this record. You can create multiple SAFDEF records.

Record ID	Fields
SAFDEF*qual*	ID(*name*) FUNCRET(4 | *retcode*) FUNCRSN(0 | *rsncode*) JOBNAME(mask | ********) MODE(IGNORE | GLOBAL | LOG | QUIET) NOAPFCHK | NONOAPFCHK PROGRAM(mask | ********) RACROUTE(keyword=value,…,keyword=value) RB(mask | ********) RETCODE(0 | 4 | 8) USERID(mask | ********)

Field Descriptions

ID(*name*)

Specifies an ID name associated with the record. You can specify up to eight characters. This field is optional. We recommend you specify an ID because this name will appear as the first field displayed in the SHOW SAFDEF output. The ID is also the key used for the SHOW SAFDEF subcommand.

Select a name that is unique and that conveys meaning about the type of SAF call you are defining. For example, VERSMS would be an appropriate ID for a SAFDEF record that defines the environment for a REQUEST=VERIFY call from DFSMS.

FUNCRET(4|*retcode*)
Specifies the SAF function-dependent return code to be returned to the caller making the RACROUTE request when MODE is specified as IGNORE. For detailed descriptions of these return codes, see the IBM document entitled *z/OS Security Server (RACROUTE) Macro Reference,* No. SA22-7692. The default is four.

FUNCRSN(0|*rsncode*)
Specifies the SAF function-dependent reason code to be returned to the caller making the RACROUTE request when MODE is specified as IGNORE. For detailed descriptions of these reason codes, see the IBM document entitled IBM document entitled *z/OS Security Server (RACROUTE) Macro Reference,* No. SA22-7692. The default is zero.

JOBNAME(mask|******)**
Specifies the job names of the address spaces that apply to this SAFDEF record. You can specify an eight-character job name or a mask. The default is all job names.

MODE(IGNORE|GLOBAL|LOG|QUIET)
Specifies the mode you want eTrust CA-ACF2 to use to process this SAF request. This field defaults to GLOBAL; a value is required. You can specify any one of the following values. **Note:** Be aware that LOG and QUIET are only valid for REQUEST=AUTH calls.

- **IGNORE**—Bypass processing this SAF request
- **GLOBAL**—Process this SAF request with the mode specified in the GSO OPTS record. For generalized resource validations, use the eTrust CA-ACF2 SVCA recommendation to allow or deny the SAF request.
- **LOG**—Process this REQUEST=AUTH call in LOG mode. Upon return of the validation call, allow access even if access is denied. LOG does not force logging if a logonid is allowed access.
- **QUIET**—Process this REQUEST=AUTH call in QUIET mode.

NOAPFCHK|NONOAPFCHK
STATUS=ACCESS is a keyword used in the RACROUTE REQUEST=AUTH security macro. It permits a user to interrogate security definitions (access and resource rules) to determine the access level for a user. No auditing is performed.

To maintain system integrity, eTrust CA-ACF2 requires that a user be APF-authorized to access security definitions; however, some products that use STATUS=ACCESS are not APF-authorized when they issue the request. The result is that eTrust CA-ACF2 abends the task with a S047 from ACF9C000.

To accommodate these products, eTrust CA-ACF2 lets the security administrator define the specific calls for which the authorization check for STATUS=ACCESS will be bypassed. This is done with the NOAPFCHK keyword on a SAFDEF record that describes the specific environment from which this call is made. For example:

```
INSERT SAFDEF.apf PROGRAM(pgmname) RB(pgmname) NOAPFCHK
       RACROUTE(REQUEST=AUTH,CLASS=DATASET,STATUS=ACCESS)
```

Users who do not want to use this method should contact the vendor of the product and request that the STATUS=ACCESS call be made in an APF-authorized environment.

Note: This field cannot be copied from a model SAFDEF using an INSERT USING command. If the new SAFDEF record requires this field it must be specified as an additional field following the newrecid, or using a separate CHANGE command against the new SAFDEF record.

PROGRAM(mask|******)**

Specifies the program name of the current program request block (PRB) making the SAF request. If no PRB exists on the active RB chain when the event occurs, the name for PROGRAM is the same as the name for RB. You can specify an eight-character program name or a mask. The default is all programs.

RACROUTE(*Keyword=value,...,Keyword=value*)

Identifies the SAF request being made. Use this field to specify any valid RACROUTE parameters and values. This is a multi-value field. The maximum length that you can specify for the parameter keyword, operator, and value is 64 characters. Separate the entries with commas or blanks.

Note: There are SAFDEF restrictions with FASTAUTH processing. FASTAUTH does not allow the use of ENTITY on the RACROUTE field.

For those RACROUTE macros that permit the use of the ENTITY keyword, an actual entity name of JES2.CANCEL.STC would be specified as ENTITY=JES2.CANCEL.STC **not** as ENTITY='JES2.CANCEL.STC'.

You can specify the following relational operators (depending on your type of keyboard) to indicate the presence of a particular value (for example, ENVIR=CREATE) or the presence of a pointer address (ACEE=>).

= Equal to

¬= Not equal to

<> Not equal to

!= Not equal to

=> Pointer value

¬=> No pointer value

!=> No pointer value.

Pointer values are valid only if the keyword operand is specified as a pointer to a data area or data structure (for example, ACEE). When you specify a pointer value, do not also specify a value for the operand. For example, the following request defines a VERIFY request for all user IDs except JOHN, where an ACEE is supplied:

```
RACROUTE(REQUEST=VERIFY,ACEE=>,USERID¬=JOHN)
```

You can mask character data types using the standard eTrust CA-ACF2 masking characters (asterisk and dash). You can mask other types of data only if the mask is complete. A complete mask indicates that the parameter matches all values. For example, you can specify the following to indicate that this parameter matches all values of USERID:

```
USERID=-
```

Whereas, the following indicates that the USERID option does not apply to this RACROUTE request.

```
USERID¬=-
```

Follow all RACROUTE macro coding conventions as described in the IBM publication *z/OS Security Server RACROUTE Macro Reference,* SA22-7692.

RB(mask|******)**

Specifies the name of the request block (RB) where the security event occurs. When an event occurs directly under a PRB, you should specify the value for PROGRAM. When an event occurs under a supervisor call request block, specify the RB name as SVC*nnn*, where *nnn* is the decimal SVC number.

You can specify an eight-character RB name or a mask. The default is all request blocks.

RETCODE(0|4|8)

Specifies the SAF return code to be returned to the caller making the RACROUTE request when MODE is specified as IGNORE.

- **0**—allow the request.
- **4**—let the caller decide how to process the request.
- **8**—deny the request.

The default is 4.

USERID(*useridmask*|******)**

Specifies the user ID of the address spaces that apply to this SAFDEF record. The default is all address spaces.

Creating Multiple GSO SAFDEF Records

To create multiple SAFDEF records, append a qualifier to the record name in the format SAFDEF*qual* so that you can define a unique record ID for that SAFDEF record for a particular SYSID. The RECID can be a maximum of 16 bytes. Therefore, you can specify a qualifier of up to ten characters. It must immediately follow the characters SAFDEF. If you use a period (.) as part of the qualifier string for the record name, eTrust CA-ACF2 counts it as one of the ten available characters.

For example, if you want to create a SAFDEF record for a VERIFY call from HSM and an AUTH call from HSM, you must use a qualifier to distinguish these two records. You could name the SAFDEF record for the VERIFY call SAFDEF.VERHSM and the SAFDEF record for the AUTH call SAFDEF.AUTHHSM. Naming records using qualifiers lets you describe multiple unique environments and lets eTrust CA-ACF2 add those records to the Infostorage database with unique identifiers.

eTrust CA-ACF2 processes all SAF calls by default. To override the default SAF processing for a specific security event, you can specify a SAFDEF record.

Displaying GSO SAFDEF Record Information

The SHOW SAFDEF and SHOW ALL subcommands display the values in the SAFDEF record. The SHOW SAFDEF output displays the following information:

```
JESPOOLR   JOBNAME=********   USERID=********   PROGRAM=HA$PSUBS   RB=HA$PSUBS
           RETCODE=0           SAFDEF=INTERNAL   MODE=IGNORE        SUBSYS=ACF2
```

Here is how to interpret the output of the SHOW SAFDEF command, from left to right:

JESPOOLR
The ID field of the SAFDEF record is defined as ID(JESPOOLR).

JOBNAME=******
The JOBNAME field of the SAFDEF record was not specified so eTrust CA-ACF2 used the default, all jobs.

USERID=******
The USERID field of the SAFDEF record was not specified so eTrust CA-ACF2 used the default, all user IDs.

PROGRAM=HA$PSUBS
The PROGRAM field of the SAFDEF record is defined as PROGRAM(HA$PSUBS).

RB=HA$PSUBS
The request block name that the security event occurs in.

RETCODE=0
The RETCODE field of the SAFDEF record is defined as RETCODE(0) so that eTrust CA-ACF2 allows the request when MODE=IGNORE is specified.

SAFDEF=INTERNAL
This field indicates that this SAFDEF record was created by Computer Associates. If this record was created by your site in a SAFDEF record, the display would read SAFDEF=GSO.

MODE=IGNORE
The MODE field of the SAFDEF record is defined as MODE(IGNORE) so that eTrust CA-ACF2 bypasses processing of this request.

SUBSYS=ACF2
This field indicates that this record is for the eTrust CA-ACF2 subsystem. Another value that can be displayed here is MACS for the CA MAC subsystem. User-defined SAFDEF records are always targeted to all SAF processing subsystems (****).

SAFDEF Record Examples

This section contains two examples. The first example shows a simple SAFDEF record that you can create when you want eTrust CA-ACF2 to ignore validating the request. The second example shows a SAFDEF record when you want eTrust CA-ACF2 to validate the SAF request.

Ignoring SAF Calls

A basic SAFDEF record that you might have to create is shown as follows:

```
set control(gso)
 CONTROL
list like(safdef-)
 PRD1 / SAFDEF.XYZ LAST CHANGED BY USER01 ON 07/02/98-12:13
                    ID(AUTHXYZ) MODE(IGNORE) RETCODE(0)
                    PROGRAM(XYZ-) JOBNAME(XYZ-)
                    RACROUTE(REQUEST=AUTH,CLASS=DATASET)
```

In this example, the SAFDEF record is for the XYZ product. Program XYZ123 makes a SAF call that eTrust CA-ACF2 intercepts. In this case, suppose the SAF call is a RACROUTE REQUEST=AUTH,CLASS=DATASET. The site decided to instruct eTrust CA-ACF2 to bypass processing of this request because it did not want eTrust CA-ACF2 to validate these calls. It also specifies that eTrust CA-ACF2 set a return code of 0 instead of the default of 4.

By specifying MODE(IGNORE) and RETCODE(0), eTrust CA-ACF2 will let program XYZ123 access the data set without creating a logging record. The site does not have to create an access rule.

Validating SAF Calls

A more complicated example is described in the following:

```
PRD1 / SAFDEF.CATWRK001 LAST CHANGED BY USER01 ON 07/18/98-13:31
                    ID(CAT001) USERID(USER-) MODE(GLOBAL)
                    RACROUTE(REQUEST=AUTH CLASS='DATASET' DSTYPE=V ATTR=UPDATE
                    ENTITY='CATALOG.MVSICF1.VWRK001') RETCODE(4)
```

In this example, the SAFDEF record defines an environment to eTrust CA-ACF2 in which USER123 makes a SAF call to update the ICF catalog for the WRK001 volume. The site wants eTrust CA-ACF2 to process the call normally. eTrust CA-ACF2 validates the data set access according to MODE specification in the GSO OPTS record. If the site wants to reduce the number of loggings, it should create an access rule like the one shown in the following example:

```
$KEY(CATALOG)
 MVSICF1.VWRK001 UID(USER-) W(A)
 MVSICF1.VWRK001 UID(*) W(L)
```

This rule lets jobs running under a USER– logonid update the catalog without creating a logging record. If user JOHNQ tries to update the catalog, eTrust CA-ACF2 creates an SMF record of the event and still allows the update to occur. The RETCODE(4) shown in the sample SAFDEF will not be used since MODE(GLOBAL) forces a validation that will result in a return code or 0 or 8 based on the rule.

Volume Mask Volume-Level Protection (SECVOLS)

The SECVOLS record defines the DASD and tape volumes for which eTrust CA-ACF2 provides volume-level protection.

Record ID	Fields
SECVOLS	VOLMASK(-|*mask1,…,mask255*)

Field Descriptions

VOLMASK(*volmask1,...,volmask255*)

Specifies up to 255 volume serial masks up to six characters each. Two symbols can be used in RESVOLS and SECVOLS to signify masking, the asterisk (*) and the dash (-). A dash represents all valid volumes that begin with the specified characters that precede the dash or all volumes if the dash is used alone. An asterisk represents one or more masking or wild card characters that can be specified anywhere in the in RESVOLS and SECVOLS.

The default is null no volume-level protection.

Note: By default, the RESVOLS parameter is set to '********'. This indicates security at the data set level. If you alter this default setting, you must specify each DASD volume in either RESVOLS or SECVOLS to ensure the data is secure.

If you want eTrust CA-ACF2 volume-level protection, at least one volser is required. You can use additional VOLMASKS, up to a total of 255.

When you specify a volume in the SECVOLS record, eTrust CA-ACF2 generates a pseudo data set name of *@volser*.VOLUME for the purpose of validating an access request to that volume. However, if the VOLRULE field in the GSO RULEOPTS record is specified, then the pseudo data set name is generated as VOLUME.*@volser*.

If a DASD volume is specified in both RESVOLS and SECVOLS, eTrust CA-ACF2 ignores the SECVOLS entry.

Displaying GSO SECVOLS Record Information

The SHOW STATE and SHOW ACF2 subcommands of the ACF command display the volume serial number of all volumes specified in the SECVOLS record.

Cache Synchronization (SYNCOPTS)

The SYNCOPTS record defines the cache synchronization processing for a system that runs in a shared eTrust CA-ACF2 database environment.

Create a SYNCOPTS record if your site uses the cache facility in a shared eTrust CA-ACF2 database environment. For information about the cache facility, see the "Maintaining Cache Records" chapter.

Define only one SYNCOPTS record for each SYSID (system). The sysid of the SYNCOPTS record can be masked to allow it to apply to more than one system. Since all synchronizers in the same shared eTrust CA-ACF2 database environment must use the same synchronization file, you must define the same FILENAME ofr those SYNCOPTS records.

For assistance in determining the most desirable values for your site's GSO SYNCOPTS record, see "Installation Options" in the *Systems Programmer Guide*.

Record ID	Fields
SYNCOPTS	ACTIVATE | NOACTIVATE FILENAME(ACF2.SYNCFILE | *filename*) POLLINTV(5 | *nn*) USECOUNT(10 | *nnn*)

Field Descriptions

ACTIVATE|NOACTIVATE
Specifies whether this CPU's synchronizer is activated. The default is NOACTIVATE. To let eTrust CA-ACF2 activate a synchronizer, you must set the SHRDASD field of the GSO OPTS record at this CPU.

FILENAME(ACF2.SYNCFILE|*filename*)
Specifies the file that is used for cache synchronization. The default synchronization file name is ACF2.SYNCFILE; however, you can specify your own file name. The maximum number of characters permitted for the file name is 44 characters.

POLLINTV(5|*nn*)
Specifies the number of seconds that elapses between accesses performed by the synchronizer to the synchronization file. The default is 5 seconds, but any number from 1 to 60 is permitted.

USECOUNT(10|*nnn*)
Specifies the maximum number of times an entry can be read by all synchronizers before the entry is deleted by this synchronizer. The default value is 10, but you can specify any number from 1 to 255.

Displaying GSO SYNCOPTS Record Information

SHOW STATE displays whether ACTIVATE was specified in the SYNCOPTS record. SHOW ACTIVE displays whether the synchronizers associated with this SYSID are active and whether the cache facility is active.

SYSPLEX Environment and Options (SYSPLEX)

The SYSPLEX record contains the information necessary for eTrust CA-ACF2 to communicate with the SYSPLEX using the proper structure and member names. You must create a GSO(SYSPLEX) record if you use XCF or XES services.

Note: If you plan on using the SYSPLEX feature you cannot have a PPT entry for ACFMAIN with KEY(1) in the SCHED00 member of SYS1.PARMLIB.

For information about implementing the SYSPLEX feature of eTrust CA-ACF2, see the "Using the SYSPLEX Coupling Facility" chapter.

Record ID	Fields
SYSPLEX	ALTNAME( ) CONSMFID | NOCONSMFID FULLACTN(CLEAR | WARNMSG | NONE) FULLTHSH(90 | nnn) INFOSTG | NOINFOSTG LIDS | NOLIDS MINLVL80 | NOMINLVL80 PRIMNAME( ) PREFIX | NOPREFIX RULES | NORULES XCFGROUP( )

Field Descriptions

ALTNAME( )
Defines the name of the alternate structure name for use in the XES feature. This structure name is defined to the Coupling Facility and is to be used as an alternate to the primary structure. This field is sixteen characters in length.

CONSMFID | NOCONSMFID
When specified on the GSO SYSPLEX record, eTrust CA-ACF2 will use the SMFID of the system as the member name in order to connect to a SYSPLEX structure. Normally, eTrust CA-ACF2 uses the eTrust CA-ACF2 SYSID that is active when the SYSPLEX is started for the member name. The member name must be unique within an XCFGROUP. The eTrust CA-ACF2 SYSID might or might not be unique within a given XCFGROUP. This allows an installation to use the SMFID which must be unique within a SYSPLEX. Also, the member name must begin with an upper case alphabetic character or a national @,#,$) character.

FULLACTN(CLEAR|WARNMSG|NONE)

Specifies the action to take when the structure reaches the threshold specified in the FULLTHSH() field. CLEAR will cause a warning message to be issued and then the structure will be cleared. When WARNMSG is specified, a warning message only will be given. If FULLACTN(NONE) is specified, no action will be taken. There will be no warning message and the structure will not be cleared.

FULLTHSH(90|*nnn*)

Use this field to designate the structure full value. When this value is exceeded, eTrust CA-ACF2 will act according to how the FULLACTN field is set.

Note: The values specified in FULTHSH and FULLACTN effect shared storage in a SYSPLEX environment. Use caution when specifying values for these fields in all eTrust CA-ACF2 systems that are a part of the SYSPLEX.

INFOSTG|NOINFOSTG

Specifies that INFOSTG records are to be placed in the Coupling Facility. This field is only valid when using the XES feature. The default is NOINFOSTG.

LIDS|NOLIDS

Specifies that LID records are to be placed in the Coupling Facility. This field is only valid when using the XES feature. The default is NOLIDS. **Note:** You have the ability to specify for each LOGONID record whether or not it should be written to the structure. Please see OMIT bit in LOGONID record field descriptions. The default is that when LIDS is specified in the GSO SYSPLEX record, all LOGONID records will be written to the structure.

MINLVL80|NOMINLVL80

Identifies the minimum level of all connected eTrust CA-ACF2 systems that are connecting to the XES structures specified using the PRMNAME and ALTNAME fields. NOMINLVL80 is the default. This option is pertinent only if you are utilizing the eTrust CA-ACF2 sysplex data sharing (XES) facility.

Specify NOMINLVL80 if your eTrust CA-ACF2 parallel data sharing (XES) implementation utilizes one or more pre-release 8 systems. When this option is set, eTrust CA-ACF2 uses the pre-release 8 structure connection characteristics, thus enabling it to co-exist in a parallel sysplex data sharing implementation with pre-release 8 systems.

Specify MINLVL80 only if you are certain that all of the eTrust CA-ACF2 systems in your parallel sysplex data sharing implementation are at r8 or above.

Note: To activate changes to this field in the GSO SYSPLEX record the following steps must be taken:

1. Terminate or stop the eTrust CA-ACF2 Coupling Facility interface. Enter the following command on each eTrust CA-ACF2 system in the SYSPLEX:

   ```
   F ACF2,SYSPLEX(STOP)
   ```

2. Make the desired GSO SYSPLEX record field change.

3. Refresh the GSO SYSPLEX record on each eTrust CA-ACF2 system in the SYSPLEX:

   ```
   F ACF2,REFRESH(SYSPLEX)
   ```

4. Start the eTrust CA-ACF2 Coupling Facility interface (SYSPLEX). Enter the following command on each eTrust CA-ACF2 system in the SYSPLEX:

   ```
   F ACF2,SYSPLEX(START)
   ```

This option is intended as a migration aid for eTrust CA-ACF2 r8 and will be removed in the next release. Any time you are trying to connect multiple systems to a structure and at least one of the systems is a pre-release 8 eTrust CA-ACF2 system, NOMINLVL80 must be set in the GSO SYSPLEX record for each system. Otherwise the structure connect will fail with error message ACF79473 CONNECT TO C.F. FAILED -DATA SET NAME CONFLICT.

PREFIX|NOPREFIX
This option places ACF@ at the beginning of the member name to fulfill the requirement of **CONSMFID**.

PRIMNAME( )
Defines the name of the primary structure name for use in the XES feature. This structure name is defined to the Coupling Facility and is the primary structure that is to be used. This field is sixteen characters in length.

RULES|NORULES
Specifies that RULE records are to be placed in the Coupling Facility. This field is only valid when using the XES feature. The default is NORULES.

XCFGROUP( )
Specifies the name of the XCF GROUP member that the SYSPLEX uses for the XCF service. The group name specified must be the same for all systems that communicate with each other.

Starting, Stopping, and Clearing the Coupling Facility

Use the following commands to start, stop, and clear the coupling facility interface. eTrust CA-ACF2 must be active when you issue these commands:

Start

To start the eTrust CA-ACF2 Coupling Facility interface (SYSPLEX), enter the following command:

```
F ACF2,SYSPLEX(START)
```

Stop

To terminate or stop the eTrust CA-ACF2 Coupling Facility interface, enter the following command:

```
F ACF2,SYSPLEX(STOP)
```

Clear

To clear the SYSPLEX XES structure, enter the following command:

```
F ACF2,SYSPLEX(CLEAR)
```

Displaying GSO SYSPLEX Record Information

The SHOW SYSPLEX subcommand displays information pertinent to the SYSPLEX feature.

Started Task (STC)

The Started Task (STC) record assigns a logonid and optional groupid based on the started task ID. See Started Task section in the "Controlling System Entry" chapter for details on how this is used.

Record ID	Fields
STC*qual*	Group(*group*) Logonid(*lid*) Stcid(*started task name*)

Note the following:

- The GSO STC record is only used if there is no matching STC LOGONID that matches the stated task proc name.
- If there is an STCXIT present, it will get control before the GSO STC record is checked. If the LID value is altered by the exit, eTrust CA-ACF2 will look for an STC lid that matches that new value. If no STC lid is found that matches the new value, then the GSO STC record willb e checked using the new value.

Field Descriptions

GROUP(*group*)
Indicates the group that will be assigned to the started task. It overrides the group field in the logonid record. This field is optional.

LOGONID(*logonid*)
Specifies the logonid that will be assigned to the started task. This field is required and is not maskable. The logonid can, but does not, require the STC attribute. If the logonid does not have the STC attribute it can be used for other system accesses including inheritance for batch jobs submitted by the started task.

STCID(*started task name*)
Specifies the name of a started task. This field is required and is maskable.

Creating Multiple GSO STC Records

If you need more than one STC record, append a qualifier to the record name in the format STC*qual* to generate a unique record ID. For example, STCVMAN or STC.DATASET. This optional qualifier can be up to 13-characters and must immediately follow the characters STC. If you use a period (.) as part of the qualifier string for the record name, eTrust CA-ACF2 counts it as one of the 13-characters.

Displaying GSO STC Records

The SHOW STCID subcommand of the ACF command displays the contents internal definitions (eTrust CA-ACF2-defined) and external definitions (site-defined) of SAF calls that are being translated.

```
acf
show stcid
--STARTED TASK TABLE –

STCID      LOGONID      GROUP
======     =========    =======
ENF        IMWEBSRV     IMWEBSRV
FFFFFFF    TESTF
```

Unicenter TNG Node Definitions (TNGNODE)

The TNGNODE record defines the Windows NT machines that act as Unicenter TNG monitors. These nodes are sent violation messages using SNMP traps.

Record ID	Fields
TNGNODE*qual*	DESC(*description*) DEBUG|NODEBUG IPADDR(*IP-address*)

Field Descriptions

TNGNOD*Equal*
Specifies a unique name for the TNGNODE record. The qualifier can be up to nine characters and is appended to the TNGNODE stem of the record ID. If you use a period as part of the qualifier string for the record ID, eTrust CA-ACF2 counts it as one of the nine characters.

DESC(description)
Specifies up to 20 characters of data. We recommend that you use this field to enter a description of the CA-Common Services node name.

DEBUG|NODEBUG
This indicator turns on the tracing for the ENFSNMPM started task for this node. This indicator should be used only when recommended by Computer Associates' personnel. (NODEBUG is the default.)
IPADDR(*IP-address*)—Specifies the IP address of the security workstation that receives the SNMP traps.

Displaying TNG Options

The SHOW TNG and SHOW ACF2 subcommands of the ACF command display the TNGNODE options as specified in the TNGNODE record.

Time-Sharing Options and Defaults (TSO)

The TSO record specifies global usage and system parameters that define and control the TSO logon process and other system parameters.

eTrust CA-ACF2 provides a default TSO record that you can modify.

Record ID	Fields
TSO	ACCOUNT(1 | *string*) BYPASS(# | *character*) CHAR(BS | *character*) CMDLIST(*moduleid*) FSRETAIN | NOFSRETAIN IKJEFLD1 | NOIKJEFLD1 LINE(ATTN | CTLX | *character*) LOGONCK | NOLOGONCK PERFORM(0 | *nnn*) PROC(IKJACCNT | *procedure*) QLOGON | NOQLOGON REGION(2048 | *nnnn*) SUBCLSS(*class*) SUBHOLD(*class*) SUBMSGC(*class*) TIME(0 | *nnn*) TSOGNAME(name) TSOSOUT(A | *class*) UNIT(SYSDA | *unitname*) WAITIME(0 | *nnn*)

Field Descriptions

The TSO record fields are described in the following (an asterisk (*) means active only if UADS is bypassed):

***ACCOUNT(1 | *string*)**
Specifies the system-wide default TSO account number. The default value is 1. If you set this string to blanks and do not specify ACCOUNT in the logonid record, an account prompt occurs at logon regardless of the UADS setting. This string is put in parentheses when it is moved to the job statement.

BYPASS(# | *character*)
Defines the TSO command list bypass character. The default value is a pound sign (#).

***CHAR(BS | *character*)**
Defines the default TSO delete character. When entered at the terminal, this character indicates that the previous character entered should be ignored. This optional field has no default value. BS indicates that the backspace character deletes the last character entered.

CMDLIST(*moduleid*)
Specifies the default TSO command limiting list. If you specify a module, no users, even privileged logonids, can run without the command list present in a link list library. This field is optional and has no default. It is effective in all modes with the exception of QUIET.

FSRETAIN | NOFSRETAIN
Controls the retention of logon values from session to session if TSO full-screen logon is supported. NOFSRETAIN, the default, indicates that the user has to provide applicable values at each logon.

FSRETAIN specifies that eTrust CA-ACF2 gets account and procedure values from the previous session, even if you specify PMT-ACCT or PMT-PROC in the logonid record.

IKJEFLD1 | NOIKJEFLD1
Indicates that eTrust CA-ACF2 will dynamically link authorized logon pre-prompt, IKJEFLD1 into LPALIB. This lets you use the authorized logon pre-prompt exit. You must perform the eTrust CA-ACF2 REFRESH command to activate the IKJEFLD1 facility. The default is NOIKJEFLD, which indicates eTrust CA-ACF2 will not dynamically link the authrozied logon pre-prompt exit. **Note:** Once activated, an IPL is required to deactivate the IKJEFLD1 facility. For more information, see the *System Programmer's Guide*.

***LINE(ATTN | CTLX | *character*)**
Specifies the system-wide default TSO line-delete character. When entered at the terminal, this character indicates that the current line should be ignored. This optional field has no default value. ATTN indicates that an attention interruption deletes the current line. CTLX indicates that the X and CTRL keys pressed simultaneously delete the current line (for Teletype terminals).

LOGONCK | NOLOGONCK
Indicates whether eTrust CA-ACF2 checks the TSO attribute in the user's logonid record. If you specify LOGONCK and the user does not have the TSO attribute in his logonid, eTrust CA-ACF2 rejects the logon attempt. The default value is NOLOGONCK.

***PERFORM(0 | *nnn*)**
Specifies the system-wide default TSO performance group. If you specify zero, no performance group (PERFORM=) parameter is placed on the job statement. The default value is zero.

***PROC(IKJACCNT | *procedure*)**
Specifies the default TSO cataloged procedure name. Specify the default value for an individual user with the TSOPROC field of the logonid record.

QLOGON|NOQLOGON
Specifies whether a quick one-line logon is permitted. This lets eTrust CA-ACF2 accept the password specified on the first line instead of forcing a prompt. When QLOGON is in effect, password integrity can be jeopardized. The default value is QLOGON.

***REGION(2048|*nnnn*)**
Specifies the default TSO region size. The TSORGN option in the logonid record or a size specification at TSO logon time can override this value. If this field is zero and the user does not specify a region size at logon time or in the logonid record, eTrust CA-ACF2 assumes that the region has been specified in the TSO logon procedure and no value is passed by eTrust CA-ACF2 to TSO. If this field is zero, the SHOW TSO command will indicate “NONE”. The default value is 2048.

***SUBCLSS(*class*)**
Specifies the default TSO job submission class. This field is active only if TSO/E is also installed. This is an optional field and has no default value.

***SUBHOLD(*class*)**
Specifies the default submit hold class. This field is active only if TSO/E is also installed. This is an optional field and has no default value.

***SUBMSGC(*class*)**
Specifies the default submit message class. This field is active only if TSO/E is also installed. The default value is null.

***TIME(0|*nnnn*)**
Specifies the default time estimate for TSO sessions in minutes. If you specify zero, no TIME parameter is placed on the job statement. The default value is zero. The maximum value is 1439.

TSOGNAME(*name*)
Specifies a one to eight character value for the application name used at TSO logon for PassTicket validation. If TSOGNAME is not specified, the application name will default to "TSO" followed by the four character SMF SYSID of the system. The default value of TSOGNAME is null.

TSOSOUT(A|*class*)
Specifies the default class for spun TSO SYSOUT. This field is active only if TSO/E is also installed. The default value is A.

***UNIT(SYSDA|*unit name*)**
Specifies the default UNITNAME used in TSO allocation requests. The default value is SYSDA.

WAITIME(0|*nnn*)
Specifies whether eTrust CA-ACF2 should time user responses. If you specify a nonzero value, that is the amount of time permitted between prompts. eTrust CA-ACF2 aborts the logon if the user exceeds wait time. The value you specify as *nnn* must be less than or equal to 120 seconds. The default value is zero (no check takes place).

Displaying TSO Options

The SHOW TSO and SHOW ACF2 subcommands of the ACF command display the TSO options as specified in the TSO record.

ASCII CRT Clear String (TSOCRT)

The TSOCRT record defines a clear string used to obliterate the logon to ASCII CRT devices.

Record ID	Fields
TSOCRT	STRING(A12FA11C1A270C0D | *hhhhhhhhhhhhh...h*)

Field Descriptions

STRING(A12FA11C1A270C0D | *hhhhhhhhhhhhh...h*)
Specifies a one to 256-byte CRT clear string, in hexadecimal. The default is A12FA11C1A270C0D.

Displaying GSO TSOCRT Record Information

This value is not displayed by the SHOW subcommand. However, as with other GSO records, you can list this record with the LIST subcommand under the CONTROL(GSO) setting.

User Logon Keywords (TSOKEYS)

The TSOKEYS record defines site-supplied keywords that eTrust CA-ACF2 permits at TSO logon time. eTrust CA-ACF2 does nothing with these keywords except to pass them on for local processing.

Record ID	Fields
TSOKEYS	KEYWORDS(*keyword1,…,keyword256*)

Field Descriptions

KEYWORDS(*keyword1,...,keyword256*)
Specifies up to 256 eight-character keywords that your site wants eTrust CA-ACF2 to recognize as valid at TSO logon time.

Displaying GSO TSOKEYS Record Information

The LIST subcommand of the ACF command displays the keywords from the TSOKEYS record.

TWX X-Out String (TSOTWX)

The TSOTWX record defines a cross-out mask used to obliterate the logon password on TWX devices.

Record ID	Fields
TSOTWX	CR(15 | *hhhh*) IDLE(17 | *nn*) LENGTH(8 | *nn*) M1(X | *c*) M2(N | *c*) M3(Z | *c*) M4(M | *c*) STRING(*hhhhhhhhhh...h*)

Field Descriptions

CR(15 | *hhhh*)
Specifies the carriage return character in hexadecimal. Acceptable values are 15, OD, or OD15. The default value is 15.

IDLE(17 | *nn*)
Specifies the TWX idle character in hexadecimal. The default value is 17.

LENGTH(8 | *nn*)
Specifies the length of the cross-out mask. Acceptable values are 8 or 17 bytes. The default value is 8.

M1(X | *c*)
Specifies the first mask character. The default value is X.

M2(N | *c*)
Specifies the second mask character. The default value is N.

M3(Z | *c*)
Specifies the third mask character. The default value is Z.

M4(M | *c*)
Specifies the fourth mask character. The default value is M.

STRING(*hhhhhhhhhh...h*)
Specifies a one to 256-character cross-out string in hexadecimal. The default is a null string, which causes the string to be built from values specified in the other fields of the TSOTWX record.

Displaying GSO TSOTWX Record Information

This value is not displayed by the SHOW subcommand. However, as with other GSO records, you can list this record with the LIST subcommand under the CONTROL(GSO) setting.

2741 X-Out Mask (TSO2741)

The TSO2741 record defines a cross-out string used to obliterate the logon password on 2741 devices.

Record ID	Fields
TSO2741	BS(16 | *nn*) LENGTH(8 | *nn*) M1(X | *c*) M2(N | *c*) M3(Z | *c*) M4(M | *c*) STRING(*hhhhhhhhhh...h*)

Field Descriptions

BS(16 | *nn*)
Specifies the backspace character. The default value is 16.

LENGTH(8 | *nn*)
Specifies the length of the cross-out mask. Acceptable values are 8 or 17. The default value is 8.

M1(X | *c*)
Specifies the first mask character. The default value is X.

M2(N | *c*)
Specifies the second mask character. The default value is N.

M3(Z | *c*)
Specifies the third mask character. The default value is Z.

M4(M | *c*)
Specifies the fourth mask character. The default value is M.

STRING(*hhhhhhhhhh...h*)
Specifies a one to 256-character cross-out string in hexadecimal. The default is a null string, which causes the string to be built from values specified in the other fields of the TSO2741 record.

Displaying GSO TSO2741 Record Information

This value is not displayed by the SHOW subcommand. However, as with other GSO records, you can list this record with the LIST subcommand under the CONTROL(GSO) setting.

Unix System Services Options (UNIXOPTS)

The UNIXOPTS record defines the system options related to UNIX System Services.

Record ID	Fields
UNIXOPTS	CHOWNRES | NOCHOWNRES DFTGROUP(*defaultgroup*) DFTUSER(*defaultuser*) DIRACC | NODIRACC DIRSRCH | NODIRSRCH FSOBJ | NOFSOBJ FSSEC | NOFSSEC GOSETGID | NOGOSETGID HFSACL | NOHFSACL HFSSEC | NOHFSSEC IPCOBJ | NOIPCOBJ NGROUPS(nn) PROCACT | NOPROCACT PROCESS | NOPROCESS

Field Descriptions

CHOWNRES | NOCHOWNRES
CHOWNRES implements POSIX CHOWN RESTRICTED which states that only the super user can modify the owner (UID) of a file. NOCHOWNRES implements POSIX CHOWN UNRESTRICTED which lets the current owner modify the owning UID of a file.

DFTGROUP(*defaultgroup*)
Specifies the name of the default group used by UNIX System Services (OMVS) if a user does not have a valid OMVS group in the logonid record.

DFTUSER(*defaultuser*)
Specifies the name of the logonid and OMVS user profile record name that defines the defaults for UNIX System Services (OMVS). If a user accesses OMVS services and does not have an OMVS user profile record, the defaults defined in this ID are used. If a user has NO-OMVS defined in his logonid, the user cannot use OMVS services and the default is not used. This is equivalent to specifying NOUID in RACF.

DIRACC | NODIRACC
Specifies if SMF records are to be cut for UNIX system services that control access checks for read/write access to directories. Some of the functions that access directories with read or write access are open, opendir, rename, rmdir, mount, mkdir, link, mknod, getcwd, and vlink. The Security Server callable services that control cutting this SMF record are ck_access and ck_owner_2_files.

DIRSRCH | NODIRSRCH
Specifies if SMF records are to be cut for UNIX system services that control directory searches. Some of the functions that search directories are chmod, chown, chaudit, getcwd, link, mkdir, open, opendir, stat, ttyname and vlink. The Security Server callable service that controls cutting this SMF record is ck_access. Be aware that auditing directory searches will generate an extremely large amount of SMF records in a short period of time.

FSOBJ | NOFSOBJ
Specifies if SMF records are to be cut for UNIX system services that control the auditing of the creation and deletion of system objects. It also cuts SMF records for all access checks except directory searches. Some of the functions that will do this are chdir, link, mkdir, open, mount, rename, rmdir, symlink, vmakedir, and vcreate. The Security Server callable services that control cutting of this SMF record are ck_access, ck_owner_2_files, ckpriv, makeFSP, make_root_FSP, makeISP, and R_audit.

FSSEC|NOFSSEC
Specifies if SMF records are to be cut for UNIX system services that control the auditing of changes to the security data (FSP) for file system objects. Some of the functions that modify the FSP are chaudit, chmod, chown, chattr, write, fchaudit, fchmod, and setfacl. The Security Server callable services that control cutting of this SMF record are R_chaudit, R_chown, R_chmod, clear_setid, and R_setfacl.

GOSETGID|NOGOSETGID
This option alters the way the makeFSP SAF callable service works. If GOSETGID (Group Owner SETGID) is set and a new directory is being created, the new directory will inherit the S_ISGID setting from the parent directory. Otherwise, the bit is set to zero. When a file or directory is being created, the owning GID of the new file is normally set to that of the parent directory. If GOSETGID is set and the parent's set-gid bit is off, then the owning GID of the new file or directory is set to the effective GID of the process.

HFSACL|NOHFSACL
When HFSACL is specified, Access Control Lists are used in the z/OS UNIX security access validation process in addition to the checking of file permission bits and superuser status. When NOHFSACL is specified, normal z/OS UNIX security access validation is done including the checking of file permission bits and superuser status. Access Control Lists (ACLs) are supported in z/OS release 1.3 and above. If HFSSEC is also specified, ACLs are not used regardless of the setting of this field. See Controlling Access to the Hierarchical File System for more information on Access Control Lists.

HFSSEC|NOHFSSEC
When HFSSEC is specified, CA SAF HFS security is activated. Normal z/OS UNIX security access validation is bypassed. This includes checking of file permission bits, superuser status and normal z/OS UNIX Security Services.

When NOHFSSEC is specified, CA SAF HFS security is not active. Normal z/OS UNIX sercurity access validation is enabled. This includes checking of file permission bits, superuser status and normal z/OS UNIX Security Services. NOHFSSEC is the default.

See Controlling Access to the Hierarchical File System for more information on CA SAF HFS security.

CA SAF HFS security can also be controlled using the F ACF2,HFS (STATUS| ENABLE|DISABLE) command. See the *Systems Programmer's Guide* for more information.

IPCOBJ|NOIPCOBJ
Specifies if SMF records are to be cut for UNIX system services that control the auditing of the access control, IPC object changes and the creation and deletion of IPC objects. Some of the functions that will do this are msgctl, msgget, msgsnd, semctl, semget, semop, shmat, shmget and shmctl. The Security Server callable services that control cutting of this SMF record are ck_IPC_access, R_IPC_ctl, and makeISP.

NGROUPS(ngroups_max)
The maximum number of the supplemental groups supported. The maximum value that can be specified is 8192. The default is 300. Increasing NGROUPS to a value higher than 300 is not recommended because IBM's MVS UNIX processing restricts the NGROUPS_MAX variable to 300 in accordance with POSIX standards.

PROCACT|NOPROCACT
Specifies if SMF records are to be cut for UNIX system services that control the auditing of services that look at data from or affect other processes. Some of the functions that effect other processes are getpsent, kill, ptrace, recv, recvmsg and sendmsg. The Security Server callable services that control cutting of this SMF record are ck_process_owner and R_ptrace.

PROCESS|NOPROCESS
Specifies if SMF records are to be cut for UNIX system services that control the dubbing and undubbing of processes, changes to the UIDs and GIDs of processes, and changes to the thread limits and other privileged options. Some of the functions that dub processes or change process values are exec, setuid, setgid, seteuid, setegid, dub, undub, and vregister. The Security Server callable services that control cutting of this SMF record are R_exec, R_setuid, R_setgid, R_seteuid, R_setegid, ck_priv, initACEE, initUSP, and deleteUSP.

System WARN Mode Message (WARN)

The WARN record specifies text of a warning message to be displayed on the terminal and job log when a violation takes place and the eTrust CA-ACF2 system is in WARN mode.

eTrust CA-ACF2 provides one default WARN record per system.

Record ID	Fields
WARN	MSG('*messagetext*')

Field Descriptions

MSG('*messagetext*')
Specifies a text string, up to 124 characters, enclosed in single quotes. The default warning string is:

```
AFTER JULY 1, 1999 THIS ACCESS WILL NOT BE ALLOWED
```

The disposition of the WARN message is dependent upon the CONSOLE field of the GSO OPTS record.

Displaying GSO WARN Record Information

This value is not displayed by the SHOW subcommand. However, as with other GSO records, you can list this record with the LIST subcommand under the CONTROL(GSO) setting.

Relationship Among LOGPGM, MAINT, and PPGM

You can define which programs must be executed by privileged logonids using the MAINT, LOGPGM, or PPGM record in GSO. You must carefully select the programs on these records to avoid excessive loggings or the absence of adequate audit trails. Programs listed on the LOGPGM record generate a logging each time the program accesses a data set, regardless of any trace or violation. The programs on this record are secured by normal access rules and they are generally not run very frequently. The PPGM and MAINT records can list the names of programs that must be run by privileged users. The programs listed on these two records are generally executed frequently, making it desirable to partially or totally suppress loggings.

This diagram shows relationships among the GSO LOGPGM, MAINT, and PPGM records.

GSO Record	Logonid Attributes	Access Rule Validation	SMF Loggings
LOGPGM	N/A	Yes	All data set accesses
MAINT	MAINT or NON-CNCL	Bypassed	None
PPGM	PPGM, unscoped SECURITY, or NON-CNCL	Yes	At step initiation and at every data set open

The MAINT record requires that the program, the library in which it resides, and the logonid that executes it, all be included in this record.

TSO Full-Screen Logon Retention Records

The eTrust CA-ACF2 TSO preprompt exit (IKJEFLD) maintains the CONTROL(TSO) records internally. These records retain the logon parameters specified during the most recent TSO logon. When the full-screen retention facility is active, eTrust CA-ACF2 assumes that the logon parameters desired are those specified in the user's logonid record. When those parameters differ from the logonid record, a retention record is created in the Infostorage database.

Use of the logon parameter retention facility requires that:

- Full-screen TSO logon is selected in the individual logonid record. The TSOFSCRN field selects the full-screen logon.
- The FSRETAIN field is specified in the global system options (GSO) TSO record.

By default, the logon parameter retention facility is inactive unless you explicitly activate it, as described previously. Once activated, management of the supporting CONTROL(TSO) records is automatic and requires no administrative attention. However, a security administrator or INFOLIST privileged user can list the retention records.

Since the management of retention records is performed internally, no external facility is provided to update the fields contained in these records. ACF command support for CONTROL(TSO) records is provided primarily to review the use of the retention facility. In fact, when using the SET subcommand, SYSID does not need to be stated. Additionally, if you discontinue the use of the retention facility for any reason, the ACF command can be used to delete any or all of the unused retention records.

When a logon retention record is created, the user's logonid becomes the name of that user's retention record ID. The fields contained in the user's record are described in the following. When listed, only those parameters that differ from the defaults found in the logonid record are displayed.

ACCOUNT
Identifies the TSO logon account string. This field corresponds to the TSOACCT field of the logonid record.

ATTR1
Defines the status of the retention record. These flags are for internal use only and have no counterpart in the logonid record.

ATTR2
Defines the status of the retention record. These flags are for internal use only and have no counterpart in the logonid record.

ATTR3
Defines the status of the retention record. These flags are for internal use only and have no counterpart in the logonid record.

COMMAND
Contains a TSO command issued immediately after logon is completed when your systems programmer activates the command string feature. This field has no counterpart in the logonid record.

MSGCLASS
Identifies the SYSOUT class to which the TSO session is assigned. This field has no counterpart in the logonid record.

PERFORM
Identifies the MVS performance group the TSO session runs under. This field corresponds to the TSOPERF field of the logonid record.

PROC
Identifies the TSO JCL procedure name. This field corresponds to the TSOPROC field of the logonid record.

REGION
Identifies the MVS region size in kilobytes. This field corresponds to the TSORGN field of the logonid record.

TIME
Identifies the CPU time limit associated with the TSO session. This field corresponds to the TSOTIME field of the logonid record.

UNIT
Identifies the default MVS unit name selected during data set allocations. This field corresponds to the TSOUNIT field of the logonid record.

Using the ISPF Panels

To create GSO records, select option 6 GSO from the eTrust CA-ACF2 ISPF Option Selection Menu.

```
---------------- eTrust CA-ACF2 ISPF OPTION SELECTION MENU ---------------------
OPTION  ===>

   1  RULES     - PROCESS eTrust CA-ACF2 ACCESS AND GENERALIZED RESOURCE RULES
   2  LOGONIDS  - eTrust CA-ACF2 LOGONID CREATION/MAINTENANCE FACILITY
   3  SYSTEM    - eTrust CA-ACF2 SHOW COMMANDS
   4  REPORTS   - eTrust CA-ACF2 REPORT PROGRAM PROCESSOR
   5  UTILITIES - PROCESS eTrust CA-ACF2 UTILITIES
   6  GSO       - GLOBAL SYSTEM OPTIONS SERVICES
   7  NET       - NETWORKING SYSTEM OPTIONS SERVICES
   8  CAC       - MVS DATABASE CACHE RECORD SERVICES
   9  XREF      - SOURCE/RESOURCE AND GROUP RECORD SERVICES
  10  MAC       - MANDATORY ACCESS CONTROL ADMINISTRATION
  11  CPF       - COMMAND PROPAGATION FACILITY SERVICES
  12  FIELD     - RECORD LEVEL PROTECTION CONTROLS
  13  TARGETS   - SET CPF TARGET NODES, DEFAULTS IN USE
  14  PROFILE   - PROCESS PROFILE INFORMATION RECORDS
  15  SMS       - PROCESS DFSMS SUPPORT RECORDS
  16  ENTRY     - PROCESS ENTRY SOURCE RECORDS
  17  SHIFT     - PROCESS SHIFT/ZONE RECORDS
  18  RACDCERT  - PROCESS KEYRING/CERTIFICATE COMMANDS
  19  C-CIC     - PROCESS C-CIC CICS INITIALIZATION RECORDS
  20  LDS       - PROCESS LDAP DIRECTORY SERVICES
```

The eTrust CA-ACF2 Global System Options Services panel is displayed:

```
------------- eTrust CA-ACF2 GLOBAL SYSTEM OPTIONS SERVICES --------------------
OPTION  ===>


   1  INSERT   - INITIALLY DEFINE GSO RECORDS

   2  CHANGE   - CHANGE EXISTING GSO RECORDS

   3  LIST     - DISPLAY GSO RECORDS AND PARAMETERS

   4  DELETE   - DELETE AN ENTIRE GSO RECORD

   5  eTrust CA-ACF2  - SHOW ALL ACFFDR AND GSO OPTIONS IN EFFECT

   6  FIELDS   - SHOW GSO RECORD FIELD NAMES

   7  TARGETS  - SET CPF TARGET NODES, DEFAULTS IN USE
```

If you want to process an ACF command at a node other than the default node or the current target setting, you must select the TARGET option and specify the target nodes **before** you select 1 INSERT, 2 CHANGE, 3 LIST, or 4 DELETE.

Creating GSO Records

Select option 1 INSERT from the eTrust CA-ACF2 Global System Options Services panel to create a GSO record. The Insert a GSO Record panel is displayed:

```
---------------------------- INSERT GSO RECORD -------------------------------
COMMAND ===>

  INSERT

  CHANGE TYPE ===>ADD      (ADD)

  SYSID       ===>         SYSTEM ID FOR GSO RECORD

  USING SYSID ===>         OPTIONAL SYSTEM ID FOR COPY OF EXISTING VALUES

  USING RECID ===>                 OPTIONAL PROTOTYPE RECORD NAME

  RECID        ===>          (APPLDEF   AUTHEXIT  AUTOERAS  AUTOIDLX  AUTOIDOM
          BACKUP    BLPPGM    CACHESRV  CERTMAP   CLASMAP   CRITMAP   DELRSRC
          EIM       ETAUDIT   EXITS     INFODIR   LINKLST   LINUX     LOGPGM
          MAINT     MLID      MLSOPTS   MUSASS    NJE       OPTS      PDS
          PPGM      PROXY     PSWD      REALM     RESDIR    RESRULE   RESVOLS
          RESWORD   RULEOPTS  SAFDEF    SECVOLS   STC       SYNCOPTS  SYSPLEX
          TNGNODE   TSO       TSOCRT    TSOKEYS   TSOTWX    TSO2741   UNIXOPTS
          WARN)

 SPECIFICATION OF RECID WILL RESULT IN DISPLAY OF APPROPRIATE PANEL
 FOR OPTION SPECIFIED
```

Panel Field Descriptions

CHANGE TYPE
Specify one of the following types of changes to the record:

- ADD
  adds the fields to an existing record.
- DEL
  deletes the fields from an existing record.
- REP
  replaces the fields in an existing record with the fields you specify.

SYSID
Specify the system ID for the GSO record that you want to create.

USING SYSID
Specify a system ID of another GSO record that you want to use to create a GSO record on the current system.

USING RECID
Specify the name of a record that you want to use as a model to create this record.

RECID
Specify the one to eight-character name of the record ID that you want to create.

Activating GSO Records

If you insert a GSO record and want that change immediately effective on your system, you **must** issue the F ACF2,REFRESH operator command; otherwise, eTrust CA-ACF2 does not recognize the change until the GSO records are built at the next IPL of the system. No ISPF panel exists for this refresh task.

Changing GSO Records

To change a GSO record, select option 2 CHANGE from the eTrust CA-ACF2 Global System Options Services panel. The Change a GSO Record panel is displayed:

```
----------------------------- CHANGE GSO RECORD ------------------------------
COMMAND ===>

  CHANGE

  CHANGE TYPE ===>ADD      (ADD, DEL, REP)

  SYSID       ===>         SYSTEM ID FOR GSO RECORD

  MASK SYSID  ===>         MASKED SYSTEM ID FOR GSO RECORD

  MASK RECID  ===>                 RECID MASKED VALUE

  RECID        ===>          (APPLDEF   AUTHEXIT  AUTOERAS  AUTOIDLX  AUTOIDOM
          BACKUP    BLPPGM    CACHESRV  CERTMAP   CLASMAP   CRITMAP   DELRSRC
          EIM       ETAUDIT   EXITS     INFODIR   LINKLST   LINUX     LOGPGM
          MAINT     MLID      MLSOPTS   MUSASS    NJE       OPTS      PDS
          PPGM      PROXY     PSWD      REALM     RESDIR    RESRULE   RESVOLS
          RESWORD   RULEOPTS  SAFDEF    SECVOLS   STC       SYNCOPTS  SYSPLEX
          TNGNODE   TSO       TSOCRT    TSOKEYS   TSOTWX    TSO2741   UNIXOPTS
          WARN)

 SPECIFICATION OF RECID WILL RESULT IN DISPLAY OF APPROPRIATE PANEL
 FOR OPTION SPECIFIED
```

Panel Field Descriptions

CHANGE TYPE
Specify one of the following types of changes to the record:

- ADD
  adds the fields to an existing record.
- DEL
  deletes the fields from an existing record.
- REP
  replaces the fields in an existing record with the fields you specify.

SYSID
Specify the system ID for the GSO record that you want to create.

MASK SYSID
Specify a mask for the system IDs of the to which this GSO record applies.

MASK RECID
Specify a mask for the record IDs to which this change applies.

RECID
Specify the one to eight-character name of the record ID that you want to change.

Activating GSO Records

If you change a GSO record and you want that change immediately effective on your system, you **must** issue the F ACF2,REFRESH operator command; otherwise, eTrust CA-ACF2 does not recognize the change until the GSO records are built at the next IPL of the system. No ISPF panel exists for this refresh task.

Displaying GSO Records

Select option 3 LIST from the eTrust CA-ACF2 Global System Options Services panel to display a GSO record. The List a GSO Record panel is displayed:

```
---------------------------- LIST GSO RECORD --------------------------------
COMMAND ===>

  LIST

  SYSID       ===>         SYSTEM ID FOR GSO RECORD

  MASK SYSID  ===>         MASKED SYSTEM ID FOR GSO RECORD

  QUALIFIER   ===>                 GSO RECORD QUALIFIER

  MASK RECID  ===>                 RECID MASKED VALUE

  RECID        ===>          (APPLDEF   AUTHEXIT  AUTOERAS  AUTOIDLX  AUTOIDOM
          BACKUP    BLPPGM    CACHESRV  CERTMAP   CLASMAP   CRITMAP   DELRSRC
          EIM       ETAUDIT   EXITS     INFODIR   LINKLST   LINUX     LOGPGM
          MAINT     MLID      MLSOPTS   MUSASS    NJE       OPTS      PDS
          PPGM      PROXY     PSWD      REALM     RESDIR    RESRULE   RESVOLS
          RESWORD   RULEOPTS  SAFDEF    SECVOLS   STC       SYNCOPTS  SYSPLEX
          TNGNODE   TSO       TSOCRT    TSOKEYS   TSOTWX    TSO2741   UNIXOPTS
          WARN)
```

Panel Field Descriptions

SYSID
Specify the system ID for the GSO record that you want to list.

MASK SYSID
Specify a mask for the system IDs for which to list GSO records.

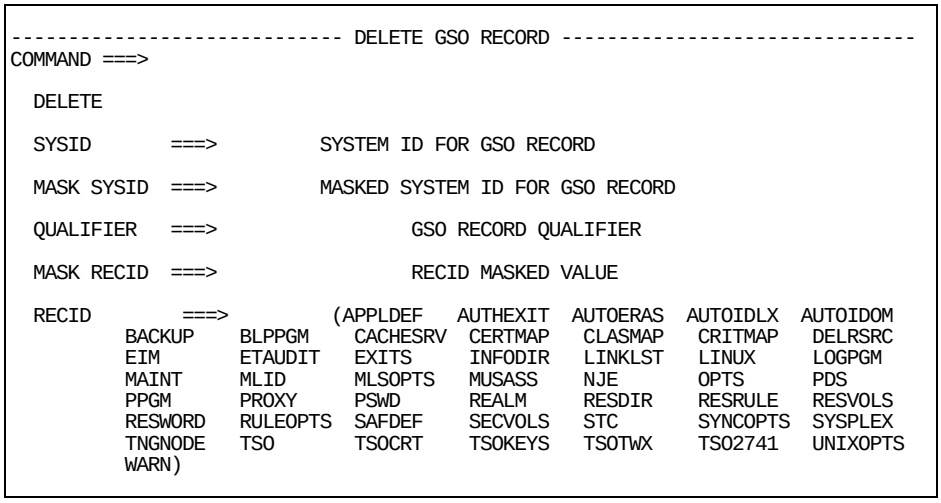
QUALIFIER
Specify the qualifier for the GSO record that you want to list.

MASK RECID
Specify a mask for the record IDs that you want to list.

RECID
Specify the one to eight-character name of the record ID that you want to list.

Deleting GSO Records

Select option 4 DELETE from the eTrust CA-ACF2 Global System Options Services panel to delete a GSO record. The Delete a GSO Record panel is displayed:



```
----------------------------- DELETE GSO RECORD ------------------------------
COMMAND ===>

  DELETE

  SYSID       ===>         SYSTEM ID FOR GSO RECORD

  MASK SYSID  ===>         MASKED SYSTEM ID FOR GSO RECORD

  QUALIFIER   ===>                 GSO RECORD QUALIFIER

  MASK RECID  ===>                 RECID MASKED VALUE

  RECID        ===>          (APPLDEF   AUTHEXIT  AUTOERAS  AUTOIDLX  AUTOIDOM
          BACKUP    BLPPGM    CACHESRV  CERTMAP   CLASMAP   CRITMAP   DELRSRC
          EIM       ETAUDIT   EXITS     INFODIR   LINKLST   LINUX     LOGPGM
          MAINT     MLID      MLSOPTS   MUSASS    NJE       OPTS      PDS
          PPGM      PROXY     PSWD      REALM     RESDIR    RESRULE   RESVOLS
          RESWORD   RULEOPTS  SAFDEF    SECVOLS   STC       SYNCOPTS  SYSPLEX
          TNGNODE   TSO       TSOCRT    TSOKEYS   TSOTWX    TSO2741   UNIXOPTS
          WARN)
```

Panel Field Descriptions

SYSID
Specify the system ID for the GSO record that you want to delete.

MASK SYSID
Specify a mask for the system IDs of the GSO records that you want to delete.

QUALIFIER
Specify the qualifier for the GSO records that you want to delete.

MASK RECID
Specify a mask for the record IDs that you want delete.

RECID
Specify the one to eight-character name of the record ID that you want to delete.

Activating GSO Records

If you delete a GSO record and you want that change immediately effective on your system, you **must** issue the F ACF2,REFRESH operator command; otherwise, eTrust CA-ACF2 does not recognize the change until the GSO records are built at the next IPL of the system. No ISPF panel exists for this refresh task.

Displaying GSO Options

Select option 5 from the eTrust CA-ACF2 Global System Options Services panel. The panel displays for the SHOW subcommands are described in the "Overview of eTrust CA-ACF2" chapter.

Displaying Field Names for a GSO Record

Select option 6 FIELDS from the eTrust CA-ACF2 Global System Options Services panel. The fields for the various GSO records are described later in this chapter. See the appropriate record to view the field descriptions.

Setting Target Nodes

Select option 7 TARGETS from the eTrust CA-ACF2 Global System Options Services panel to set the target nodes where you want the commands to take effect. The Set CPF Target Nodes panel is displayed:

```
-----------------eTrust CA-ACF2 GLOBAL SYSTEM OPTIONS SERVICES----------------
COMMAND ===>


  SET THE CPF TARGET NODE(S) FOR ALL LOGONID AND INFOSTORAGE DATABASE
  PROCESSING. THE TARGET NODE LIST IS RETAINED ACROSS ISPF SESSIONS.

  THE NODE NAMES MAY BE MASKED. IF THEY ARE BLANK, eTrust CA-ACF2 WILL USE
  THE DEFAULT TARGET NODES. ENTERING THE LAST ENTRY WILL PROMPT FOR
  ADDITIONAL LINES.

          CMDWAIT   ===>     (Y, N)  SYNCHRONOUS/ASYNCHRONOUS

                    ===>             ===>
                    ===>             ===>
                    ===>             ===>
                    ===>             ===>
                    ===>             ===>
                    ===>             ===>
                    ===>             ===>
                    ===>             ===>
                    ===>             ===>
                    ===>             ===>
```

Enter up to 100 target nodes. The CPF address space must be active on each node and you must have defined the nodes in CPF NODEDEF and OPTIONS records for the commands to take effect.

Using the ACF Command

You can use the ACF command to create and maintain GSO records. For descriptions of the syntax of the INSERT, LIST, CHANGE, and DELETE subcommands, see the "Maintaining Structured Infostorage Records" chapter.

If you insert, change or delete a GSO record and you want that change immediately effective on your system, you **must** issue the F ACF2,REFRESH operator command; otherwise, eTrust CA-ACF2 does not recognize the change until the GSO records are built at the next IPL of the system.

Chapter

15 Maintaining Profile Records

The System Authorization Facility (SAF) is the z/OS interface to external security. SAF has the ability to extract information and data from the security database that can be used by system applications. Instead of keeping this information in an application database, the information is kept in the security database and the application requests the information by issuing a SAF RACROUTE REQUEST=EXTRACT call. This chapter describes how such information is kept and maintained in eTrust CA-ACF2.

Profile and Segment Information

The information that can be retained in the security database is associated with a user or a z/OS resource. This general classification is commonly referred to as a *profile*. (For example, a USER profile, a DATASET profile, or other resource profile.) The group of data that is to be extracted is commonly referred to as *segment data*. For example, a SAF call can be issued to extract the OPERPARM segment of the USER profile, the DFP segment of the DATASET profile, or the DLFDATA segment of the DLFCLASS general resource profile.

The following profile and segment information is supported by eTrust CA-ACF2:

Profile Record	Segment
APPCLU	NEXTKEY, SESSION
DATASET	DFP
DLFCLASS	DLFDATA
GROUP	LINUX, OMVS
KEYSMSTR	SSIGNON
PTKTDATA	SSIGNON
SDB2	
SECLABEL	SECLEVEL, CATEGORY, SECLABEL
SECLABEL	CATEGORY

Profile Record	Segment
SECLABEL	SECLABEL
SYSMVIEW	SVFMR
USER	CERTDATA, CICS, DCE, EIM, KERB, KERBLINK, KEYRING, LANGUAGE, LINUX, LNOTES, NDS, NETVIEW, OMVS, OPERPARM, PASSWORD, PROXY, SECLABEL, WORKATTR

Note: The USER profile records are documented in the "Maintaining Logonid Records" Chapter. Also under the umbrella of the USER profile are the BASE and TSO segments. The data for these segments is maintained within the eTrust CA-ACF2 logonid record. See BASE and TSO Segment Considerations in the "RACF to eTrust CA-ACF2 Translation" appendix for a more detailed description of eTrust CA-ACF2 support for these two segments.

eTrust CA-ACF2 keeps this information in the infostorage database. The records are administered through the eTrust CA-ACF2 ACF command. There are two types of infostorage records that contain the information that can be requested through SAF: the *profile record* and the *profile data record*.

- The profile record is associated with a resource whose entity name is similar to a data set name. Since these entity names can be larger than the maximum record key of an infostorage record, the compiled infostorage record is used; these compiled records are similar to data set access rules. Masking of the entity name can ease administration of these resources. The compiled infostorage records do not contain the data that is to be extracted; they are used to point to the profile data records that contain the data.
- The profile data record contains the actual segment data that is to be extracted from the profile. Since the entity name of a user profile consists of a simple userid, user profiles use only profile data records and the userid is the record key; masking of the userid is allowed.

APPCLU Profile Records

APPCLU profile records are used by VTAM to provide LU session-level security when the VTAM APPL statement specifies VERIFY=OPTIONAL or VERIFY=REQUIRED. The entity name for this profile is a node name followed by the local and remote LU names. The profile record contains a key that is used as a password. A connection between VTAM sessions is only allowed if identical keys are specified for both sessions. VTAM does not let the sessions start if the keys do not match or if the key is expired (per the KEYINTVL parameter in the SESSION profile data record).

Note: A CLASMAP override for APPCLU, which maps to ALU, is not allowed.

APPCLU Compiled Records

The APPCLU compiled records define the LU connection that is being protected and indicates which SESSION profile data record is to be used to obtain information about the session.

The entity name is a node name followed by the local and remote LU names.

```
SET PROFILE(APPCLU) DIVISION(PROFILE)

COMPILE *

$KEY(node1)
 loc allu.remotelu SESSION(sessrec)

STORE
```

VTAM validates the APPCLU resource name using one of the following formats, depending on the NQNAMES= specification on the VTAM ACB. If NQNAMES=NO, the format used for validation is illustrated in the previous example as node1.locallu.remotelu, where node1 is NETID, locallu is LU1, and remotelu is LU2. If NQNAMES=YES, the format used for validation is node1.locallu.node2.remotelu where node1 is NETID1, locallu is LU1, node2 is NETID2, and remotelu is LU2. In this case, the rule would look like this:

```
$KEY(node1)
 loc allu.node2.remotelu SESSION(sessrec)
```

When you mask $KEYs, make the rules resident using the GSO INFODIR record.

You can use the ACF subcommand RECKEY to maintain this compiled profile record.

NEXTKEY Segment

The NEXTKEY segment can be specified in an APPCLU profile record to cause eTrust CA-ACF2 to evaluate an alternate APPCLU profile record. This lets a large APPCLU profile record be split into multiple APPCLU profile records. eTrust CA-ACF2 only checks the NEXTKEY segment when the VTAM session connection matches the environment in the APPCLU profile record, but the record prevents the connection.

SESSION Profile Data Records

The SESSION profile data records define the security key to be used, expiration dates, and type of conversation security to be used. Ifconversation security type is specified, it overrides the SECACPT keyword value specified on the VTAM APPL statement.

Record ID	Fields
Recid	SESSKEY(*sesskey*) KEYINTVL(*keyintvl*) LOCK | NOLOCK VERIFY | ALREADYV | PERSISTV | AVPV | CONVNONE

Field Descriptions

recid
The record name as defined in the compiled record SESSION field.

SESSKEY(*sesskey*)
A one to 16-hexadecimal value representing the session key. To protect the session key from disclosure when the PTKTDATA record is listed, the SESSKEY field value is displayed as "*SUPPRESSED*". A message similar to the following appears when the record is listed:

```
SESSION / SESSREC LAST CHANGED BY ACFUSER ON 12/31/04 - 12:59
                  K EYINTVL(60) SESSKEY(*SUPPRESSED*) VERIFY
```

KEYINTVL(*keyintvl*)
The number of days from 1 to 32767 that the session key is valid.

LOCK | NOLOCK
Lock the profile to prevent establishment of a session or unlock a locked profile. NOLOCK is the default.

VERIFY | ALREADYV | PERSISTV | AVPV | CONVNONE

- **VERIFY**—conversation security verify option to perform userid and password validation for inbound requests.
- ALREADYV—conversation security already verified option.
- PERSISTV—conversation security persistent verification option. If Persistent Verification is to use VLF, you must have the following entries in member COVLFxx in SYS1.PARMLIB:

```
CLASS NAME(SECINFO)  /*  Class name for security information
     E MAJ(PVSEC)    /* Major name for security information
```

This defines a specific class name and major name to the Virtual Lookaside Facility (VLF), which Persistent Verification uses to exploit data spaces.

- **AVPV**—conversation security already verified and persistent verification option.
- **CONVNONE**—no conversation security validation is to be performed.

Example

To create a SESSION profile data record, issue the following ACF subcommands:

```
SET PROFILE(APPCLU) DIVISION(SESSION)

INSERT sessrec SESSKEY(c1c3c6f2acf2) KEYINTVL(60) VERIFY
```

This example shows the creation of a SESSION profile data record named *sessrec* that says security key *clc3c6f2acf2*, which is active for 60 days, is to be used to verify inbound requests for userid and password information every time a signon request is made from the LU defined in the APPCLU profile record (which is *node1.locallu.remotelu* from the previous example).

After you create the SESSION profile data record, issue the following command to activate this new record:

```
F ACF2,REBUILD(ALU),CLASS(P)
```

DATASET Profile Records

DATASET profile records currently use only the DFP profile data, which is used by DFSMS to obtain the RESOWNER value for a data set. Use of a profile record to define the resowner of a data set supersedes the use of the $RESOWNER data set access rule keyword; if no profile record is found for a data set, see "Implementing DFSMS Support" for more information.

One benefit of using profile records includes assignment of RESOWNER at the data set level, as opposed to rule set level (a restriction of $RESOWNER). Another is the ability to separate the duties of the security administrator from those of the storage administrator. The data set access rule administrator can be scoped to allow access only to the data set access rules, and the storage administrator can be scoped to allow access only to the data set profile records.

```
SET SCOPE(SCP)

INSERT stgadm INF(PDSN-)

SET LID

CHANGE userid SECURITY SCPLIST(stgadm)
```

DATASET Compiled Records

The compiled records in the DATASET profile are used to specify which DFP profile data record is used to obtain the RESOWNER for a particular data set. Masking in the compiled record is similar to that in data set access rules. In the following example, one rule set points to four different profile data records that define the RESOWNER userids.

```
SET PROFILE(DATASET) DIVISION(PROFILE)

COMPILE *

$KEY(personnel)
ben efits.-  DFP(persben)
pay roll.- D FP(perspay)
adm in.-  DFP(persadm)
-.- D FP(persgen)

STORE
```

Unlike data set access rules, masking is allowed in the $KEY. For example, to use the RESOWNER specified in the SYSTEMS DFP profile data record for all SYS1, SYS2 and SYSn data sets, compile and store the following:

```
$KEY(sys*)
-.- D FP(systems)
```

When you mask $KEYs, make the records resident using the GSO INFODIR record.

You can use the ACF subcommand RECKEY to maintain this compiled profile record.

DFP Profile Data Records

The compiled records specify which DFP profile data record is to be used to obtain the RESOWNER for the data set.

Record ID	Fields
recid	RESOWNER(*resowner*)

Field Descriptions

recid
The record ID as specified in the compiled record DFP field.

RESOWNER
A one to eight-character RESOWNER userid. A logonid with this name **must** exist.

Example

To continue with the first example in DATASET Compiled Records, the DFP profile data records are inserted for the personnel data sets:

```
SET PROFILE(DATASET) DIVISION(DFP)

INSERT persben RESOWNER(benefits)
INSERT perspay RESOWNER(payroll)
INSERT persadm RESOWNER(admin1)
INSERT persgen RESOWNER(personnl)
```

DLFCLASS Profile Records

DLFCLASS profile records are used by the z/OS Data Lookaside Facility. The entity name of this profile is a data set name that becomes a DLF object. Support for DLF requires the use of both profile records and resource rules.

To determine if a data set object should reside in storage, DLF issues an authorization call to see if the current user is allowed access to the object for DLF processing. Only the resource rule is checked for authorization. SECURITY or NON-CNCL privileges in the user's logonid are not sufficient to allow access. If the resource rule allows access, DLF issues an extract call to get a list of jobnames under which the object can be processed by DLF.

In the following examples, certain payroll data sets can be processed through DLF if accessed using the correct production userid and jobnames.

DLFCLASS Compiled Records

Compiled records specify which profile data records are used to obtain DLFDATA information:

```
SET PROFILE(DLFCLASS) DIVISION(PROFILE)

COMPILE *

$KEY(payroll)
 bat ch1.data DLFDATA(payjob1)
 bat ch*.data DLFDATA(payjob2)

STORE
```

When you mask $KEYs, make the records resident using the GSO INFODIR record.

You can use the ACF subcommand RECKEY to maintain this compiled profile record.

DLFDATA Profile Data Records

DLFDATA profile data records specify the list of jobnames under which the data set object can be processed by DLF, and whether the object should be retained in storage for further processing after the current user is finished.

Record ID	Fields
recid	JOBNAMES(*jobid1,…jobidn*) RETAIN | NORETAIN

Field Descriptions

recid
The record ID as specified in the compiled record DLFDATA field.

JOBNAMES(*jobid1,...*)
A list of jobnames under which the data set object can be processed by DLF. This field can be masked using the RACF conventions of * and %. A single * masks one or more characters until the end of the jobname or zero or more qualifiers until the end of the data set name. This is similar to using a dash in eTrust CA-ACF2 for masking. The % is used to mask a single character in a job name. This is similar to using a * in eTrust CA-ACF2 masking.

RETAIN
Specifies whether the data set can be retained by DLF after initial use.

NORETAIN
The data set is not to be retained.

To continue with the previous example, DLFDATA profile data records are inserted for the payroll data sets:

```
SET PROFILE(DLFCLASS) DIVISION(DLFDATA)

INSERT payjob1 JOBNAMES(job1,job2,job3,job4,job5) RETAIN
INSERT payjob2 JOBNAMES(job2,job3,job4,job5)
```

DLFCLASS Resource Rules

eTrust CA-ACF2 checks DLF resource rules to ensure that a userid has access to the resource (data set name) for DLF processing. DLF resource rules are written so that DLF processing of data sets only occurs when used by the specified userids:

```
SET Resource(DLF)

COMPILE *

$KEY(payroll) TYPE(DLF)
batch*.data UID(prodid) ALLOW

STORE
```

When you mask $KEYs, make the rules resident using the GSO INFODIR record.

GROUP Profile Records Data Records

The GROUP profile records contain information needed by z/OS Unix System Services and LINUX/390 applications (PAM Server) to verify user access. The profile data information segments that can be extracted for a GROUP include LINUX and OMVS.

The key of a GROUP profile record is a group name, which cannot be masked. GID numeric values can be automatically assigned with GSO AUTOIDLX and AUTOIDOM records. For more information, see Automatic UID/GID assignment (AUTOIDLX) and Automatic UID/GID Assignment (AUTOIDOM) in the "Maintaining Global System Options Records" chapter.

OMVS Profile Data Records

The OMVS segment of the GROUP profile contains information needed by z/OS Unix System Services to verify user access.

Record ID	Fields
Recid	AUTOGID GID(*gid#*)

recid
Specifies one to eight character group name. This value cannot be masked.

AUTOGID
This will result in an automatically assigned GID value when there is an active GSO AUTOIDOM record that specifies ASSIGNG. The GID keyword and the AUTOGID keyword are mutually exclusive. If neither AUTOGID nor GID is specified on an insert command, and there is an active GSO AUTOIDOM record with ASSIGNG specified, AUTOGID will be assumed. AUTOGID is never assumed on a change command, it must be stated explicitly.

On an insert command when neither GID nor AUTOGID is specified, and there is no active GSO AUTOIDOM record, or the AUTOIDOM record specifies NOASSIGNU, the GID number will default to zero.

GID(*gid#*)
A numeric field that accepts values from zero to 2,147,483,647. It defaults to 0 if neither GID nor AUTOGID is specified on an INSERT command and there is **no** active GSO AUTOIDOM record, or there is an active GSO AUTOIDOM record that specifies NOASSIGNG. The GID and AUTOGID keywords are mutually exclusive.

For more information about OMVS, see the "z/OS Unix System Services Support" chapter.

Rebuild Command

If you insert or change a Group profile record and it is resident then you must issue a REBUILD command to activate the changes.

```
F ACF2,REBUILD(GRP),CLASS(P)
```

Examples

This section explains how to set up eTrust CA-ACF2 to automatically assign GID numbers for PROFILE(GROUP),DIV(OMVS) records. Readers should already be familiar with the AUTOIDOM record. See the "Maintaining Global System Options Records" chapter for information on the AUTOIDOM record. Considerations are discussed for shared database environments and CPF environments.

AUTO Assignment of GID Numbers

To use this feature there must be an active GSO AUTOIDOM record. For this example, there is an AUTOIDOM record with the following fields:

```
CPU1/AUTOIDOM LAST CHANGED BY USER01 ON 06/26/04-14:04
                 AS SIGNU ASSIGNG GIDEND(50,000) GIDNEXT(25) GIDSTART(9)
                 UI DEND(2,147,483,647) UIDNEXT(195) UIDSTART(1)
```

The following command inserts an OMVS Group profile record, OMVSGRP. The GID field is automatically assigned. **Note:** The AUTOGID field is assumed on the INSERT command and does not need to be specified.

```
SET PROFILE(GROUP) DIV(OMVS)
INSERT OMVSGRP

OMVS/OMVSGRP LAST CHANGED BY USER01 ON 06/26/04-16:26
                 G ID(25)
```

The following command automatically assigns the new GID value to a recently inserted OMVSGRP record. On the CHANGE command, the AUTOGID field must be specified to automatically assign the new GID value; otherwise you must explicitly specify the new GID value.

```
CHANGE OMVSGRP AUTOGID

OMVS/OMVSGRP LAST CHANGED BY USER01 ON 06/26/04-16:30
                 GI D(26)
```

After updating the OMVS Group profile record (OMVSGRP) the AUTOIDOM record now reflects the recent updates with a new value in GIDNEXT(27).

```
SET CONTROL (GSO)
LIST AUTOIDOM

CPU1/AUTOIDOM LAST CHANGED BY USER01 ON 06/26/04-14:04
             ASS IGNU ASSIGNG GIDEND(50,000) GIDNEXT(27) GIDSTART(9)
             UID END(2,147,483,647) UIDNEXT(195) UIDSTART(1)
```

SHOW OMVS

The default value range is 1 to 2,147483,647 for GIDs. Some sites may want to specify a range of numbers that are not in use by any existing GID records. To see what numbers are already in use, issue the SHOW OMVS command. If, for example, you want to see what GID values are already in use in the range 900 through 399999, you can issue:

```
SHOW OMVS GROUP(900-399999)
```

If you wish to see only the GID values that belong to more than one group, issue the ACF command:

```
SHOW OMVS DUPLICATES
```

The DUPLICATES keyword can be used together with another keyword. For example the following will show only duplicate GID values that are in the range of 1 to 2000:

```
SHOW OMVS GROUP(1-2000) DUPLICATES
```

AUTO Assignment Within a CPF Environment

When the AUTOUID or AUTOGID keyword is used or implied, the keyword itself is not sent across to connected CPF nodes. Rather, the actual assigned value is sent. For example, suppose you issue the command:

```
SET P(GROUP) DIV(OMVS)
INS DEVGROUP AUTOGID
 OMVS / DEVGROUP LAST CHANGED BY USER071 ON 02/06/04-17:16
                      GID(57)
```

When this command is sent to other CPF nodes, it will look like this:

```
INS DEVGROUP GID(57)
```

LINUX Profile Data Records

The LINUX segment of the GROUP profile contains information needed by LINUX application (PAM Server) to verify user access.

Record ID	Fields
Recid	AUTOGIDL LINUXGID(*gid*#)

Field Descriptions

recid
Specifies one to eight character eTrust CA-ACF2 logon id. This value cannot be masked.

AUTOGIDL
It automatically assigns LINUXGID value when there is an active GSO AUTOIDLX record that specifies ASSIGNG field. AUTOGIDL is implied if neither AUTOGIDL nor LINUXGID is specified on an INSERT command. AUTOGIDL is never implied on a CHANGE command, it must be stated explicitly. The LINUXGID and the AUTOGIDL keywords are mutually exclusive.

Note: When using the AUTOGIDL feature, you must have a nonqualified AUTOIDLX GSO record. The Linux segment of the GROUP profile data records do not support qualifiers.

LINUXGID(*gid#*)
A numeric field that accepts values from 0 to 2,147,483,647. It defaults to 100 if neither LINUXGID nor AUTOGIDL is specified on an INSERT command and there is no active GSO AUTOIDLX record, or there is an active GSO AUTOIDLX record that specifies NOASSIGNG. The LINUXGID and the AUTOGIDL keywords are mutually exclusive.

Rebuild Command

If you insert or change a group profile record and it is resident, then you must issue a REBUILD command to activate the changes.

```
F ACF2,REBUILD(GRP),CLASS(P)
```

Examples

The following example automatically assigns a LINUXGID number using both the INSERT and CHANGE subcommand. There is an active GSO AUTOIDLX record that specifies ASSIGNG, GIDSTART(100), and GIDNEXT(100). By the end of these commands the AUTOIDLX record holds GIDSTART(100) and GIDNEXT(102).

```
SET PROFILE(GROUP) DIV(LINUX)
INSERT LNXGRP2
LINUX / LNXGRP2 LAST CHANGED BY USER01 ON 06/26/04-16:33
        LI NUXGID(100)

CHANGE LNXGRP2 AUTOGIDL
LINUX / LNXGRP2 LAST CHANGED BY USER01 ON 06/26/04-16:40
        LI NUXGID(101)
```

KEYSMSTR Profile Record

The eTrust CA-ACF2 KEYSMSTR profile record provides support for administration of DCE and LDAP BIND passwords. This allows the DCE or LDAP BIND password to be encrypted for storage on the CA-ACF2 database.

The KEYSMSTR profile class can hold up to two records, DCE.PASSWORD.KEY and LDAP.BINDPW.KEY. DCE.PASSWORD.KEY is used in the administration of DCE user profile records while LDAP.BINDPW.KEY is used in the administration of PROXY user profile records and PROXY GSO records. If no DCE.PASSWORD.KEY record is defined, eTrust CA-ACF2 will utilize its own internal routine to process the user's DCE password before storing it in the user's DCE user profile record. The DCE.PASSWORD.KEY can be changed at any time without impacting existing DCE user profile records. If no LDAP.BINDPW.KEY record is defined, eTrust CA-ACF2 will utilize its own internal routine to process the user's LDAP BIND password before storing it in the proper PROXY record.

The SSKEY field of the KEYSMSTR profile record contains the encryption key mask used for encoding or encrypting the DCE or LDAP BIND password. In addition, one of two attributes, KEYMASK or KEYCRYPT, indicates whether the DCE or LDAP BIND password should be encoded or encrypted using the key value supplied in the SSKEY in the Infostorage database.

***IMPORTANT**! In order to use the KEYSMSTR class, you must load the DCE.PASSWORD.KEY and/or LDAP.BINDPW.KEY record into storage by adding an entry for the PKEY infostorage directory in the GSO INFODIR record. For example:*

```
acf
 ACF
set control(gso) sysid(prod)
 CO NTROL
change infodir types(r-pkey) ADD
```

You must also refresh the GSO INFODIR record and rebuild the P(KEY) directory whenever any changes are made to the DCE.PASSWORD.KEY or LDAP.BINDPW.KEY record.

For example:

```
F ACF2,REFRESH(INFODIR)
F ACF2,REBUILD(key),CLASS(p)
```

Note: If sharing databases with a release prior to r6.5 of eTrust CA-ACF2, you cannot define a KEYSMSTR profile record.

Record ID	Fields
Recid	SSKEY(*key-value*) KEYMASK|KEYCRYPT

Field Descriptions

Recid
This field is the record ID. The record ID can be DCE.PASSWORD.KEY and LDAP.BINDPW.KEY. No other value is valid. Recid is a required field.

SSKEY
A 16-hexadecimal digit key representing the eight-byte key that is used to encode (mask) or encrypt the DCE or LDAP BIND password. SSKEY is a required field.

KEYMASK|KEYCRYPT
Indicates whether the SSKEY value is to be used to encode (mask) the DCE password.

If KEYMASK is specified, the SSKEY value will be used to encode (mask) or encrypt the DCE or LDAP BIND password whenever the password itself is changed. KEYMASK is mutually exclusive with KEYCRYPT.

If KEYCRYPT is specified, the SSKEY value will be used to encrypt the DCE or LDAP BIND password whenever the password itself is changed. KEYCRYPT is mutually exclusive with KEYMASK.

You must specify KEYMASK or KEYCRYPT; there is no default.

Note: Since KEYMASK and KEYCRYPT are mutually exclusive, if you specify **both** KEYMASK and KEYCRYPT together, only the last one supplied will be used.

Examples

Issue the following commands to encrypt the DCE password within a user's DCE user profile record.

```
ACF
 se t profile(keysmstr) division(ssignon)
PROFILE
Insert DCE.PASSWORD.KEY SSKEY(1234567890abcdef) KEYCRYPT
```

Issue the following commands to encrypt the LDAP BIND password within a PROXY record.

```
ACF
 se t profile(keysmstr) division(ssignon)
PROFILE
 In sert LDAP.BINDPW.KEY SSKEY(7890abcdef123456) KEYCRYPT
```

To protect the key coded in the SSKEY field from disclosure when the KEYSMSTR record is listed, the SSKEY field never displays its contents. Depending on whether KEYMASK or KEYCRYPT is selected, a message similar to one of the following appears when you list the KEYSMSTR record:

```
SSIGNON / DCE.PASSWORD.KEY LAST CHANGED BY ACFUSER ON 02/31/04 - 12:53 KEYMASK
SSIGNON / DCE.PASSWORD.KEY LAST CHANGED BY ACFUSER ON 02/31/04 - 12:53 KEYCRYPT
SSIGNON / LDAP.BINDPW.KEY LAST CHANGED BY ACFUSER ON 02/31/04 - 12:59 KEYCRYPT
```

PTKTDATA Profile Records

eTrust CA-ACF2 PTKTDATA profile records provide support for the Secured Signon function of the IBM Network Security Program. The Secured Signon feature lets network users sign on to a z/OS host using a dynamically generated password substitute called a PassTicket. PassTickets have a short lifespan and are resistant to password interception and replay attacks. They can be generated for access to z/OS, TSO, CICS, IMS, or APPC/MVS.

To generate a PassTicket, the PassTicket generator algorithm must be used. This algorithm requires specific information as input data:

- The user's USERID
- The application ID, APPLID, for which the PassTicket is being generated
- A time and date stamp
- A security key, which should be known to both the application generating the PassTicket, and the target application which will be using the PassTicket

There are two ways to generate a PassTicket using this cryptographic algorithm.

1. If you are running on a z/OS system, you can use the RCVTPTGN callable service to generate the PassTicket on the host. You must pass RCVTPTGN a USERID and an APPLID; the callable service will then use the current time and date stamp, and extract the necessary profile record from the Infostorage database to obtain the security key. Using all four pieces of information and applying the cryptographic algorithm an 8 byte PassTicket is generated.
2. You can create a program that incorporates the algorithm for any function that generates a PassTicket. Thismethod allows the PassTicket to be generated on a network.

See IBM SecureWay Security Server for z/OS RACF Macros and Interfaces, for more information about generating PassTickets.

To support PassTicket signon, eTrust CA-ACF2 Security finds security key values in profile records on its Infostorage database. The PTKTDATA profile record has a field named SSKEY. It contains the security key used for generating PassTickets or for decrypting the PassTicket and validating the signon.

Record ID	Fields
Recid	SSKEY(*key-value*) MULT-USE|NOMULT-USE

Field Descriptions

recid

A record ID that is used to identify a user's corresponding encryption key. The record can consist of a combination of userid, group name, and application name. The userid is that of the user attempting to sign on to the system. The group name is the group specified at sign on or the group value in the user's logonid record. The application name is a value that identifies the application to which the key value applies. For CICS, IMS, or APPC/MVS applications, this is the VTAM application ID. For TSO, it is the characters "TSO" suffixed with the SMF system ID as defined in the SMFPRMxx member of SYS1.PARMLIB. For z/OS batch jobs, it is the characters "MVS" suffixed with the SMF system ID. In both cases, any special characters in the SMF system ID are ignored (for example, "SY*6" becomes "SY6").

All system using PassTickets must have identical application names and session keys for all nodes on the network. For example, if you are accessing TSO on z/OS from an OS2 system that supports PassTicket processing, the application name must match the z/OS SYSID on TSO. In addition, all systems must use the same standard rules to establish the validity of the PassTickets.

When decrypting a PassTicket, up to four record ID combinations are evaluated, in the following order:

1. The application name concatenated with the group name and userid.
2. The application name concatenated with the userid.
3. The application name concatenated with the group name.
4. The application name.

When a profile record matching one of the first three combinations is found but does not contain a valid session key value, the fourth selection (application name only) is processed, ignoring all other selections. Therefore, session key values can be assigned at the userid or group level, at the application level, or at both levels. However, when generating a PassTicket only the profile record at the application level is used. This will be further illustrated by example, later in this section.

SSKEY(*key-value*)

A 16-character hexadecimal representation of the eight-byte encryption key for this application. This encryption key must match the encryption key defined on the workstation.

To protect encryption keys from disclosure when PTKTDATA records are listed, the SSKEY field never displays. A message similar to the following is displayed when a PTKTDATA record is listed:

```
SSIGNON / TSOSYS2 LAST CHANGED BY ACFUSER ON 04/04/04-14:07
          MULT-USE
```

MULT-USE | NOMULT-USE

A generated PassTICKET, to be accepted multiple times, within the allowed plus or minus 10 minutes.

Note: The PassTicket MULT-USE value should only be specified in secure environments where access to generate PassTickets is contained within a secure internal network.

Example

Create a program which uses the RCVTPTGN callable service, and run it on a system with an SMFID of SYS1. The purpose of your program is to generate a PassTicket for a user with a USERID of USERA, so that this user may logon to a TSO application on a system with an SMFID of SYS2.

In order to generate this PassTicket on SYS1, you would need to insert a TSOSYS2 profile record into your Infostorage database via the following commands from a workstation:

```
SET PROFILE(PTKTDATA) DIVISION(SSIGNON)
INSERT TSOSYS2 SSKEY(c237d18425cfe12d)
```

Recall that in order to generate a PassTicket you need to define a profile record that matches the APPLID of the target application the PassTicket is being generated for, in our case TSOSYS2.

As stated earlier, system SYS2 will also need to have knowledge of the SSKEY for the PassTicket validation to be successful. Issue the following commands to create a few profile records enabling a user to sign onto a host application (TSO on a system with an SMFID of SYS2) from a workstation:

```
SET PROFILE(PTKTDATA) DIVISION(SSIGNON)
INSERT TSOSYS2.USERA SSKEY(c237d18425cfe12d)
INSERT TSOSYS2.GRP1 SSKEY(88db3ea2824fa972)
INSERT TSOSYS2 SSKEY(c237d18425cfe12d)
```

After you create the profile records you need, issue the following command from the console to rebuild the directory for PTKTDATA records:

```
F ACF2,REBUILD(PTK),CLASS(P)
```

If USERA signs on with GRP1, the session key value from record TSOSYS2.USERA is evaluated first and, if the value is not valid, eTrust CA-ACF2 attempts to use the session key value from record TSOSYS2. All users signing on to TSOSYS2 with GRP1 use the session key value from record TSOSYS2.GRP1 and, if the value is not valid, eTrust CA-ACF2 attempts to use the session key value from record TSOSYS2. All other users use only the session key value from record TSOSYS2.

In the example the SSKEY for the TSOSYS2.USERA profile record will be tested first when USERA attempts to logon to TSOSYS2. This is according to the search order used to determine the profile record to be used during decryption, as described earlier in this section. As you can see the SSKEY used to generate the PassTicket is the same as the SSKEY used to decrypt and test the PassTicket. If the PassTicket has not expired USERA will be able to logon to TSOSYS2 successfully. Had the SSKEY not matched, validation would fail during the decryption, and the SSKEY from profile record TSOSYS2 would then be tested.

If you make changes to profile records at a later date, remember to reissue this REBUILD command; otherwise, the pointers to these new records do not exist and the changes are not recognized.

Example

The following is an example of inserting a PTKTDATA profile record with the MULT-USE attribute:

```
SET PROFILE(PTKTDATA) DIVISION(SSIGNON)
INSERT TSOSYS2 SSKEY(C237D18425CFE12D) MULT-USE
```

SDB2 Compiled Profile Record

In an MLS environment, eTrust CA-ACF2 SDB2 Compiled Profile Records are used to assign security labels to DB2 resources. To assign a security label to a DB2 resource, create an SDB2 Compiled Profile Record and, in each rule entry in the SECLABEL field, specify the record ID of an existing SECLABEL Profile Data Record (which is a defined security label), or a system defined label (SYSLOW, SYSHIGH, SYSNONE, SYSMULTI).

WARNING! *If you change or delete an existing security label, (for example, Seclabel Profile data record) that has been assigned to a DB2 resource, you may get unexpected results during MLS validation. Before changing or removing a security label from the system, check whether it has been assigned to any DB2 resource. If it has, confirm that the change or deletion is intended. If it is, make any necessary changes to SDB2 profile records that are using the security label. Likewise, if you delete a security level or category that is used in any existing security label, before removing the level or category from the system, confirm that the deletion is intended. If it is, make any necessary changes to existing security labels, and any SDB2 profile records that are using the security labels.*

For more information on how to implement MLS on a system, including defining, assigning and using security classifications, see the *Multilevel Security Planning Guide*.

Similar to resource rules, masking of entity names in SDB2 Compiled Profile Records is allowed, including in the $KEY. Masking eases administration by allowing a security label to be assigned to multiple DB2 resources at one time.

You can use the following ACF subcommands to administer and maintain this compiled profile record: SET, COMP, DECOMP, DELETE, STORE, and RECKEY.

COMPILE

The following is an example of the COMPILE syntax of an SDB2 Compiled Profile Record:

Command Syntax

```
Set PROFILE(SDB2) DIVISION(PROFILE)
COMPILE *
   $KEY(DB2 rsrcname| mask)
   [$RTYPE(typecode)]
   [$MODE(mode)]
   rsrcmask
   SECLABEL(seclabel)
STORE
```

Field Descriptions

$KEY(DB2 *rsrcname*|*mask*)

Specifies the DB2 resource name to which the security label applies. The $KEY field is required. A resource name is from one to 256 characters long, depending on if it is *qualified or unqualified*. Resource names can contain embedded blanks or spaces, and can also be masked.

- An unqualified resource name is a one- to 40-character string with no periods delimiting sections or qualifiers of the resource name. **If the resource name is longer than 29 characters, it must be masked**.
- A qualified resource name is in the format, qualifier1.qualifier2…, where the first qualifier is one to 40 characters long, followed by a period and one or more subsequent qualifiers. **If the first qualifier is longer than 29 characters, it must be masked**. All the qualifiers together make up the complete resource name. The maximum length for qualified resource names is 256-characters.

[$RTYPE(*typecode*)]

Specifies the three-character resource type for this compiled record. Resource type denotes a logical class of resources. Each resource type is identified by a type code that consists of three characters, preferably three, alphanumeric characters (for example, TBL for TABLE). The $RTYPE field is required.

If a corresponding eTrust CA-ACF2 CLASMAP record does not exist, which translates an eight-character resource class into a three-character resource type code, eTrust CA-ACF2 will use the first three characters of the resource name as the type code to validate MLS calls for the specified resource. For more detailed information on clasmap records, see the "Maintaining Global System Options Records" chapter.

WARNING! You cannot mask the $RTYPE statement for a resource type. eTrust CA-ACF2 will not allow the first character of an $RTYPE value for a resource type to be an asterisk (*) or dash (-) and any subsequent asterisks or dashes will be treated as actual character values in the $RTYPE field.

[$MODE(QUIET|LOG|WARN|MLS)]

Specifies the MLS mode you want eTrust CA-ACF2 to use when it validates MLS access to the specified DB2 resource classified by this compiled record. The value specified in the $MODE control statement overrides the mode in the CONTROL GSO MLSOPTS record. The $MODE field takes effect when access to the specified DB2 resource is denied for a user. Valid modes are:

- **$MODE(QUIET)**—Permits MLS accesses to classified resources. No SMF logging occurs.
- **$MODE(LOG)**—Permits MLS accesses to classified resources that normally would violate MLS validation rules. SMF loggings occur.
- **$MODE(WARN)**—Permits MLS accesses to classified resources that normally would violate MLS validation rules and sends a warning message to the user. SMF loggings occur.
- **$MODE(MLS)**—Prevents MLS accesses to classified resources that violate MLS validation rules. SMF loggings occur.

For more information on MLS global modes, see the "Maintaining Global System Options Records" chapter.

rsrcmask

Completes the DB2 resource name that is classified with a security label. Do not enclose the value in single quotes. This field is required.

Note: If there is a CLASMAP record for it, which has the MIXED indicator turned on, then the resource name is case sensitive.

SECLABEL(*seclabel*)
Specifies the security label used to validate MLS access to the DB2 resource. The *seclabel* value is the 1 to 8-character name of an existing SECLABEL Profile Data Record segment that contains the security label data.

WARNING! When specifying the field name, SECLABEL, do not abbreviate it in any form (such as SEC or SECL); otherwise, although the record may be successfully inserted, eTrust CA-ACF2 will not recognize that it exists.

DECOMP

The following is an example of the DECOMP syntax of an SDB2 Compiled Profile Record:

Command Syntax

```
DECOMP or LIST {*|DB2 rsrcname|Like(DB2 rsrcnamemask) RTYPE(rsrc-type)}
               [Into(dsn)]   [ALL]
```

Note: Use the RTYPE parameter to specify the 1 to 3-character resource type to which the profile record applies. For example, if you want to decompile an SDB2 Compiled Profile Record with $KEY(TESTRSR) and $RTYPE(ABC), issue the following command::

```
decomp testrsr rtype(abc)
```

The RTYPE parameter is required.

Note: You cannot specify a dash ('-') as a masking character in this field. However, you may specify an asterisk ('*') as a masking character. If you specify a fully-masked value, such as RTYPE(***), when issuing the DECOMP LIKE or LIST LIKE command, all profiles for resources that match the record key will be displayed, including profiles for data sets. The following command displays all profile records for data set and resource names that start with "X".

```
Decomp like(x-) rtype(***)
ACFAB072 PROFILE   X STORED BY QAMLSEC ON 09/10/04-0830
$KEY(X)
 - SECLABLE(TESTCOMP)
ACFAB051 TOTAL RECORD LENGTH= 145 BYTES, 3 PERCENT UTILIZED


ACFAB072 PROFILE AAAX STORED BY QAMLSEC ON 09/10/04-9:30
$KEY(X)
$MODE(QUIET)
$RTYPE(AAA)
 - SECLABEL(TESTCOMP)
ACFAB051 TOTAL RECORD LENGTH  145 BYTES, 3 PERCENT UTILIZED
```

DELETE

The following is an example of the DELETE syntax of an SDB2 Compiled Profile Record:

Command Syntax

```
DELETE {*|DB2 rsrcname|Like(DB2 rsrcnamemask) RTYPE(rsrc-type)}
```

Note: Use the RTYPE parameter to specify the 1 to 3-character resource type to which the profile record applies. For example, if you want to delete an SDB2 Compiled Profile Record with $KEY(TESTRSR) and $RTYPE(ABC), issue the following command:

```
delete testrsr rtype(abc)
```

The RTYPE parameter is required.

STORE

The following is an example of the STORE syntax of an SDB2 Compiled Profile Record:

Command Syntax

```
STORE
```

RECKEY

The RECKEY command can be used to insert and delete single entries without recompiling the entire record, since there can be many entries in a record (like in access and resource rules).

The following is an example of the RECKEY syntax of an SDB2 Compiled Profile Record:

Command Syntax

```
RECKEY {*|rsrcname|Like(mask) {ADD(profile-entry)|DELETE(profile-entry)
RTYPE(rsrc-type)}
```

Note: Use the RTYPE parameter to specify the 1 to 3-character resource type that the profile record applies. For example, if you want to add an entry to an SDB2 Compile Profile Record with $KEY(TESTRSR) and $RTYPE(TBL), issue the following command:

```
reckey testrsr add(somename seclabel(label1)) rtype(tbl)
```

The RTYPE parameter is required.

Creating an SDB2 Compiled Record

To create an SDB2 Compiled Record, you must have the SECURITY privilege in your logonid. Issue the following commands to create the record:

```
acf
 ACF
set profile(sdb2) div(profile)
 PROFILE
compile *
 ACFAB010 ACF PROFILE COMPILER ENTERED

 .  $key(testdb2)
 .  $rtype(tbl)
 .  $mode(log)
 .  - seclabel(SYSHIGH)
 .  end
 ACFAB051 TOTAL RECORD LENGTH= 145 BYTES, 3 PERCENT UTILIZED
 PROFILE
store
 ACF6D027 COMPILED RECORD PROFILE / TBLTESTDB2 STORED
 PROFILE
```

Viewing an SDB2 Compiled Record

To view an SDB2 Compiled Record, you must have the SECURITY privilege in your logonid. Issue the following commands to view the record:

```
acf
 ACF
set profile(SDB2) division(profile)
 PROFILE
decomp test rtype(tbl)
ACFAB072 PROFILE TBLTEST STORED BY M1ADMN ON 04/22/04-15:56
$KEY(TEST)
$RTYPE(TBL)
$MODE(MLS)
QEWRQER.-.ASDF.TESTCOLUMN3 SECLABEL(LABELA)
ACFAB051 TOTAL RECORD LENGTH= 163 BYTES, 3 PERCENT UTILIZED
 PROFILE
```

Changing an SDB2 Compiled Record

To change an SDB2 Compiled Record, you must have the SECURITY privilege in your logonid. Issue the following commands to change the record:

```
acf
 ACF
set profile(sdb2) div(profile)
 PROFILE
reckey testdb2 add(table1 seclabel(syshigh)) rtype(tbl)
 ACFAB072 PROFILE TBLTESTDB2 STORED BY ADMIN01 ON 08/02/04-14:18
 ACFAB010 ACF PROFILE COMPILER ENTERED

 ******** PROFILE TBLTESTDB2 STORED BY ADMIN01 ON 08/02/04-14:18
 $KEY(TESTDB2)
 $MODE(LOG)
 $RTYPE(TBL)
  - SECLABEL(SYSHIGH)
 TABLE1 SECLABEL(SYSHIGH)
 ACFAB051 TOTAL RECORD LENGTH= 164 BYTES, 4 PERCENT UTILIZED
 ACF60207 RULE P SDB PROFILE TBLTESTDB2 REPLACED
 PROFILE
```

Deleting an SDB2 Compiled Record

To delete an SDB2 Compiled Record, you must have the SECURITY privilege in your logonid. Issue the following commands to delete the record:

```
acf
 ACF
set profile(sdb2) div(profile)
 PROFILE
 delete testdb2 rtype(tbl)
 ACF6D073 PROFILE / TBLTESTDB2 RECORD DELETED
 PROFILE
```

Activating an SDB2 Compiled Record

To activate changes to an SDB2 Compiled Record, issue the following commands:

```
set control(gso)
CONTROL
change infodir types(r-psdb)
f acf2,refresh(infodir)
f acf2,rebuild(sdb),class(p)
```

SECLEVEL Profile Data Records

In a MLS environment, after determining what degrees of sensitivity and trust are necessary to the organization or parts of the organization, an authorized security administrator can create security levels, which are the hierarchical elements of security labels.

An eTrust CA-ACF2 SECLEVEL Profile Data Record segment defines a security level available in the system. You must define a separate record for each level you want to use in the system. You must define levels before you can define and assign security labels to users, data sets and resources.

WARNING! *If you change or delete an existing security label, (for example, Seclabel Profile data record) that has been assigned to users or resources, you may get unexpected results during MLS validation. Before changing or removing a security label from the system, check whether it has been assigned to any users or resources. If it has, confirm that the change or deletion is intended. If it is, make any necessary changes to user and resource Seclabel profile records that are using the security label. Likewise, if you delete a security level or category that is used in any existing security label, before removing the level or category from the system, confirm that the deletion is intended. If it is, make any necessary changes to existing security labels, and any user and resource Seclabel profile records that are using the security labels.*

For more information on how to implement MLS on a system, including defining, assigning, and using security classifications, see the *Multilevel Security Planning Guide.*

Command Syntax

```
{Insert|Change|List|Delete} seclevelname(seclevel-name)
```

Record ID	Fields
seclevel	NAME(*seclevel-name*)

Field Descriptions

seclevel

Specifies a one- to 3-character record ID, which is a number between 1 and 254 without leading zeros or internal spaces. The number specified is the numeric rank of a security level. This field is required. The value supplied places the level in the hierarchy of all levels. The higher the number, the higher the level. A security label with a particular level dominates labels, whose levels have lower values, except as further restricted by categories. You cannot assign the same value for more than one level for a system. You cannot use the CHANGE subcommand to modify the seclevel value. To change the value of a level, delete the SECLEVEL record and insert a new one.

The following are examples of valid and invalid security levels:

Valid Record IDs	Invalid Record IDs
1	01
5	005
10	010
254	255

Name*(seclevel-name)*

Specifies the unique, 1- to 255-character, alphanumeric name of a security level. The name is always uppercased. Internal spaces are allowed, however, any leading or trailing blanks are trimmed off of the specified name. The name may never begin with the letters 'SYS', since this may cause confusion with any existing or future system-defined security labels. This field is optional.

Creating a SECLEVEL Profile Data Record

To create a SECLEVEL Profile Data Record, you must have the SECURITY privilege in your logonid. Issue the following commands to create the record:

```
acf
 ACF
set profile(seclabel) division(seclevel)
 PROFILE
insert 200 name(top secret)
  SECLEVEL / 200 LAST CHANGED BY USERA ON 05/03/04-12:11 NAME(TOP SECRET)
insert 100 name(secret)
  SECLEVEL / 100 LAST CHANGED BY USERA ON 05/03/04-13:11 VALUE(SECRET)
insert 75 name(classified)
  SECLEVEL / 75 LAST CHANGED BY USERA ON 05/03/04-14:11 NAME(CLASSIFIED)
insert 25 name(unclassified)
  SECLEVEL / 25 LAST CHANGED BY USERA ON 05/03/04-15:11 NAME(UNCLASSIFIED)
PROFILE
```

Viewing a SECLEVEL Profile Data Record

To view a SECLEVEL Profile Data Record, you must have the SECURITY privilege in your logonid. Issue the following commands to view the record:

```
acf
 ACF
set profile(seclabel) division(seclevel)
 PROFILE
list like(-)
  SECLEVEL / 100 LAST CHANGED BY USERA ON 05/03/04-13:11 VALUE(SECRET)
  SECLEVEL / 200 LAST CHANGED BY USERA ON 05/03/04-12:11 NAME(TOP SECRET)
  SECLEVEL / 25 LAST CHANGED BY USERA ON 05/03/04-15:11 NAME(UNCLASSIFIED)
  SECLEVEL / 75 LAST CHANGED BY USERA ON 05/03/04-14:11 NAME(CLASSIFIED)
PROFILE
```

Changing a SECLEVEL Profile Data Record

To change a SECLEVEL Profile Data Record, you must have the SECURITY privilege in your logonid. Only the NAME field can be changed using the CHANGE command. However, the CHANGE LIKE command may not be used to change the NAME field, since this value *should be* unique, although eTrust CA-ACF2 does not **require** that this value be unique.

The CHANGE command can never be used to change the actual value of a security level, which is the SECLEVEL Profile Data Record ID. To change the value of a security level, delete the SECLEVEL record and insert a new one specifying a different number between 1 and 254.

Issue the following commands to change the record:

```
acf
 ACF
set profile(seclabel) division(seclevel)
 PROFILE
change 150 name(internal use only)
  SECLEVEL / 150 LAST CHANGED BY USERA ON 05/30/04-12:11 NAME(INTERNAL USE ONLY)
```

Deleting a SECLEVEL Profile Data Record

To delete a SECLEVEL Profile Data Record, you must have the SECURITY privilege in your logonid. Issue thefollowing commands to delete the record:

```
acf
 ACF
set profile(seclabel) division(seclevel)
 PROFILE
delete 100
 ACF6D073 SECLEVEL / 100 RECORD DELETED
 PROFILE
```

Activating a SECLEVEL Profile Data Record

To activate new or changed security levels, issue the following commands:

```
set control(gso)
 CONTROL
change infodir types(r-psec)
f acf2,refresh(infodir)
f acf2,rebuild(sec),class(p)
f acf2,mls
```

CATEGORY Profile Data Records

In a multilevel security (MLS) environment, after determining if it is necessary to isolate users, data and resources within the organization, an authorized security administrator can create categories, which are the optional, non-hierarchical elements of security labels. If security labels in your system will contain categories, you must define these records before you can define and assign security labels to users, data sets and resources.

An eTrust CA-ACF2 CATEGORY Profile Data Record segment defines a category available in the system. You must define a separate record for each category you want to use in the system.

WARNING! *If you change or delete an existing security label, (for example, Seclabel Profile data record) that has been assigned to users or resources, you may get unexpected results during MLS validation. Before changing or removing a security label from the system, check whether it has been assigned to any users or resources. If it has, confirm that the change or deletion is intended. If it is, make any necessary changes to user and resource Seclabel profile records that are using the security label. Likewise, if you delete a security level or category that is used in any existing security label, before removing the level or category from the system, confirm that the deletion is intended. If it is, make any necessary changes to existing security labels, and any user and resource Seclabel profile records that are using the security labels.*

For more information on how to implement MLS on a system, including defining, assigning and using security classifications, see the *Multilevel Security Planning Guide.*

Command Syntax

{Insert|List|Delete}*category-name*

Record ID	Fields
category-name	None

Field Descriptions

category-name

Specifies the one- to 32-character, unique, uppercase, alphanumeric name of a category in the system. The category name cannot contain internal spaces. Duplicate categories are not allowed. In addition, the category name may never begin with the letters 'SYS', since this may cause confusion with any existing or future system-defined security labels. This field is required. The maximum number of categories that can be defined is limited only by the size of the database. You cannot issue the CHANGE command to modify this record. To change a category, delete the CATEGORY record and insert a new one.

Creating a CATEGORY Profile Data Record

To create a CATEGORY Profile Data Record, you must have the SECURITY privilege in your logonid. Issue the following commands to create the record:

```
acf
 ACF
set profile(seclabel) division(category)
 PROFILE
insert humanresources
 CATEGORY / HUMANRESOURCES LAST CHANGED BY USERA ON 05/03/04-12:11 -
  NO DATA AVAILABLE
  PROFILE
insert finance
 CATEGORY / FINANCE LAST CHANGED BY USERA ON 05/03/04-12:11 - NO DATA AVAILABLE
 PROFILE
insert sales
 CATEGORY / SALES LAST CHANGED BY USERA ON 05/03/04-12:11 - NO DATA AVAILABLE
 PROFILE
insert development
 CATEGORY / DEVELOPMENT LAST CHANGED BY USERA ON 05/03/04-12:11 -
  NO DATA AVAILABLE
 PROFILE
```

Viewing a CATEGORY Profile Data Record

To view a CATEGORY Profile Data Record you must have the SECURITY privilege in your logonid. Issue the following commands to view the record:

```
acf
 ACF
set profile(seclabel) division(category)
 PROFILE
list like(-)
 CATEGORY / DEVELOPMENT LAST CHANGED BY USERA ON 05/03/04-12:11 -
  NO DATA AVAILABLE
 CATEGORY / FINANCE LAST CHANGED BY USERA ON 05/03/03-12:11 - NO DATA AVAILABLE
 CATEGORY / HUMANRESOURCES LAST CHANGED BY USERA ON 05/03/04-12:11 -
  NO DATA AVAILABLE
 CATEGORY / SALES LAST CHANGED BY USERA ON 05/03/04-12:11 - NO DATA AVAILABLE
  PROFILE
```

Changing a CATEGORY Profile Data Record

The CHANGE command cannot be used on a CATEGORY Profile record. To change a category's value, delete the CATEGORY record and insert a new one.

Deleting a CATEGORY Profile Data Record

To delete a CATEGORY Profile Data Record, you must have the SECURITY privilege in your logonid. Issue the following commands to delete the record:

```
acf
 ACF
set profile(seclabel) division(category)
 PROFILE
delete humanresources
 ACF6D073 CATEGORY / HUMANRESOURCES RECORD DELETED
 PROFILE
```

Activating a CATEGORY Profile Data Record

To activate the security categories, issue the following commands:

```
set control(gso)
 CONTROL
change infodir types(r-psec)
f acf2,refresh(infodir)
f acf2,rebuild(sec),class(p)
f acf2,mls
```

SECLABEL Profile Data Records

In an MLS environment, after defining and creating security levels and categories in the system, an authorized security administrator can define the security labels that will be assigned to users, data sets and resources.

The SECLABEL Profile Data Record defines the value of a security label. You must define this record before you can assign the security label to users, data sets and resources.

WARNING! *If you change or delete an existing security label, (for example, Seclabel Profile data record) that has been assigned to users or resources, you may get unexpected results during multilevel security (MLS) validation. Before changing or removing a security label from the system, check whether it has been assigned to any users or resources. If it has, confirm that the change or deletion is intended. If it is, make any necessary changes to user and resource Seclabel profile records that are using the security label. Likewise, if you delete a security level or category that is used in any existing security label, before removing the level or category from the system, confirm that the deletion is intended. If it is, make any necessary changes to existing security labels, and any user and resource Seclabel profile records that are using the security labels.*

For more information on how to implement MLS on a system, including defining, assigning and using security classifications, see the *Multilevel Security Planning Guide.*

Command Syntax

```
{Insert|Change|List|Delete} seclabel SECLEVEL(seclevel)
[CATEGORY(category1,…category50)]
[SYSTEMID(sysid1,…sysidn)]
{add|rep|del}
```

Record ID	Fields
seclabel □	SECLEVEL(*seclevel*) CATEGORY(*category1,…category50*) SYSTEMID(*sysid1,…sysidn*)

Field Descriptions

seclabel

Specifies the 1- to 8-byte, alphanumeric-national character name of a security label. The security label cannot start with the letters 'SYS'. Security labels that begin with 'SYS' are reserved for existing or future system-defined security labels. This field is required.

Note: To assign a security label to a resource, the security label record ID must be specified in the SECLABEL field of a SECLABEL Compiled Profile Record. To assign the security label to a user, the security label record ID must be specified in the SECLABEL or DEFLABEL fields of a User SECLABEL Profile Data Record.

Seclevel(*seclevel*)

Specifies the 1- to 3-character security level which is the record ID of an existing SECLEVEL Profile Data Record. The security level must be a number between 1 and 254 without leading zeros. This field is required.

Warning! Any seclevel specified must be a valid SECLEVEL Profile Data Record defined in the system. Otherwise, this security label will be ignored by the system at the time the security classification tables are built and any users or resources that have been assigned this security label and try to use it will be not be able to.

Category(*category1,...category50*)
Specifies the 1- to 32-character, alphanumeric names of 1 to 50 categories that are the record IDs of existing CATEGORY Profile data records. This field is optional and can be masked by using asterisk(*) or dash(-) masking characters. A comma or blank is the only valid delimiter between specified categories.

Warning! Any category specified must be a valid CATEGORY Profile Data Record defined in the system. Otherwise, this security label will be ignored by the system at the time the security classification tables are built and any users or resources that have been assigned this security label and try to use it will be not be able to.

Important! To implement roles-based classification (for example, classification using categories only), assign the same SECLEVEL in each security label when it is created. To properly implement security classification with categories only, the security level (rank) of all active security labels must be the same. Otherwise, this could cause problems with MLS validations on your system.

Systemid(*sysid1,...sysidn*)
Specifies one or more 1- to 4-character, alphanumeric system IDs on which this security label can be used, if the MLS option for use of system-specific security labels is active (MLSECBYS). However, if the option is inactive, this field is ignored during MLS validations, and this security label can be used on any system. This field is optional and can be masked by using asterisk(*) or dash(-) masking characters. If no system IDs are specified, by default, the security label will apply to all systems. A comma or blank is the only valid delimiter between specified system IDs. This field is limited only by the size of the record which is 4K.

Note: You must specify MLSECBYS on the GSO MLSOPTS record to limit the use of security labels to certain systems. See GSO MLSOPTS record.

Important! After activating MLSECBYS, you must rebuild the security classifications table with the command 'F ACF2,MLS'.

System-Defined Security Labels

eTrust CA-ACF2 provides the following system-defined security labels which are internal to eTrust CA-ACF2 and can never be directly created or modified by a user but can only be assigned to users, data sets and resources:

SYSHIGH
The highest security label in any system. It is comprised of the highest level in the system and all categories. Therefore, it dominates all security labels.

SYSLOW
The lowest security label in any system. It is comprised of the lowest level in the system and no categories. Therefore, it is dominated by all security labels.

SYSNONE
A security label used in a system when write-down is not allowed. It should be assigned only to non-sensitive data and resources to which everyone, regardless of their security label, must have access, such as catalogs. It compares equivalent to any other security label. This security label cannot be assigned to users.

SYSMULTI
A security label that is equivalent to all other defined security labels. It should be assigned only to servers that can properly isolate users and data based on security labels or UNIX directories that contain subdirectories and files at different security levels. This security label is usually not assigned to users, but there are some exceptions.

Creating a SECLABEL Profile Data Record

To create a SECLABEL Profile Data Record, you must have the SECURITY privilege in your logonid. Issue the following commands to create the record:

```
acf
 ACF
set profile(seclabel) division(seclabel)
 PRO FILE
insert labelaaa seclevel(150) category(humanresources,finance,sales)
 SECLABEL / LABELAAA LAST CHANGED BY M1ADMN ON 04/22/04-15:38
                     C ATEGORY(FINANCE HUMANRESOURCES SALES) SECLEVEL(150)
 PRO FILE
```

Viewing a SECLABEL Profile Data Record

To view a SECLABEL Profile Data Record, you must have the SECURITY privilege in your logonid. Issue the following commands to view the record:

```
acf
 ACF
set profile(seclabel) division(seclabel)
 PRO FILE
list labelaaa
 SECLABEL / LABELAAA LAST CHANGED BY M1ADMN ON 04/22/04-15:38
                     C ATEGORY(FINANACE HUMANRESOURCE SALES) SECLEVEL(150)
 PRO FILE
```

Changing a SECLABEL Profile Data Record

To change a SECLABEL Profile Data Record, you must have the SECURITY privilege in your logonid. Issue the following commands to change the record:

```
acf
 ACF
set profile(seclabel) division(seclabel)
 PRO FILE
change labelaaa seclevel(50) category(marketing)
 SECLABEL / LABELAAA LAST CHANGED BY M1ADMN ON 04/22/04-15:40
                     C ATEGORY(MARKETING)
                     S ECLEVEL(50)
 PRO FILE
```

Deleting a SECLABEL Profile Data Record

To delete a SECLABEL Profile Data Record, you must have the SECURITY privilege in your logonid. Issue the following commands to delete the record:

```
acf
 ACF
set profile(seclabel) division(seclabel)
 PRO FILE
delete labelaaa
 ACF 6D073 SECLABEL / LABELAAA RECORD DELETED
 PRO FILE
```

Activating a SECLABEL Profile Data Record

To activate new or changed security labels, issue the following commands:

```
set control(gso)
 CON TROL
change infodir types(r-psec)
f acf2,refresh(infodir)
f acf2,rebuild(sec),class(p)
f acf2,mls
```

SECLABEL Compiled Profile Records

In an MLS environment, there are two ways that security labels can be assigned to data, depending on whether or not write-down protection is enabled on a system and whether or not the data is being newly created or previously existed before write-down protection was enabled.

1. If the NOMLWRITE option in the CONTROL GSO MLSOPTS record is set and write-down is not allowed, when a data set is created, eTrust CA-ACF2 will assign to it the session security label of the user who created the data. The security label assigned is stored in the SECLABEL(DSN) unstructured Infostorage record, which can never be modified, only viewed or deleted. For more information, see the SECLABEL(DSN) Records section. Once data has been labeled in this way, to reclassify it by assigning a different security label, a security administrator must create a SECLABEL Compiled Profile record for the data set with the changed security label.
2. If the MLWRITE option in the CONTROL GSO MLSOPTS record is set and write-down is allowed, when data is created, eTrust CA-ACF2 will NOT assign to it a security label. Instead, if the data should be classified, a security administrator must create a SECLABEL Compiled Profile record for the data set.

   eTrust CA-ACF2 SECLABEL Compiled Profile Records are used to assign security labels to existing data sets and non-DB2 resources. To assign a security label to an existing data set or non-DB2 resource, create a SECLABEL Compiled Profile Record and, in each rule entry in the SECLABEL field, specify the record ID of an existing SECLABEL Profile Data Record (which is a defined security label), or a system-defined label (such as, SYSLOW, SYSHIGH, SYSNONE, SYSMULTI).

   WARNING! *If you change or delete an existing security label, (for example, Seclabel Profile data record) that has been assigned to users or resources, you may get unexpected results during multilevel security (MLS) validation. Before changing or removing a security label from the system, check whether it has been assigned to any users or resources. If it has, confirm that the change or deletion is intended. If it is, make any necessary changes to user and resource Seclabel profile records that are using the security label. Likewise, if you delete a security level or category that is used in any existing security label, before removing the level or category from the system, confirm that the deletion is intended. If it is, make any necessary changes to existing security labels, and any user and resource Seclabel profile records that are using the security labels.*

   For more information on how to implement MLS on a system, including defining, assigning, and using security classifications, see the *Multilevel Security Planning Guide*.

Similar to resource rules, masking of entity names in SECLABEL Compiled Record Profile Records is allowed, including in the $KEY. Masking eases administration by allowing a security label to be assigned to multiple data sets or resources at one time. To assign a security label to all data sets in a system, create a fully-masked $KEY value and data set name entry.

You can use the following ACF subcommands to administer and maintain this compiled profile record: SET, COMP, DECOMP, DELETE, STORE, and RECKEY.

COMPILE

The following is an example of the COMPILE syntax of a SECLABEL Compiled Profile Record:

Command Syntax

```
Set PROFILE(SECLABEL) DIVISION(PROFILE)
COMPILE *

   $KEY(dsname-high-level-index|rsrcname|mask)
   [$RTYPE(typecode)]
   [$MODE(mode)]
    dsname-mask|rsrcmask
    SECLABEL(seclabel)

STORE
```

Field Descriptions

$KEY(*dsname-high-level-index*|*rsrcname*|*mask*)

Specifies the name of the data set high-level index or resource name to which the security labels apply. The $KEY field is required.

A *data set name high-level index* is the first index of a data set name and can be from one to eight characters.

Warning! You cannot mask the $KEY statement for a data set. eTrust CA-ACF2 will not allow the first character of a $KEY value for a data set to be an asterisk (*) or dash (-) and any subsequent asterisks or dashes will be treated as actual character values in the $KEY field.

A*resource name* is from one to 256-characters long, depending on whether it is *qualified or unqualified*. Resource names can contain embedded blanks or spaces, and can also be masked.

- An *unqualified* resource name is a one to 40-character string with no periods delimiting sections or qualifiers of the resource name. **If the resource name is longer than 29 characters, it must be masked**.

- A *qualified* resource name is in the format, *qualifier1.qualifier2…*, where the first qualifier is one to 40-characters long, followed by a period and one or more subsequent qualifiers. **If the first qualifier is longer than 29 characters, it must be masked**. All the qualifiers together make up the complete resource name. The maximum length for qualified resource names is 256-characters.

[$RTYPE(*typecode*)]

Specifies the three-character resource type for this compiled record. Resource type denotes a logical class of resources. Each resource type is identified by a type code that consists of three characters, preferably three, alphanumeric characters (for example, TBL for TABLE). The $RTYPE field is required only for resources. If the $RTYPE field is omitted, then the compiled record is recognized by eTrust CA-ACF2 as applying to data sets.

If the $RTYPE field is specified, and a corresponding eTrust CA-ACF2 CLASMAP record does not exist, which translates an eight-character resource class into a three-character resource type code, eTrust CA-ACF2 will use the first three characters of the resource name as the type code to validate MLS calls for the specified resource. For more information on CLASMAP records, see the "Maintaining Global System Options Records" chapter.

WARNING! You cannot mask the $RTYPE statement for a resource type. eTrust CA-ACF2 will not allow the first character of an $RTYPE value for a resource type to be an asterisk (*) or dash (-) and any subsequent asterisks or dashes will be treated as actual character values in the $RTYPE field.

[$MODE(QUIET|LOG|WARN|MLS)]

Specifies the MLS mode you want eTrust CA-ACF2 to use when it validates MLS access to the data set or resource classified by this compiled record. The value specified in the $MODE control statement overrides the mode in the CONTROL GSO MLSOPTS record. The $MODE field takes effect when access to the data set or resource is denied for a user. Valid modes are:

- **$MODE(QUIET)**—Permits MLS accesses to classified data sets and resources. No SMF logging occurs.
- **$MODE(LOG)**—Permits MLS accesses to classified data sets and resources that normally would violate MLS validation rules. SMF loggings occur.
- **$MODE(WARN)**—Permits MLS accesses to classified data sets and resources that normally would violate MLS validation rules and sends a warning message to the user. SMF loggings occur.
- **$MODE(MLS)**—Prevents MLS accesses to classified data sets and resources that violate MLS validation rules. SMF loggings occur.

For more information on MLS global modes, see the "Maintaining Global System Options Records" chapter.

dsname-mask | rsrcmask
Completes the name of the data set or resource name that is classified with a security label. Do not enclose the value in single quotes. This field is required.

A data set name can have from one to 22 levels of qualifiers. Each level must begin with an alphabetic character or the character @, $, or #. You can specify up to 8-characters per level. The entire data set name, including periods, can contain up to 44-characters.

Note: If there is a resource and there is a CLASMAP record for it that has the the MIXED indicator turned on, then the resource name is case sensitive.

SECLABEL(*seclabel*)
Specifies the security label used to validate MLS access to the data set or resource. The *seclabel* value is the 1 to 8-character name of an existing SECLABEL Profile Data Record segment that contains the security label data.

WARNING! When specifying the field name, SECLABEL, do not abbreviate it in any form (such as SEC or SECL); otherwise, although the record may be successfully inserted, eTrust CA-ACF2 will not recognize that it exists.

DECOMP

The following is an example of the DECOMP syntax of a SECLABEL Compiled Profile Record:

Command Syntax

```
DECOMP or LIST {*|dsname-high-level-index|rsrcname|mask|Like(mask)}
               [RTYPE(rsrc-type)]
               [Into(dsn)]
               [ALL]
```

Note: Use the RTYPE parameter to specify the 1 to 3-character resource type to which the profile record applies. For example, if you want to decompile a SECLABEL Compiled Profile Record with $KEY(TESTRSR) and $RTYPE(ABC), issue the following command:

```
decomp testrsr rtype(abc)
```

If the RTYPE parameter is not specified, then it is assumed that the record applies to a data set.

Note: You cannot specify a dash ('-') as a masking character in this field. However, you may specify an asterisk ('*') as a masking character. If you specify a fully-masked value, such as RTYPE(***), when issuing the DECOMP LIKE or LIST LIKE command, all profiles for resources that match the record key will be displayed, including profiles for data sets. The following command displays all profile records for data set and resource names that start with "X".

```
Decomp like(x-) rtype(***)
ACFAB072 PROFILE   X STORED BY QAMLSEC ON 09/10/04-0830
$KEY(X)
 - SECLABLE(TESTCOMP)
ACFAB051 TOTAL RECORD LENGTH= 145 BYTES, 3 PERCENT UTILIZED


ACFAB072 PROFILE AAAX STORED BY QAMLSEC ON 09/10/04-9:30
$KEY(X)
$MODE(QUIET)
$RTYPE(AAA)
 - SECLABEL(TESTCOMP)
ACFAB051 TOTAL RECORD LENGTH  145 BYTES, 3 PERCENT UTILIZED
```

DELETE

The following is an example of the DELETE syntax of a SECLABEL Compiled Profile Record:

Command Syntax

```
DELETE {*|dsname-high-level-index|rsrcname|Like(mask)}
       [RTYPE(rsrc-type)]
```

Note: Use the RTYPE parameter to specify the 1 to 3-character resource type to which the profile record applies. For example, if you want to delete a SECLABEL Compiled Profile Record with $KEY(TESTRSR) and $RTYPE(ABC), issue the following command:

```
delete testrsr rtype(abc)
```

If the RTYPE parameter is not specified, then it is assumed that the record applies to a data set.

STORE

The following is an example of the STORE syntax of a SECLABEL Compiled Profile Record:

Command Syntax

```
STORE
```

RECKEY

The RECKEY command can be used to insert and delete single entries without recompiling the entire record, since there can be many entries in a record (like in access and resource rules).

The following is an example of the RECKEY syntax of a SECLABEL Compiled Profile Record:

Command Syntax

```
RECKEY {*|dsname|rsrcname|Like(mask) {ADD(profile-entry)|DELETE(profile-entry)
RTYPE(rsrc-type)}
```

Note: Use the RTYPE parameter to specify the 1 to 3-character resource type to which the profile record applies. For example, if you want to add an entry to a SECLABEL Compile Profile Record with $KEY(TESTRSR) and $RTYPE(ABC), issue the following command:

```
Reckey testrsr add(somename seclabel(label1)) rtype(abc)
```

If the RTYPE parameter is not specified, then it is assumed that the record applies to a data set.

Creating a SECLABEL Compiled Record

To create a SECLABEL Compiled Record, you must have the SECURITY privilege in your logonid. Issue the following commands to create the record:

```
acf
 ACF
set profile(seclabel) division(profile)
 PROFILE
compile *
 ACFAB010 ACF PROFILE COMPILER ENTERED
 .  $KEY(sys1)
 .  $MODE(MLS)
 .   MAN- SECLABEL(SYSHIGH)
 .
 ACFAB051 TOTAL RECORD LENGTH= 145 BYTES, 3 PERCENT UTILIZED
 PROFILE
store
 ACF6D027 COMPILED RECORD PROFILE / SYS1 REPLACED
 PROFILE
```

Viewing a SECLABEL Compiled Record

To view a SECLABEL Compiled Record, you must have the SECURITY privilege in your logonid. Issue the following commands to view the record:

```
acf
 ACF
set profile(seclabel) division(profile)
 PROFILE
decomp sys1
 ACFAB072 PROFILE SYS1 STORED BY M1ADMN ON 08/02/04-14:18
 $KEY(SYS1)
 $MODE(MLS)
  MAN- SECLABEL(SYSHIGH)
 ACFAB051 TOTAL RECORD LENGTH= 145 BYTES, 3 PERCENT UTILIZED
 PROFILE
```

Changing a SECLABEL Compiled Record

To change a SECLABEL Compiled Record, you must have the SECURITY privilege in your logonid. Issue the following commands to change the record:

```
acf
 ACF
set profile(seclabel) division(profile)
 PROFILE
reckey sys1 add(dump1 seclabel(syshigh))
 ACFAB072 PROFILE SYS1 STORED BY M1ADMN ON 08/02/04-14:18
 ACFAB010 ACF PROFILE COMPILER ENTERED

 ******** PROFILE SYS1 STORED BY M1ADMN ON 08/02/04-14:18
 $KEY(SYS1)
 $MODE(MLS)
  DUMP1 SECLABEL(SYSHIGH)
  MAN- SECLABEL(SYSHIGH)
 ACFAB051 TOTAL RECORD LENGTH= 164 BYTES, 4 PERCENT UTILIZED
 ACF60207 RULE P SEC PROFILE SYS1 REPLACED
 PROFILE
```

Deleting a SECLABEL Compiled Record

To delete a SECLABEL Compiled Record, you must have the SECURITY privilege in your logonid. Issue the following commands to delete the record:

```
acf
 ACF
set profile(seclabel) division(profile)
 PROFILE
 delete sys1
 ACF6D073 PROFILE / SYS1 RECORD DELETED
 PROFILE
```

Activating a SECLABEL Compiled Record

To activate changes to a SECLABEL Compiled Record, issue the following commands:

```
set control(gso)
CONTROL
change infodir types(r-psec)
f acf2,refresh(infodir)
f acf2,rebuild(sec),class(p)
```

SECLABEL(DSN) Records

If the NOMLWRITE option in the CONTROL GSO MLSOPTS record is set and write-down is not allowed, when data is created, eTrust CA-ACF2 will assign to it the session security label of the user who created the data set. The security label assigned is stored in the SECLABEL(DSN) unstructured Infostorage record, which can never be modified, only viewed or deleted. This record will be automatically deleted when the data set is deleted. Once data has been labeled in this way, to reclassify it by assigning a different security label, a security administrator must create a SECLABEL Compiled Profile record for the data set with the changed security label.

WARNING! *If you change or delete an existing security label, (for example, Seclabel Profile Data Record) that has been assigned to users or resources, you may get unexpected results during MLS validation. Before changing or removing a security label from the system, check whether it has been assigned to any users or resources. If is has, confirm that the change or deletion is intended. If it is, make any necessary changes to user and resource Seclabel profile records that are using the security label. Likewise, if you delete a security level or category that is used in any existing security label, before removing the level or category from the system, confirm that the deletion is intended. If it is, make any necessary changes to existing security labels, and any user and resource Seclabel profile records that are using the security labels.*

Command Syntax

```
{List|List Like(recidmask)}
{Delete|Delete Like(recidmask)}dsname
```

Record ID	Fields
dsname	SDSN(*data-set-name*) SLABEL(*seclabel*)

Field Descriptions

dsname
Specifies the name of the z/OS data set that eTrust CA-ACF2 assigned a security label when the data set was created.

SDSN
Specifies the name of the z/OS dataset that eTrust CA-ACF2 assigned a security label when the data set was created. The value in this field is also the same as the record ID.

SLABEL
Specifies the security label that eTrust CA-ACF2 assigned to the data set.

Note: A security administrator can issue DELETE or LIST commands to delete or view the security label that eTrust CA-ACF2 assigned to a new data set in the SECLABEL(DSN) unstructured Infostorage record. However, INSERT and CHANGE commands cannot be used to administer this record. Instead, to reclassify the data set by changing its security label, a security administrator must create a SECLABEL Compiled Profile record for the data set with the desired security label.

Viewing a SECLABEL(DSN) Record

To view a SECLABEL(DSN) Record, you must have the SECURITY privilege in your logonid. Issue the following commands to view the record:

```
acf
 ACF
set seclabel(dsn)
 SECLABEL
list MLSUSER.MINI.JCLLIB5
 MLSUSER.MINI.JCLLIB5 LAST CHANGED BY MLSUSER ON 08/03/04-13:52
                      SDSN(MLSUSER.MINI.JCLLIB5) SLABEL(SYSLOW)
```

Deleting a SECLABEL(DSN) Record

To delete a SECLABEL(DSN) Record, you must have the SECURITY privilege in your logonid. Issue the following commands to delete the record:

```
acf
 ACF
set seclabel(dsn)
 SECLABEL
delete MLSUSER.MINI.JCLLIB5
ACF6D072 MLSUSER.MINI.JCLLIB5 RECORD DELETED
```

SYSMVIEW Profile Records

eTrust CA-ACF2 support of IBM's SystemView for z/OS requires SYSMVIEW profile records, SVFMR profile data records and resource rules.

eTrust CA-ACF2 SYSMVIEW profile is a compiled profile record that identifies a SVFMR profile data record or segment. The SVFMR profile data record defines the SystemView scripts and parameters that are used when the application is accessed through SystemView. The SVFMR record can also reference a PTKTDATA profile record that is used when generating a PassTicket for the user.

The compiled record allows ease of administration through masking of the profile name and allowing different profile names to point to the same data record. Thus when one data record is modified, all applicable profile names access the changed data.

The SYSMVIEW compiled record key consists of program name, system name, and an optional userid as follows:

```
pgmname.sysname.userid
```

The name is maskable in the same manner as other compiled profile records.

To create a SYSMVIEW profile record, issue the following commands:

```
SET PROFILE(SYSMVIEW) DIvision(PROFILE)

COMPILE *.

$KEY(pgmname)
sysname.userid SVFMR(datarec)

STORE
```

You can use the ACF subcommand RECKEY to maintain this compiled profile record.

SVFMR Data Profile Records

The SVFMR data segment contains the script and parameter definitions needed by SystemView. The fields of the SVFMR record follow:

Record ID	Fields
datarec	APPLDATA(*appldata*) SCRIPTN(*scriptname*) PARMN(*parmname*)

Field Descriptions

datarec
The record ID referenced by the SVFMR keyword in the SYSMVIEW profile record.

APPLDATA(*appldata*)
APPLDATA contains the name of the PTKTDATA profile record that is used when a PassTicket is generated as part of the Secured Signon feature. See PTKTDATA Profile Records for more information about the PassTicket Process.

SCRIPTN(*scriptname*)
The eight-character name of the PDS member that contains the script to be used for access to this application.

PARMN(*parmname*)
The eight-character name of the PDS member that contains the parameter information for the associated script.

Example

The following example shows how to define the profile records for SYSMVIEW information to the CONSOLE application. The production system, PROD1, has both script parameters and APPLDATA defined; the APPLDATA points to a PTKTDATA profile record. Any test systems that begin with the characters TEST use different script parameters and the default PTKTDATA profile name.

```
SET PROFILE(SYSMVIEW) DIvision(PROFILE)

COMPILE *.
$KEY(CONSOLE)
 PROD1.- SVFMR(SVPROD)

 TEST-.- SVFMR(SVTEST)

STORE

SET PROFILE(SYSMVIEW) DIV(SVFMR)
INSERT SVPROD APPLDATA(CONSPT) SCRIPTN(SCRIPT1) PARMN(PARM1)
INSERT SVTEST SCRIPTN(SCRIPT2) PARMN(PARM2)
```

SYSMVIEW Resource Rules

The SYSMVIEW resource rules define which users can access which applications on a particular system. The resource rules use type code SMV. The rule $KEY consists of program name, system name, and an optional userid as follows:

```
pgmname.sysname.userid
```

Note: You cannot mask the $KEY or use the extended rule format. These resource rules must be resident.

In the following example, certain users can access the CONSOLE application on the production system, PROD1, using the PRODUSER userid for their access. On the test system, TEST1, anyone can access the application using their own userid.

```
SET Resource(SMV)

COMPILE *

$KEY(CONSOLE.PROD1.PRODUSER) TYP(SMV)
 UID(sysprog) ALLOW

STORE

COMPILE *

$KEY(CONSOLE.TEST1) TYP(SMV)
 UID(-) ALLOW

STORE
```

To make the rules resident, add the SMV resource type to the GSO INFODIR record:

```
SET Control(GSO)

CHANGE INFODIR TYPES(R-RSMV)
```

Any changes to the SMV resource rules requires that you issue the following operator command to rebuild the rules:

```
F ACF2,REBUILD(SMV)
```

Using the ISPF Panels

To administer profile records, select option 14 PROFILE on the eTrust CA-ACF2 ISPF Option Selection Menu.

```
---------------- eTrust CA-ACF2 ISPF OPTION SELECTION MENU ---------------------
OPTION  ===>

   1  RULES     - PROCESS eTrust CA-ACF2 ACCESS AND GENERALIZED RESOURCE RULES
   2  LOGONIDS  - eTrust CA-ACF2 LOGONID CREATION/MAINTENANCE FACILITY
   3  SYSTEM    - eTrust CA-ACF2 SHOW COMMANDS
   4  REPORTS   - eTrust CA-ACF2 REPORT PROGRAM PROCESSOR
   5  UTILITIES - PROCESS eTrust CA-ACF2 UTILITIES
   6  GSO       - GLOBAL SYSTEM OPTIONS SERVICES
   7  NET       - NETWORKING SYSTEM OPTIONS SERVICES
   8  CAC       - MVS DATABASE CACHE RECORD SERVICES
   9  XREF      - SOURCE/RESOURCE AND GROUP RECORD SERVICES
  10  MAC       - MANDATORY ACCESS CONTROL ADMINISTRATION
  11  CPF       - COMMAND PROPAGATION FACILITY SERVICES
  12  FIELD     - RECORD LEVEL PROTECTION CONTROLS
  13  TARGETS   - SET CPF TARGET NODES, DEFAULTS IN USE
  14  PROFILE   - PROCESS PROFILE INFORMATION RECORDS
  15  SMS       - PROCESS DFSMS SUPPORT RECORDS
  16  ENTRY     - PROCESS ENTRY SOURCE RECORDS
  17  SHIFT     - PROCESS SHIFT/ZONE RECORDS
  18  RACDCERT  - PROCESS KEYRING/CERTIFICATE COMMANDS
  19  C-CIC     - PROCESS C-CIC CICS INITIALIZATION RECORDS
```

The eTrust CA-ACF2 Profile Record Selection panel is displayed. Indcate the profile record you want to process from this panel and press Enter:

```
------------ eTrust CA-ACF2 Security PROFILE RECORD SELECTION ----------------
COMMAND ===>

  USER:
    CERTDATA  ===>         CICS      ===>         DCE       ===>
    EIM       ===>         KERB      ===>         KERBLINK  ===>
    KEYRING   ===>         LANGUAGE  ===>         LINUX     ===>
    LNOTES    ===>         NDS       ===>         NETVIEW   ===>
    OMVS      ===>         OPERPARM  ===>         PROXY     ===>
    SECLABEL  ===>         WORKATTR  ===>         PASSWORD  ===>
  GROUP:
    LINUX     ===>         OMVS      ===>
  DATASET:
    DFP       ===>
  GENERAL:
    APPCLU    ===>         DLFCLASS  ===>         KEYSMSTR  ===>
    EIM       ===>         PROXY     ===>         PTKTDATA  ===>
    SYSMVIEW  ===>
  MULTILEVEL SECURITY (MLS):
    SECLEVEL  ===>         CATEGORY  ===>         SECLABEL  ===>
    SDB2      ===>

PRESS ANY NON-BLANK KEY TO SELECT AN OPTION
OR PRESS END TO REDISPLAY PREVIOUS PANEL.
```

The Profile Record Selection panel is displayed. Indcate the profile record you want to process from this panel and press Enter: Use this panel to identify the record you want to administer and the processing option you want to perform on that record.

The following sample panels assume you are administering UNIX System Services (OMVS) records.

```
-------------------------- PROFILE RECORD SELECTION --------------------------
COMMAND ===>


  PROCESSING GROUP PROFILE RECORDS


  PROCESSING OPTION:

                   I - INSERT PROFILE RECORDS
                   C - CHANGE PROFILE RECORDS
                   D - DELETE PROFILE RECORDS
                   L - LIST PROFILE RECORDS


     RECORD KEY    ===>

     MASKED KEY    ===>

     USING KEY     ===>
```

Use the following panel to assign a user identification (UID) number to an UNIX System Services user:

```
------------------------ PROFILE RECORD ADMINISTRATION --------------- --------
 COMMAND ===>

  INSERT OMVS PROFILE RECORDS

  UID        ==>              UID value or 'AUTOUID'


 PRESS ENTER TO CONTINUE OR END TO CANCEL.
```

To modify a UID, select option C on the Profile Record Selection panel and then complete the following panel to change an existing user identification (UID) number.

```
------------------------ PROFILE RECORD ADMINISTRATION -----------------------
COMMAND ===>

 CHANGE OMVS PROFILE RECORDS

 UID        ==>           UID value or 'AUTOUID'


PRESS ENTER TO CONTINUE OR END TO CANCEL.
```

Use this panel to specify the initial directory pathname UNIX System Services is to use when the user enters the OMVS command:

```
------------------------ PROFILE RECORD ADMINISTRATION -----------------------
COMMAND ===>

 INSERT OMVS PROFILE RECORDS

 HOME       ==>


PRESS ENTER TO CONTINUE OR END TO CANCEL.
```

Use this panel to change existing HOME information:

```
------------------------ PROFILE RECORD ADMINISTRATION -----------------------
COMMAND ===>

 CHANGE OMVS PROFILE RECORDS
  (ENTER A DASH TO CLEAR THIS FIELD)

 HOME       ==>

PRESS ENTER TO DISPLAY NEXT PANEL OR END TO REDISPLAY PREVIOUS PANEL
```

Use the following panel to specify a group identification (GID) number to an UNIX System Services user.

```
------------------------ PROFILE RECORD ADMINISTRATION -----------------------
COMMAND ===>

 INSERT GROUP PROFILE RECORDS

 Specify either a GID value or AUTOGID

 GID        ==>                Group value or 'AUTOGID'


PRESS ENTER TO CONTINUE OR END TO CANCEL.
```

Use this panel to modify an existin group identification (GID) number:

```
------------------------ PROFILE RECORD ADMINISTRATION -----------------------
COMMAND ===>

 CHANGE GROUP PROFILE RECORDS

 GID        ==>              Group value or 'AUTOGID'


PRESS ENTER TO CONTINUE OR END TO CANCEL.
```

To process another profile record, return to the Profile Record Selection panel and enter the appropriate information. The sample panels that follow assume you are inserting and changing a user identification (UID) number for UNIX System Services.

To create the UID, select I and specify the record key on the following panel. Press Enter.

```
------------------------ PROFILE RECORD SELECTION ----------------------------
COMMAND ===>

 PROCESSING OMVS PROFILE RECORDS

 PROCESSING OPTION:

                          I  - INSERT PROFILE RECORDS
                          C  - CHANGE PROFILE RECORDS
                          D  - DELETE PROFILE RECORDS
                          L  - LIST PROFILE RECORDS

        RECORD KEY       ===>

        MASKED KEY       ===>

        USING KEY        ===>
```

Use this to specify the user's UNIX System Services shell program:

```
------------------------ PROFILE RECORD ADMINISTRATION -----------------------
COMMAND ===>

 INSERT OMVS PROFILE RECORDS

 OMVSPGM    ==>

PRESS ENTER TO DISPLAY NEXT PANEL OR END TO REDISPLAY PREVIOUS PANEL
```

Or use this panel to change the PROGRAM information:

```
------------------------ PROFILE RECORD ADMINISTRATION -----------------------
COMMAND ===>

 CHANGE OMVS PROFILE RECORDS
   (ENTER A DASH TO CLEAR THIS FIELD)

 OMVSPGM    ==>


  PRESS ENTER TO PROCESS RECORD(S) OR END TO REDISPLAY PREVIOUS PANEL
```

Use this panel to specify CPUTIME, ASSIZE, FILEPROC, PROCUSER, THREAD, and MMAPAREA information that identify UNIX System Services for a user:

```
------------------------ PROFILE RECORD ADMINISTRATION -----------------------
COMMAND ===>

 INSERT OMVS PROFILE RECORDS

      CPUTIME    ==>

      Assize     ==>

      FILEPROC   ==>

      PROCUSER   ==>

      THREAD     ==>

      MMAPAREA   ==>

PRESS ENTER TO CONTINUE OR END TO CANCEL
```

Or use this panel to change CPUTIME, ASSIZE, FILEPROC, PROCUSER, THREAD, and MMAPAREA information:

```
------------------------ PROFILE RECORD ADMINISTRATION -----------------------
COMMAND ===>


 CHANGE OMVS PROFILE RECORDS
  (ENTER A DASH TO CLEAR THIS FIELD)


      CPUTIME    ==>

      Assize     ==>

      FILEPROC   ==>

      PROCUSER   ==>

      THREAD     ==>

      MMAPAREA   ==>


PRESS ENTER TO CONTINUE OR END TO CANCEL.
```

Use this panel to indicate whether you want to process compiled profile records or the segment data contained in profile data records:

```
------------------ eTrust CA-ACF2 PROFILE RECORD PROCESSING MENU
--------------------
OPTION   ===>

   SELECT ONE OF THE FOLLOWING OPTIONS TO PROCESS PROFILE RECORD:

   1.  PROFILE RECORD -
        TO CREATE, MAINTAIN OR DISPLAY PROFILE RECORD

   2.  PROFILE DATA RECORD -
        TO CREATE, MAINTAIN OR DISPLAY PROFILE DATA RECORD
```

Using the ACF Command

Once within the ACF command, use the SET command to enter profile administration. Enter:

```
SET PROFILE(USER) DIVISION(PROFILE|profile-data-name)
```

Where PROFILE can be one of the following: USER, GROUP, DATASET, KEYSMSTR, DLFCLASS, APPCLU, PTKTDATA, or SYSMVIEW. DIVISION can be PROFILE, which indicates a compiled record, or the profile data name of the data record. Remember that compiled records point to the profile data records.

Define a USER profile record as follows:

```
SET PROFILE(USER) DIVISION(WORKATTR)

INSERT userid WANAME(username)
```

Define a DATASET profile record as follows:

```
SET PROFILE(DATASET) DIVISION(PROFILE)

COMPILE *

$KEY(hlq)
data.set.name DFP(ownrname)

STORE

SET PROFILE(DATASET) DIVISION(DFP)

INSERT ownrname RESOWNER(resowner)
```

Use the following commands for all user profile records:

```
LIST *|recid|LIKE(recidmask)

DELETE *|recid|LIKE(recidmask)
```

Profile data records use the following commands:

```
INSERT *|recid|USING(urecid) recid

CHANGE recid|LIKE(urecid)
```

Compiled records use the COMPILE command. The keys of both the compiled records and the profile data records can be masked when these records are defined in the eTrust CA-ACF2 GSO INFODIR record. The infostorage class for profile records is PROFILE; the letter representing the class is P. eTrust CA-ACF2 distributes the following CLASMAP resource type definitions with the product; adjustments might be needed if your site uses different types for these resources.

Profile record	Resource Type
APPCLU	ALU
DATASET	DSN
DLFCLASS	DLF
GROUP	GRP
KEYSMSTR	KEY
PTKTDATA	PTK
SDB2	SDB
SECLABEL	SEC
SYSMVIEW	SMV
USER	USR

All profiles should be defined as resident in the GSO INFODIR record. To accomplish this, issue the following subcommands:

```
SET Control(GSO)
 CONTROL
CHANGE INFODIR TYPES(R-PDLF,R-RDLF,R-PGRP,R-PDSN,R-PALU,R-PSEC,R-PSDB,
R-PUSR,R-PPTK,R-RSMV)
```

Do not add R-PKEY to this list; its use is optional.

This example assumes you are using the types distributed with eTrust CA-ACF2; if you are using different types, make the appropriate adjustments. These directories do not appear when you issue a SHOW RESIDENT until they have been added to the GSO INFODIR record and that record has been refreshed.

When records are resident in storage, changes are made effective immediately by issuing a console rebuild command. For example:

```
F ACF2,REBUILD(USR),CLASS(P)
```

Using the RECKEY Subcommand

The RECKEY subcommand assists security administrators in maintaining rule sets and compiled infostorage rule records. This subcommand allows the user to decompile, add or delete a rule entry, recompile, and store the updated rule set on one command. This command can be used in any ACF mode that handles compiled records and it executes on other CPF-defined nodes.

The syntax of the RECKEY subcommand is:

```
RECKEY ruleid {ADD(rule-entry)|DELETE(rule-entry)}
```

For example, the following sequence shows how to add one additional rule line to an access rule using the RECKEY subcommand:

```
ACF
SET RESOURCE(CKC)
RECKEY PAYM ADD(HRS UID(HRS-) ALLOW))
```

The RECKEY subcommand takes the following parameters:

ruleid
Specifies the key of the rule set being modified.

ADD|DELETE
Specifies the function to be performed. ADD inserts a rule line or entry. DELETE removes a rule line or entry.

rule-entry
Indicates the actual rule line or entry to be inserted or removed.

OPERANDS-RULEID
Specifies the key of the rule set being modified.

CLASS(,)
The generalized resource class. If specified, a TYPE() must also be specified. If not specified and type is specified, CLASS(R) is assumed.

TYPE(…)
The generalized resource class. If not specified the type from the set command is used.

SYSID(..,)
Up to 8-character SYSID. For use with externally compiled records that use a SYSID such as DB2 rules. You must specify the SYSID parameter when using the RECKEY command for eTrust CA-ACF2 for DB2.

RULE-LINE
Indicates the actual rule line or entry to be inserted or deleted.

LIST|NOLIST
Causes the input to the compiler to be displayed on your screen. NOLIST causes no such display. LIST is the default.

NOVERIFY|VERIFY
Prompts to verify deletion of rule during delete processing. NOVERIFY is the default.

For descriptions of the syntax for the INSERT, LIST, CHANGE, and DELETE subcommands, see the "Maintaining Structured Infostorage Records" chapter. For descriptions of the syntax for the COMPILE, DECOMPILE, and STORE commands, see the "Maintaining Access Rules" chapter.

Chapter

16

Implementing DFSMS Support

This chapter provides the information required for using eTrust CA-ACF2 to manage the System Authorization Facility (SAF) security calls issued by the IBM DFSMS product series. This information clarifies the eTrust CA-ACF2 requirements used to secure all DFSMS functions and to make planning for the DFSMS family easier. The *IBM System-Managed Storage Migration Planning Guide* advises that a SAF-compatible external security product, like eTrust CA-ACF2, eTrust CA-Top Secret, or RACF, is required to protect DFSMS functions. The philosophy of eTrust CA-ACF2 and eTrust CA-Top Secret to be SAF-compatible enables this use of SAF classes to smoothly integrate into eTrust CA-ACF2. The resource names and the associated validations are described in the *IBM System-Managed Storage Migration Planning Guide*.

The ACF commands needed to define and manage the new DFSMS resources and to enable them to be used by selected users are described in this chapter. There are no RACF dependencies in the new SAF resource definition or the validations performed against them.

The recommendations made in this chapter represent the minimum default specifications you need for establishing DFSMS security using eTrust CA-ACF2. See IBM DFSMS documentation if you have additional security considerations beyond the scope of this discussion.

What is DFSMS?

DFSMS is an IBM designation for the DF/HSM, DFDSS, DFSORT, DFRMM, and RACF products when used in a DFSMS system.

The Data Facility family and RACF provide complementary functions that make up the Data Facility Storage Management Subsystem (DFSMS). As with all previous and planned DFSMS releases, the DFSMS components (DFHSM, DFDSS, DFSORT, DFRMM and RACF), are complementary, but separate products that use MVS/DFSMS services. Although RACF has also been designated as part of DFSMS, RACF is **not** a DFSMS product. Furthermore, no DFSMS products require RACF. All SAF-compatible security systems, like eTrust CA-ACF2, eTrust CA-Top Secret, and RACF, provide equivalent functionality in the DFSMS environment.

DFSMS is the foundation of the Storage Management Subsystem. In releases earlier than MVS/DFP Version 3, basic data management facilities are provided for data set allocation and the enforcement of data and storage attributes. In DFP Version 3, several significant enhancements have been added to simplify catalog processing, integrate VSAM into standard storage management functions, and to allow definition and enforcement of storage allocation requirements using a CLIST-like language for coding data management definitions. The use of explicit device-type allocation, such as UNIT=3380, has also been superseded by the use of generic storage groups and attributes.

Storage Management Classes

The DFSMS Storage Management Subsystem (SMS) includes new features that handle new data set allocations to volumes that it controls, and associates new attributes with those data sets. The Interactive Storage Management Facility (ISMF) provides a mechanism for the storage administrator to specify which volumes are to be under SMS control and which data sets are placed on SMS-controlled volumes; that is, which data sets are SMS-controlled. The process SMS uses to determine whether the data set being allocated is SMS-controlled involves classifying the data set in storage management terms. The data set's storage management classification has no correspondence to or effect on security data classification or security resource classes. SMS implements three types of storage management data set classes: data, storage, and management.

Data
Defines attributes about the data set that would normally be associated with the SPACE, DCB, or AMP parameters in standard JCL. Data class definitions take the place of model space DSCBs, but also include information for VSAM data sets as well. Even if a data set is not SMS-controlled, a data class can be associated with it.

Storage
Defines the type of process that is generally performed for the data set (read or write) and states the performance objectives desired from the backing DASD. Assignment of a storage class to a data set defines the data set as SMS-managed. The storage class assignment affects which SMS-controlled volume is selected to hold that data set.

Management
Defines space and availability management attributes of the data set. This includes how often to perform backups and how many backup versions are kept. A management class can only be associated with an SMS-controlled data set.

The class definitions are kept in a control data set. Each class is defined by a unique eight-character name. It is through this name that access to storage and management classes is secured through external security calls. SMS-controlled data sets carry the name of the data, storage, and management classes associated with them in their ICF catalog records.

Storage Groups

The DASD volumes that are SMS-controlled are divided into storage groups similar in concept to DASD esoteric unit names. They are similar in that they define a group of volumes, but they are different in these ways:

- Storage groups have volume maintenance characteristics associated with them.
- All volumes in a storage group must have the same number of tracks per cylinder and number of bytes per track.
- A volume can be in only one storage group.
- Storage group definitions are shared among multiple SMS systems because each system shares the control data sets that define them.

If you are using SMS volume control, you do not need eTrust CA-ACF2 Note 3. SMS provides the same protection and overrides eTrust CA-ACF2 Note 3.

Automatic Class Selection Routines

When a new data set is allocated, the SMS-managed classes are determined by the Automatic Class Selection (ACS) routines coded by the storage administrator in a language similar to TSO CLISTs. There are four types of ACS routines: three to select the three storage management classes and one to select the storage group where the data set is to reside. These routines have access to information about the job, the user, the data set, and the environment, all of which can impact the selection of the appropriate classes.

When SMS is active, it first passes all new allocations through the data class ACS routine to select the data class and then to the storage class ACS routine to acquire a storage class. If no storage class is selected, the data set **is not** SMS-controlled and the allocation continues through normal allocation processing. Otherwise, SMS calls the security system to check the authority of the selected storage class name. The selected storage class is validated through the security system using the eight-byte storage class name.

If a storage class is selected, the data set is controlled by SMS, and the allocation request passes through the management class ACS routine to assign a management class. The SMS-controlled allocation request then passes to the final ACS routine to select its storage group. When users define a data set with JCL, they can code JCL parameters to specify data class, storage class, and management class. Only the ACS routines can assign the storage group. By associating this new information with data sets, data management software can address specific data set requirements. The management of data sets by system software is the basic goal of SMS. To this end, the components of DFP Version 3 provide the following:

- A z/OS component that honors standard subsystem interfaces for communication and the System Authorization Facility (SAF) for all security controls. (This makes SMS information fully accessible by other system software components.)
- A real-time handler of data set classification and new data set allocations on SMS-controlled volumes.
- A DASD device grouping facility.
- A facility to store additional data set information in the ICF catalog entry of SMS-controlled data sets.
- A facility that can help reduce JCL coding requirements.
- A facility to define performance preferences for allocations.
- A source of information for system programs that request it.

In summary, SMS brings new information about DASD data sets that it manages and provides new services to programs and systems that are prepared to use that information.

Understanding the DFSMS Resource Classes

SMS resource protection is provided through two existing and two new SAF resource classes. The classes that existed before DFSMS are FACILITY and PROGRAM. The new classes are STORCLAS and MGMTCLAS.

FACILITY
Controls the use of catalog, IDCAMS, and DFDSS functions against SMS-managed volumes. These new calls supersede the FACILITY resource name of IGG.CATLOCK previously used for controlling catalog access. The new resource names can be found in the Controlling DFSMS Functions and Commands section later in this chapter.

PROGRAM
Controls the use of all programs, including those used by DFSMS. The programs invoked for the various DFSMS ISMF functions can be protected as resources by using the program name as the resource name. The new program names can be found in the Controlling DFSMS Programs section later in this chapter.

STORCLAS
Controls user access to a specific storage class defined by the storage administrator. Each unique storage class is identified by a unique one to eight-byte name.

MGMTCLAS
Controls user access to a management class defined by the storage administrator. Each unique management class is identified by a unique one to eight-byte name.

You can define these four resource classes to eTrust CA-ACF2 using the GSO CLASMAP record. For details on how to change CLASMAP records for these classes and the internal default values for these classes, see Defining the DFSMS Classes to eTrust CA-ACF2 later in this chapter.

Implementing RESOWNER

During data set allocation processing, DFSMS requests the external security system to derive the resource owner or RESOWNER of the specified data set name. DFSMS validates access to the STORCLAS and MGMTCLAS used during allocation based on the RESOWNER.

In eTrust CA-ACF2, data set owners can allocate their data sets. Data sets are owned when their high-level qualifier matches the logonid of a user. Other data sets, such as SYS1 or PROD data sets, are not explicitly owned by a logonid, but are used by groups in the organization. For users of eTrust CA-ACF2 this distinction is important to remember: RESOWNER is used only to validate access to the STORCLAS and MGMTCLAS, not to the data set itself.

To supply DFSMS with a RESOWNER, eTrust CA-ACF2 must determine a single resource owner of the data set. You can use eTrust CA-ACF2 DFP profile infostorage records as one way to define the RESOWNER. DFP profile infostorage records let a storage administrator who does not have the authority to update eTrust CA-ACF2 access rules assign a resource owner to a data set. This method also lets the storage administrator assign a RESOWNER for each data set or group of data sets as desired. For example:

```
SET PROFILE(DATASET) DIVISION(PROFILE)
COMPILE
$KEY(pay)
 online.data DFP(pay001)
 -.— DFP(pay003)

 SET PROFILE(DATASET) DIVISION(DFP)
 INSERT pay001 RESOWNER(paycurr)
 INSERT pay002 RESOWNER(payhist)
 INSERT pay003 RESOWNER(payothr)
```

eTrust CA-ACF2 supports $KEY masking when you make the rule resident:

```
SET CONTROL(GSO)
CHANGE INFODIR TYPES(R-PDSN) ADD
```

You then use the REBUILD command to implement the changes:

```
F ACF2,REBUILD(DSN),CLASS(PROFILE)
```

If eTrust CA-ACF2 does not find a profile infostorage record for the data set, it processes the request as it did in previous eTrust CA-ACF2 versions. If the data set is owned (that is, the data set's high-level qualifier matches the value specified in the prefix field of the requesting user's logonid), eTrust CA-ACF2 returns the logonid of the requesting user to DFSMS as the RESOWNER. For data sets that are not owned by the requesting logonids, eTrust CA-ACF2 searches the rule set matching the high-level qualifier for a value to return to DFSMS as the RESOWNER.

You can assign a RESOWNER by including a $RESOWNER control statement in the corresponding rule set. The $RESOWNER control statement specifies the logonid acting as the RESOWNER of the data set. For example, the following rule set indicates the PRODMGR logonid is the RESOWNER of the data set and that eTrust CA-ACF2 validates PRODMGR's authority to use the STORCLAS and MGMTCLAS resource classes to allocate a PAY data set.

```
$KEY(pay)
$RESOWNER(prodmgr)
 - UID(payroll) R(A) W(A)
```

The RESOWNER is only used to validate access to the STORCLAS and MGMTCLAS selected for the data set allocation. The logonid making the allocate request still needs the authority to actually allocate the data set.

If eTrust CA-ACF2 does not find a $RESOWNER control statement, it checks the Logonid database to see if there is a logonid equal to the high-level index of the data set. If there is a logonid that matches, it is returned as the RESOWNER.

If there is no logonid that matches the high-level index, eTrust CA-ACF2 defaults to using the logonid of the requesting user. To avoid the overhead of these additional checks, we recommend that you use DATASET profile records to specify the RESOWNER.

Specifying Automatic Class Selection Defaults

SMS lets the RESOWNER supply default values for data, storage, and management classes. While this process is completely supported through eTrust CA-ACF2 in the same manner as RACF, IBM does not recommend their use for most applications. The *IBM System-Managed Storage Migration Planning Guide* states, "We do not recommend the use of default classes in the user or group profile because it is highly unlikely that a given SMS class is applicable to all data sets a user creates. We suggest instead that you use ACS to determine the data, storage and management classes, as well as the application identifier attribute."

The ACSDEFAULTS option in the IGDSMS*xx* member of SYS1.PARMLIB determines whether an external security product is used to store the default class values.

- If you do not want the ACS routines to use the default storage management data saved in the eTrust CA-ACF2 CONTROL(SMS) infostorage record, set the ACSDEFAULTS option to NO. This prevents additional calls to eTrust CA-ACF2 during allocation processing.
- If you do want the ACS routines to use the default storage management data saved in the eTrust CA-ACF2 CONTROL(SMS) infostorage record, set the ACSDEFAULTS option to YES. The RESOWNER is determined as discussed earlier. The application identifier and default class information are extracted from a eTrust CA-ACF2 infostorage record pointed to by the RESOWNER logonid.

An eight-byte logonid record field, SMSINFO, points to the CONTROL(SMS) infostorage record that contains the default data class name, management class name, storage class name, and application identifier. This information is passed back to SMS by eTrust CA-ACF2. The values can be used by the ACS routines as part class determination or they can be overwritten by the ACS routines. The SMSINFO field can be specified for any logonid.

Use the following syntax to specify the SMSINFO field in the logonid record:

```
insert prodmgr smsinfo(defprod)
```

prodmgr
The logonid record that is used to point to the CONTROL(SMS) record when an allocation is requested.

defprod
The name of the infostorage record where the SMS default class attributes are stored. In this example, the record name is *defprod*.

The infostorage record that contains the ACS default values is a CONTROL(SMS) record.

Creating the Automatic Class Selection Defaults

If you are going to use default attributes for the application identifier, data class, storage class, and management class, you must define them in a CONTROL(SMS) record.

Record ID	Fields
recid	DESC(*description*) DATAAPPL(*dataappl*) DATACLAS(*dataclas*) MGMTCLAS(*mgmtclas*) STORCLAS(*storclas*)

Field Descriptions

recid
Defines the one to eight-character name of the record that specifies the default values for the application identifier and the default data, storage, and management classes that will be passed to the ACS routines. For these defaults to apply to a RESOWNER, the name of the record ID must match the contents of the SMSINFO field for the RESOWNER's logonid.

DESC
A 32-character description. You might want to specify a description, such as: **Provides defaults for sys progs**.

DATAAPPL
An eight-character value for the application identifier.

DATACLAS
An eight-character value for the data class.

MGMTCLAS
An eight-character value for the management class.

STORCLAS
An eight-character value for the storage class.

Note: SYSID is not a valid operand for CONTROL(SMS) records.

The SHOW FIELDS subcommand of the ACF command displays the SMSINFO field of the logonid record. The SHOW APPLDEF subcommand displays how the site specifies the SMS record.

Example

You can use the following ACF subcommands to create a CONTROL(SMS) record that specifies the defaults used for all production data set allocations:

```
set control(sms)
 CON TROL
insert defprod dataappl(applprod) dataclas(dataprod) storclas(storprod)
 mg mtclas(mgmtprod) desc(defaults for production payroll)
```

Note: When using set control(sms) to create the SMSINFO record, a REFRESH is not required. The records are picked up at the time of validation.

The previously created DEFPROD infostorage record contains the default values tested by the ACS routines. eTrust CA-ACF2 locates the DEFPROD record by means of the record ID specified in the SMSINFO field of the logonid record.

Validating DFSMS SAF Calls

eTrust CA-ACF2 processes DFSMS SAF calls by default. The SAF classes of STORCLAS, MGMTCLAS, and FACILITY are processed by eTrust CA-ACF2 without the need for additional SAFDEF records. For a list of the eTrust CA-ACF2-provided SAFDEF records, see the "Understanding SAF" chapter. This section describes the only required SAFDEF record for DFSMS support.

If you write resource rules to protect DFSMS programs, you must create a SAFDEF record as follows:

```
set control(gso)
 CON TROL
insert safdef.pgm id(progmchk) mode(global)
 ra croute(request=fastauth,reqstor=progmchk,subsys=contents) rep
```

This SAFDEF record instructs eTrust CA-ACF2 to always validate a program load with the PROGRAM resource class. **All** programs are protected, not just SMS programs.

Note: You should also see Securing DFSMS Programs in this chapter for details on writing resource rules to protect programs.

Defining the DFSMS Classes to eTrust CA-ACF2

The following table lists the eTrust CA-ACF2 default type codes for the four DFSMS resource classes:

Type Code	Resource Class
FAC	FACILITY
PGM	PROGRAM
MGM	MGMTCLAS
STR	STORCLAS

To view the list of internal and external SAF call definitions in effect on your system, issue the SHOW SAFDEF subcommand. To display the defined SAF classes, both internal and external, and the associated resource rule type codes, issue the SHOW CLASMAP subcommand.

To use a different resource type code for a resource class, insert a CLASMAP record. For example, to change the type code for resource class PROGRAM from PGM to PRO, insert the following CLASMAP record:

```
SET CONTROL(GSO) 
INSERT CLASMAP.pro RESOURCE(program) RSRCTYPE(pro) ENTITYLN(8) 
```

Controlling DFSMS Functions and Commands

IBM defines the following resource names for each respective DFSMS function/command. Adefinition of each documented function is provided.

Resource Value	Definition
STGADMIN.IGD.ACTIVATE.CONFIGURATION	Provides the ability to activate a configuration.
STGADMIN.IDC.DIAGNOSE.CATALOG	Provides the ability to run IDCAMS DIAGNOSE against catalogs.
STGADMIN.IDC.DIAGNOSE.VVDS	Provides the ability to run IDCAMS DIAGNOSE against VVDSs (VSAM Volume Data Sets) when a comparison against the BCS is being performed. This protects the BCS. It does not protect the ability to run AMS DIAGNOSE against the VVDS.
STGADMIN.IGG.ALTBCS	Provides the ability to alter BCS (VSAM Basic Catalog Structure) entries.
STGADMIN.IGG.DIRCAT	Provides the ability to use a directed catalog.
STGADMIN.IGG.DEFNVSAM.NOBCS	Provides the ability to define non-VSAM entries without BCS entries.
STGADMIN.IGG.DEFNVSAM.NONVR	Provides the ability to define non-VSAM entries without NVR (non-VSAM Volume Record) entries.
STGADMIN.IGG.DELNVR.NOBCSCHK	Provides the ability to delete an NVR entry without checking the BCS entry.
STGADMIN.IGG.DELETE.NOSCRTCH	Provides the ability to delete a BCS entry without deleting the associated VTOC entry and NVR.
STGADMIN.IGG.DELGDG.FORCE	Provides the ability to delete a GDG using the FORCE operand.
STGADMIN.ADR.COPY.BYPASSACS	Allows the DFDSS COPY function to bypass the ACS routines and use the original (pre-DFSMS) constructs.
STGADMIN.ADR.RESTORE.BYPASSACS	Allows the DFDSS RESTORE function to bypass the ACS routines and use the original (pre-DFSMS) constructs.
STGADMIN.ADR.COPY.INCAT	Provides the ability to use the INCAT parameter on a DFDSS COPY function.
STGADMIN.ADR.DUMP.INCAT	Provides the ability to use the INCAT parameter on a DFDSS DUMP function.
STGADMIN.ADR.CONVERTV.INCAT	Provides the ability to use the INCAT parameter on a DFDSS CONVERTV function.
STGADMIN.ADR.CONVERTV	Provides the ability to use the DFDSS CONVERTV function.

Securing SMS Functions and Commands

The FACILITY resource class is identified to eTrust CA-ACF2 by the three-character type code FAC (unless you assigned a different type code in a GSO CLASMAP record). FACILITY authorizations can be shared or protected using a resource rule for each of the functions.

To simplify administration, we suggest that you write a rule to let all users use all functions and commands, as shown in the following sample:

```
set resource(fac)
 RESOURCE
compile * store
 ACF70010 ACF COMPILER ENTERED
$key(************************************) type(fac)
 uid(-) allow
```

Or:

```
$key(********) type(fac)
 - uid(*) allow
```

Note: In the first example, the $KEY is 39 characters long.

To protect specific commands or functions, you should write specific rules to permit users access to specific functions or commands. For example, to restrict the IGD.ACTIVATE.CONFIGURATION function to only operations staff, you might write the following rule:

```
set resource(fac)
 RESOURCE
compile * store
 ACF70010 ACF COMPILER ENTERED
$key(stgadmin.igd.activate.configuration) type(fac)
 uid(oper-) allow
```

All other users are prevented by default.

Or you could write a rule like the following:

```
set resource(fac)
 RESOURCE
compile * store
 ACF70010 ACF COMPILER ENTERED
$key(stgadmin) type(fac)
 igd.- uid(oper-) allow
```

The dash masking character is not supported for $KEY names in resource rules, although asterisks are. You must create a resource directory for type (FAC) records if you intend to use resource name masking. For details on creating resource directories, see the "Maintaining Resource Rules" chapter.

Controlling DFSMS Programs

Access to ISMF functions is controlled by limiting access to the programs that perform those functions. The following are the program names and their DFP 3.0 ISMF functions. These program names are protected by writing resource rules.

Applications

DFSMS ISMF Programs	DFSMS ISMF ISMF Functions
DGTFFLAD	Invokes ACS application.
DGTFSACD	Invokes CDS application.
DGTFSGDR	Invokes SG application.

STORCLAS Applications

DFSMS ISMF Programs	DFSMS ISMF ISMF Functions
DGTFSCAA	STORCLAS Alter Dialog.
DGTFSCDI	STORCLAS Display Dialog.
DGTFSCLD	STORCLAS List Dialog.
DGTFDIS1	STORCLAS List Display Line Operator.
DGTFALS1	STORCLAS List Alter Line Operator.
DGTFCAS1	STORCLAS List Copy Line Operator.

DATACLAS Applications

DFSMS ISMF Programs	DFSMS ISMF ISMF Functions
DGTFDCDA	DATACLAS Define Dialog.
DGTFDCAA	DATACLAS Alter Dialog.
DGTFDCDI	DATACLAS Display Dialog.
DGTFDCLD	DATACLAS List Dialog.
DGTFDID1	DATACLAS List Display Line Operator.
DGTFALD1	DATACLAS List Alter Line Operator.
DGTFCAD1	DATACLAS List Copy Line Operator.

MGMTCLAS Applications

DFSMS ISMF Programs	DFSMS ISMF ISMF Functions
DGTFMCDA	MGMTCLAS Define Dialog.
DGTFMCAA	MGMTCLAS Alter Dialog.
DGTFMCDI	MGMTCLAS Display Dialog.
DGTFMCLD	MGMTCLAS List Dialog.
DGTFDIM1	MGMTCLAS List Display Line Operator.
DGTFALM1	MGMTCLAS List Alter Line Operator.
DGTFCAM1	MGMTCLAS List Copy Line Operator.

Commands

DFSMS ISMF Programs	DFSMS ISMF ISMF Functions
DGTFACAT	Activate command.

Securing DFSMS Programs

PROGRAM authorizations are protected by using a resource rule for each of the programs. They can also be protected by supplying a program name mask and writing rules for all or some selected programs. You must create a resident resource rule with type (R-RPGM) in the GSO INFODIR record. First, write a rule to let all users load all programs, as shown in the following:

```
set resource(pgm)
 RESOURCE
compile * store
 ACF70010 ACF COMPILER ENTERED
$key(********) type(pgm)
 uid(-) allow
```

Note: The entry for the $KEY is eight characters long.

To secure specific programs, you can write rules to protect only those programs. For example, to restrict that program to only operations staff, you might write the following rule:

```
set resource(pgm)
 RESOURCE
compile * store
 ACF70010 ACF COMPILER ENTERED
$key(dgtfflad) type(pgm)
 uid(oper-) allow
```

All other users are prevented by default.

Although you cannot use the dash to mask program names, you can use asterisks. For example, all program names can be masked using the DGTF**** character string. The following rule let operations staff use all ISMF functions while denying access to all other users implicitly.

```
set resource(pgm)
 RESOURCE
compile * store
 ACF70010 ACF COMPILER ENTERED
$key(dgtf****) type(pgm)
 uid(oper-) allow
```

Note: You must also create a SAFDEF record for programs. For details, see ""Validating DFSMS SAF Calls earlier in this chapter.

The dash masking character is not supported for $KEY names in resource rules, although asterisks are. Due to the way eTrust CA-ACF2 validates the program, you must create a resource directory for type PGM records and specify that the resource rules are resident. For example:

```
SET CONTROL(GSO)
CHANGE INFODIR TYPES(R-RPGM)
```

For details on creating resource directories, see the "Maintaining Resource Rules" chapter.

Securing DFSMS Storage and Management Classes

The STORCLAS and MGMTCLAS resource classes let you control a RESOWNER's access to specific storage classes and management classes, respectively. These storage and management classes are each identified by a unique one to eight-byte resource name. This resource name lets you write resource rules to control access.

The following example shows the ACF subcommands used to let the production manager use a site-defined storage class of PRODSTOR and management class of PRODMGMT. eTrust CA-ACF2 checks these rules when DFSMS requests whether the RESOWNER PRODMGR has access to PRODSTOR and PRODMGMT.

To let him use the storage class PRODSTOR, use the following commands:

```
set resource(str)
 RESOURCE
compile * store
 ACF70010 ACF COMPILER ENTERED
$key(prodstor) type(str)
 uid(prodmgr) allow
```

To let him use the PRODMGMT management class, use the following commands:

```
set resource(mgm)
 RESOURCE
compile * store
 ACF70010 ACF COMPILER ENTERED
$key(prodmgmt) type(mgm)
 uid(prodmgr) allow
```

Processing Storage and Management Classes

When a user makes a request to allocate a data set, DFSMS makes a series of SAF calls to eTrust CA-ACF2 to extract the information it needs to allocate the data set.

The first call is a request to extract the name of the RESOWNER of the data set. The next call is an optional request to extract the default data, storage, and management class names and the application identifier name. This call is performed only if the value for ACSDEFAULTS=YES in the IGDSMS*xx* member of SYS1.PARMLIB. The third call is a request to validate the RESOWNER's authority to use the STORCLAS and MGMTCLAS resource classes.

How eTrust CA-ACF2 Obtains the RESOWNER

When SMS requests eTrust CA-ACF2 to extract the RESOWNER, eTrust CA-ACF2 does the following:

1. If a matching DFP PROFILE record exists, eTrust CA-ACF2 extracts the RESOWNER value from the selected DFP record. If no matching profile record exists, the process continues as follows.
2. If the requesting user owns the data set, eTrust CA-ACF2 uses that user's logonid as the RESOWNER. To determine that a user owns a data set, eTrust CA-ACF2 checks the prefix field of the user's logonid to see if it matches the high-level qualifier (HLQ) of the data set.
3. If the requesting user does not own the data set, eTrust CA-ACF2 retrieves the data set access rule for the HLQ and checks for the $RESOWNER control statement. If the rule exists and the $RESOWNER control statement is present and nonblank, eTrust CA-ACF2 uses that logonid as the RESOWNER.
4. If no rule set exists or the $RESOWNER statement is blank, eTrust CA-ACF2 uses the HLQ of the data set as the RESOWNER.
5. If the RESOWNER was identified to SMS using Steps 3 or 4, eTrust CA-ACF2 verifies that the logonid exists. If the logonid does not exist, eTrust CA-ACF2 uses the logonid of the requesting user as the RESOWNER. If a logonid representing the RESOWNER does not exist, the allocation fails.
6. After eTrust CA-ACF2 determines the RESOWNER, it returns control to SMS.

How eTrust CA-ACF2 Obtains the Default Classes

Next, SMS requests eTrust CA-ACF2 to extract the default classes. eTrust CA-ACF2 does the following to obtain the default values for the application identifier, data class, storage class, and management class:

1. eTrust CA-ACF2 locates the RESOWNER's logonid and uses its SMSINFO field to locate the infostorage record. eTrust CA-ACF2 returns the default information specified in the record to SMS. If the SMSINFO field is blank or if no corresponding infostorage record is found, no default information is returned.
2. After eTrust CA-ACF2 returns the default information, it returns control to SMS. See Specifying Automatic Class Selection Defaults earlier in this chapter for advice about specifying or not specifying defaults at the user level.

How eTrust CA-ACF2 Validates STORCLAS and MGMTCLAS Use

Next, SMS requests eTrust CA-ACF2 to validate the RESOWNER's authority to use the STORCLAS and MGMTCLAS resources selected for this data set allocation. eTrust CA-ACF2 locates the logonid of the RESOWNER and uses it to validate access to the storage class and/or the management class selected by SMS. The validation is based on the resource rules written to protect the STORCLAS and MGMTCLAS resource classes.

Securing Catalogs

Prior to DFSMS, sites used operating system passwords to protect the master catalog from implicit and explicit updates. *Implicit* updates to the master catalog are created through catalog processing when the high level index of the data set name does not have an associated alias directing it to a user catalog. *Explicit* updates to the master catalog are through direct catalog manipulation like IDCAMS ALTER. Since catalog password processing no longer takes place in a system managed storage (SMS) environment, this might or might not have an effect at an installation, depending on the implementation of eTrust CA-ACF2. The most common problem is undesired updates to the master catalog.

Undesired implicit updates are avoided when eTrust CA-ACF2 is active in abort mode. eTrust CA-ACF2 does not permit an update to the catalog if the user is not allowed to create the high level qualifier of the data set name. For example, if a user inadvertently misspells a data set high level qualifier, eTrust CA-ACF2 stops the allocation when no rule is found for that data set. No explicit rule, granting or denying access to the catalog is required. However, explicit catalog manipulation, like IDCAMS ALTER, requires write or allocate access to the catalog through eTrust CA-ACF2 access rules.

Using SAF to Protect Catalogs

You can also use the catalog system authorization facility (SAF) calls to protect catalogs. When activated, these SAF calls validate user access to any master or user catalog regardless of whether the update is implicit or explicit.

You must also write rules giving users read and write access to the user-catalogs in which they are defined because SAF validates implicit updates to the catalog. This is necessary to let users allocate their own data sets. However, this additional authority can allow users catalog update capabilities that they would not have under the non-SAF scenario. Also, users that are allowed to allocate data sets that have their high level index defined in the master catalog need write access to the master catalog. SAF validation also requires that users allowed to directly manipulate a catalog using IDCAMS are allowed eTrust CA-ACF2 allocate access. LISTCAT processing requires READ access to the catalog.

To activate SAF catalog protection, override default SAFDEF CATAUTH as follows:

Supplied SAFDEF:

```
CATAUTH JOBNAME=******** USERID=******** PROGRAM=******** RB=SVC026
        RETCODE=4        SAFDEF=INTERNAL MODE=IGNORE      SUBSYS=ACF
        FUNCRET=4        FUNCRSN=0

        RACROUTE REQUEST=AUTH,CLASS='DATASET'
```

Example SAFDEF override:

```
INSERT SAFDEF.MASTCAT RB(SVC026),MODE(GLOBAL),
RACROUTE(REQUEST=AUTH,CLASS=DATASET,ENTITY=MSTRCAT.-) REP
```

Note: REP is required.

You should write rules before activating the SAFDEF. To protect master catalogs only, write a catchall allow rule for user catalogs or add ENTITY= to the RACROUTE parm of the override SAFDEF to make it apply only to your master catalogs. This assumes you have good maskable naming conventions. If the master catalogs do not have straight forward naming conventions, it is easier to leave the ENTITY parameter off the SAFDEF and write rules to distinguish between the catalogs. Masking in the SAFDEF entity parameter is limited. The dash (-) can be used at the end of the mask only, and asterisks (*) are treated as place holders. For example, m**5****, can match only an eight-character name. If ENTITY is omitted from the SAFDEF, then rules are needed for both master catalogs and user catalogs.

Note: CA-ACF2 VSAM processing for catalogs does not use the VOLUME passed in the SVC026 SAF call if DSTYPE=V. CA-ACF2 does not use the volser passed because it is always the volser of the catalog during DSTYPE=V processing, there is no distinction between a catalog open, or a dataset open. For a dataset open, the volser would be incorrect. Therefore, for DSTYPE=V processing, CA-ACF2 always blanks out the volser for validation.

Sample Case

The following case is intended to serve as an example of how eTrust CA-ACF2 DFSMS support can be implemented at your site. Conditions at your site might differ from those presented in this example.

Company ABC knows it needs to perform the following steps to implement eTrust CA-ACF2 SMS support for the production data sets at their site:

1. Analyze rule sets.
2. Create new logonids to be used as $RESOWNERs.
3. Create CONTROL(SMS) records (only if you ignore the information in Specifying Automatic Class Selection Defaults earlier in this chapter).
4. Compile resource rules to protect the classes.
5. Create DFP PROFILE records, or add $RESOWNER control statements to rule sets. (This step should be performed last and only if needed.)

The first step the security administrator might take is to analyze all production data sets. Company ABC has two types of production data sets: payroll and accounts payable. Their high-level qualifiers begin with the values PAY and ACP.

The rule sets for these data sets are displayed in the following:

```
$KEY(paymon)
 - UID(***paymgr) A(L) W(A) R(A)

$KEY(payweek)
  - UID(***paymgr) A (L) W(A) R(A)

$KEY(acpmon)
 - UID(***prodacp) A(L) W(A) R(A)

$KEY(acpweek)
 - UID(***prodacp) A(L) W(A) R(A)
```

The storage administrator wants to use PRODMGR as the RESOWNER for all these data sets. He uses both methods previously described to accomplish this.

First, he uses a $RESOWNER control statement for his payroll rule sets:

```
$KEY(paymon)
$RESOWNER(prodmgr)
 - UID(***paymgr) A(L) W(A) R(A)

$KEY(payweek)
$RESOWNER(prodmgr)
 - UID(***paymgr) A(L) W(A) R(A)
```

To define the RESOWNER for his accounts payable data sets, he creates DFP profile records:

```
SET PROFILE(DATASET) DIVISION(PROFILE)
COMPILE
$KEY(acp*****)
 - DFP(acpowner)

SET PROFILE(DATASET) DIVISION(DFP)
INSERT acpowner RESOWNER(PRODMGR)
```

Since he is using masking in the $KEY to represent all accounts payable data sets, he must define resident directories and records:

```
SET CONTROL(GSO)
CHANGE INFODIR TYPES(R-PDSN) ADD
```

He must then use the following command to rebuild the directories:

```
F ACF2,REBUILD(DSN),CLASS(P)
```

To enable eTrust CA-ACF2 to return the default classes to SMS if default classes are used, the PRODMGR logonid record must point to an SMS record in the Infostorage database where the default values for the application identifier, data class, storage class, and management class are kept. This SMS record is named DEFPROD.

```
change prodmgr smsinfo(defprod)
```

He also can write the CONTROL(SMS) record to define the default classes. Notice that the record ID, DEFPROD, is the same name he specified in the SMSINFO field of the PRODMGR logonid record.

```
set control(sms)
 CONTROL
insert defprod dataclas(dataprod) storclas(storprod) mgmtclas(mgmtprod)
  dataappl(applprod)
```

Next the security administrator compiles the following resource rules to protect the MGMTCLAS and STORCLAS used to allocate all production data sets:

```
$KEY(storprod) TYPE(STR)
 UID(***prodmgr) ALLOW

$KEY(mgmtprod) TYPE(MGM)
 UID(***prod) ALLOW
```

The names of the STORCLAS and MGMTCLAS to protect are STORPROD and MGMTPROD. The logonid PRODMGR can allocate production data sets using these classes. FACILITY and PROGRAM resource protection are implemented as described earlier. Now that the security administrator has finished implementing eTrust CA-ACF2 SMS support, what happens when PAYMGR tries to allocate the data set PAYMON.JULY.2002? Here is the access rule:

```
$KEY(paymon)
$RESOWNER(prodmgr)
 - UID(***paymgr) A(L) R(A) W(A)
```

Here are the steps SMS and eTrust CA-ACF2 follow to determine if PAYMGR can perform the allocation:

1. SMS extracts the RESOWNER.
   a. eTrust CA-ACF2 cannot find a matching DFP PROFILE record for the data set.
   b. eTrust CA-ACF2 sees that PAYMGR does not own the high-level qualifier of the data set, PAYMON, so eTrust CA-ACF2 reads in the rule set.
   c. eTrust CA-ACF2 searches for the $RESOWNER control statement. It finds the $RESOWNER, reads in the PRODMGR logonid record, and returns PRODMGR to SMS as the RESOWNER.
2. If the site has specified ACSDEFAULTS=YES in the IGDSMS*xx* member of SYS1.PARMLIB, SMS tries to extract the default values for the application identifier, data class, storage class, and management class for use by the ACS routines.
   a. eTrust CA-ACF2 locates the CONTROL(SMS) infostorage record, DEFPROD, specified in the PRODMGR logonid and returns the default values to SMS.
3. Assuming the ACS routines use the defaults passed to them, SMS requests whether the RESOWNER, PRODMGR, is authorized to use STORCLAS's STORPROD and MGMTCLAS's MGMTPROD resources.
   a. eTrust CA-ACF2 validates the resource rules for STORPROD and MGMTPROD and sees that the RESOWNER, PRODMGR, is authorized to use those resource classes. Itreturns that information to SMS.
4. Before the allocation takes place, however, standard eTrust CA-ACF2 validation is performed on the PAYMON data set. eTrust CA-ACF2 checks to see whether PAYMGR can allocate a PAYMON data set. He can, but the rule instructs eTrust CA-ACF2 to log the allocation.

Now SMS can perform the allocation. When PAYMGR allocates the ACPWEEK.WEEK3.DATA data set, eTrust CA-ACF2 finds the RESOWNER in the DFP PROFILE record as in 1.a. and bypasses 1.b. and 1.c. The subsequent steps are the same.

Using the ISPF Panels

To process eTrust CA-ACF2 records for DFSMS, select option 15 SMS from the eTrust CA-ACF2 ISPF Option Selection Menu and press Enter:

```
---------------- eTrust CA-ACF2 ISPF OPTION SELECTION MENU -------------------
OPTION  ===>

   1  RULES     - PROCESS eTrust CA-ACF2 ACCESS AND GENERALIZED RESOURCE RULES
   2  LOGONIDS  - eTrust CA-ACF2 LOGONID CREATION/MAINTENANCE FACILITY
   3  SYSTEM    - eTrust CA-ACF2 SHOW COMMANDS
   4  REPORTS   - eTrust CA-ACF2 REPORT PROGRAM PROCESSOR
   5  UTILITIES - PROCESS eTrust CA-ACF2 UTILITIES
   6  GSO       - GLOBAL SYSTEM OPTIONS SERVICES
   7  NET       - NETWORKING SYSTEM OPTIONS SERVICES
   8  CAC       - MVS DATABASE CACHE RECORD SERVICES
   9  XREF      - SOURCE/RESOURCE AND GROUP RECORD SERVICES
  10  MAC       - MANDATORY ACCESS CONTROL ADMINISTRATION
  11  CPF       - COMMAND PROPAGATION FACILITY SERVICES
  12  FIELD     - RECORD LEVEL PROTECTION CONTROLS
  13  TARGETS   - SET CPF TARGET NODES, DEFAULTS IN USE
  14  PROFILE   - PROCESS PROFILE INFORMATION RECORDS
  15  SMS       - PROCESS DFSMS SUPPORT RECORDS
  16  ENTRY     - PROCESS ENTRY SOURCE RECORDS
  17  SHIFT     - PROCESS SHIFT/ZONE RECORDS
  18  RACDCERT  - PROCESS KEYRING/CERTIFICATE COMMANDS
  19  C-CIC     - PROCESS C-CIC CICS INITIALIZATION RECORDS
  20  LDS       - PROCESS LDAP DIRECTORY SERVICES
```

The eTrust CA-ACF2 DFSMS Support Services panel is displayed. Choose the process you want to initiate:

```
------------- eTrust CA-ACF2 DFSMS SUPPORT SERVICES ------------------------
OPTION  ===>

   1  INSERT   - INITIALLY DEFINE SMS RECORDS

   2  CHANGE   - CHANGE EXISTING SMS RECORDS

   3  LIST     - DISPLAY SMS RECORDS AND PARAMETERS

   4  DELETE   - DELETE AN ENTIRE SMS RECORD

   5  SHOW ALL - ACFFDR AND SMS OPTIONS IN EFFECT

   6  FIELDS   - SHOW SMS RECORD FIELD NAMES

   7  TARGETS  - SET CPF TARGET NODES, DEFAULTS IN USE
```

Creating SMS Records

Select option 1, INSERT from the eTrust CA-ACF2 DFSMS Support Services panel to create an SMS record. The Insert An SMS Record panel is displayed:

```
----------------------------- INSERT A SMS RECORD ----------------------------
COMMAND ===>

  INSERT

  USING RECID   ===>                   OPTIONAL PROTOTYPE RECORD NAME

  RECID         ===>



 SPECIFICATION OF RECID WILL RESULT IN DISPLAY OF APPROPRIATE PANEL
 FOR OPTION SPECIFIED
```

Specify the record ID for the record you are creating and press Enter. The following panel is displayed when you are creating a record:

```
----------------------------- INSERT A SMS RECORD ----------------------------
COMMAND ===>

  MODE=CONTROL  TYPE=SMS

  COMMAND: INSERT

  RECORD-ID: TESTSMS

  FIELDS  DESC     ===>
          DATAAPPL  ===>
          DATACLAS  ===>
          MGMTCLAS  ===>
          STORCLAS  ===>
```

Enter specific field definitions and press Enter.

Changing SMS Records

Select option 2 CHANGE on the eTrust CA-ACF2 DFSMS Support Services panel to modify existing SMS records. The Change An SMS Record panel is displayed:

```
----------------------------- CHANGE A SMS RECORD ----------------------------
COMMAND ===>

  CHANGE

  MASK RECID    ===>                    RECID MASKED VALUE

  RECID         ===> TESTSMS




 SPECIFICATION OF RECID WILL RESULT IN DISPLAY OF APPROPRIATE PANEL
 FOR OPTION SPECIFIED
```

Specify the record ID of the SMS record you want to modify and press Enter. The following detailed panel is displayed:

```
----------------------------- CHANGE A SMS RECORD ----------------------------
COMMAND ===>

  MODE=CONTROL  TYPE=SMS

  COMMAND: CHANGE

  RECORD-ID: TESTSMS

  FIELDS  DESC     ===>
          DATAAPPL  ===>
          DATACLAS  ===>
          MGMTCLAS  ===>
          STORCLAS  ===>
```

Use this panel to modify the fields of the SMS record. When you are finished, press Enter.

Displaying SMS Records

Select option 3 LIST on the eTrust CA-ACF2 DFSMS Support Services panel to display SMS records. The List An SMS Record panel is displayed:

```
----------------------------- LIST A SMS RECORD ------------------------------
COMMAND ===>

  LIST

  MASK RECID    ===>                    RECID MASKED VALUE

  RECID         ===> TESTSMS
```

Specify the record ID of the record you want to see and press Enter. eTrust CA-ACF2 displays the record.

Deleting SMS Records

Select option 4 DELETE on the eTrust CA-ACF2 DFSMS Support Services panel to delete SMS records. The Delete An SMS Record panel is displayed:

```
----------------------------- DELETE A SMS RECORD ---------------------------
COMMAND ===>

  DELETE

  MASK RECID    ===>                    RECID MASKED VALUE

  RECID         ===> TESTSMS
```

Specify the record ID of the SMS record you want to delete and press Enter. eTrust CA-ACF2 deletes the record.

Using the ACF Command

Once within the ACF command, use the SET command to enter profile administration. Enter:

```
SET PROFILE(USER) DIVISION(PROFILE|profile-data-name)
```

Where PROFILE can be one of the following: USER, GROUP, DATASET, SDB2, SECLABEL, KEYSMSTR, DLFCLASS, APPCLU, PTKTDATA, or SYSMVIEW. DIVISION can be PROFILE, which indicates a compiled record, or the profile data name of the data record. Remember that compiled records point to the profile data records.

Define a USER profile record as follows:

```
SET PROFILE(USER) DIVISION(WORKATTR)

INSERT userid WANAME(username)
```

Define a DATASET profile record as follows:

```
SET PROFILE(DATASET) DIVISION(PROFILE)

COMPILE *

$KEY(hlq)
data.set.name DFP(ownrname)

STORE

SET PROFILE(DATASET) DIVISION(DFP)

INSERT ownrname RESOWNER(resowner)
```

Use the following commands for all user profile records:

```
LIST *|recid|LIKE(recidmask)

DELETE *|recid|LIKE(recidmask)
```

Profile data records use the following commands:

```
INSERT *|recid|USING(urecid) recid

CHANGE recid|LIKE(urecid)
```

Compiled records use the COMPILE command. The keys of both the compiled records and the profile data records can be masked when these records are defined in the eTrust CA-ACF2 GSO INFODIR record. The infostorage class for profile records is PROFILE; the letter representing the class is P. eTrust CA-ACF2 distributes the following CLASMAP resource type definitions with the product; adjustments might be needed if your site uses different types for these resources.

Profile record	Resource Type
APPCLU	ALU
DATASET	DSN
DLFCLASS	DLF
GROUP	GRP
KEYSMSTR	KEY
PTKTDATA	PTK
SDB2	SDB
SECLABEL	SEC
SYSMVIEW	SMV
USER	USR

All profiles should be defined as resident in the GSO INFODIR record. To accomplish this, issue the following subcommands:

```
SET Control(GSO)
 CONTROL
CHANGE INFODIR
TYPES(R-PDLF,R-RDLF,R-PGRP,R-PDSN,R-PALU,R-PUSR,R-PPTK,R-RSMV,R-PSEC,R-PSDB)
```

Do not add R-PKEY to this list; its use is optional.

This example assumes you are using the types distributed with eTrust CA-ACF2; if you are using different types, make the appropriate adjustments. These directories do not appear when you issue a SHOW RESIDENT until they have been added to the GSO INFODIR record and that record has been refreshed.

When records are resident in storage, changes are made effective immediately by issuing a console rebuild command. For example:

```
F ACF2,REBUILD(USR),CLASS(P)
```

Using the RECKEY Subcommand

The RECKEY subcommand assists security administrators in maintaining rule sets and compiled infostorage rule records. This subcommand allows the user to decompile, add or delete a rule entry, recompile, and store the updated rule set on one command. This command can be used in any ACF mode that handles compiled records and it executes on other CPF-defined nodes. For more information on the RECKEY subcommand see the "Maintaining Profile Records" chapter.

Chapter

17 Maintaining Field Records

Field records are a class of structured infostorage records that you can use to control access to records or screen fields based on the information in those fields. We refer to this type of control as *record-level protection*. Record-level protection is a specific application of environment controls that is part of resource rules.

The following field records establish part of the record-level protection controls:

EXPRESSN
Specifies a Boolean expression that you want eTrust CA-ACF2 to use to validate access to a record based on the contents of one or more fields.

RECORD
Defines the fields in records or screens that are referenced in EXPRESSN records.

The RECORD definition and EXPRESSN records are used to specify the environment that you want eTrust CA-ACF2 to validate. You specify the environment control using a control statement and a parameter in a resource rule. The $RECNAME control statement specifies the name of a RECORD definition record. You must specify this statement if you want eTrust CA-ACF2 to validate the environment. The RECCHECK parameter is part of the rule entry that specifies an EXPRESSN record that you want eTrust CA-ACF2 to use when it validates access to the record or screen field.

How to Implement Record-Level Protection

eTrust CA-ACF2 provides record-level protection through eTrust CA-ACF2 CICS. You must complete the following steps to implement record-level protection:

1. Specify EXTENDED=YES for the RECORD parameter in the CICS parameter data set
2. Define record layouts using RECORD definition records
3. Define Boolean expression using EXPRESSN records

4. Write resource rules to provide validation
5. Build infostorage directories using the GSO INFODIR record
6. Select appropriate resource rules by specifying eTrust CA-ACF2 CICS CICSKEY or USERKEY definitions.

Also see Implementing Record-level Protection in the "Administration of eTrust CA-ACF2 CICS" chapter in the *CICS Support Guide*.

How Record-Level Protection Works

Record-level protection is a type of environment checking for resource rules. With record-level protection, you are trying to provide eTrust CA-ACF2 with the information it needs to determine the following:

- Can a user or group of users access a screen field based on the information in that field?
- Can a user or group of users access a record based on the information in that record?

You must provide eTrust CA-ACF2 with the information it needs to make those decisions. Here is the list of the basic steps to follow to supply eTrust CA-ACF2 with this information:

1. Define the type of information in the record in a RECORD definition record.
2. Define the comparison you want eTrust CA-ACF2 to use in an EXPRESSN record.
3. Write a resource rule that identifies the name of the RECORD definition record in a $RECNAME control statement and identifies the name of the EXPRESSN record in a rule entry using the RECCHECK parameter.

The rest of this section describes each of these steps. It also includes examples of how violations appear in the ACFRPTRV report.

Define the Type of Information in the Record

To perform its validation, eTrust CA-ACF2 must know the layout of the record the user wants to access. eTrust CA-ACF2 needs to know such information as:

- What is the name of the record?
- What are the fields in the record?
- What is the format of the information in those fields?
- What are the sizes of the fields?

You specify this information in a RECORD definition record.

Suppose you want to protect personnel information that is stored in a file named EMPADDR that is defined in the CICS file control table. The EMPADDR record includes these fields:

```
+0   Record type (length=1)
+1   Name (length=20)
+21  Address line 1 (length=20)
+41  Address line 2 (length=20)
+61  Address line 3 (length=20)
+81  Emergency contact (length=20)
```

You must define this record layout to eTrust CA-ACF2 in a RECORD definition record, as shown in the following:

```
set field(rec)
 FIELD
compile *
 ACF6D100 RECORD COMPILER ENTERED

 . recname(empaddr)
 . fldname(rectype) offset(1) length(1) type(char)
 . fldname(name) offset(2) length(20) type(char)
 . fldname(addr1) offset(22) length(20) type(char)
 . fldname(addr2) offset(42) length(20) type(char)
 . fldname(addr3) offset(62) length(20) type(char)
 . fldname(emercont) offset(82) length(20) type(char)
 .
 ACF6D110 TOTAL RECORD LENGTH= 644 BYTES, 15 PERCENT UTILIZED
 FIELD
store
 ACF6D026 COMPILED RECORD ******** / EMPADDR REPLACED
 FIELD
end
```

Now the record layout is defined to eTrust CA-ACF2.

You can use the ACF subcommand RECKEY to maintain this record.

Define the Comparison You Want eTrust CA-ACF2 to Make

After you define the layout of the record to eTrust CA-ACF2, you must define the following:

- What record or field of the record do you want eTrust CA-ACF2 to validate?
- What type of comparison should eTrust CA-ACF2 make?
- What is the field or record being compared to?

Use the EXPRESSN record to define this information to eTrust CA-ACF2. Here is the basic format for an EXPRESSN record:

```
IF (left-hand side relational-operator right-hand side)
```

IF
You must begin an expression with IF.

left-hand side
Specifies the name of the field in the record or the screen field for which eTrust CA-ACF2 validates access.

relational-operator
Defines a logical operation to be performed between the left- and right-hand sides. The result of the operation is true or false. For example, the left-hand side must equal the right-hand side for the operation to be true.

right-hand side
Specifies a constant value that you want eTrust CA-ACF2 to compare with the left-hand side.

Here is what a very simple expression might look like:

```
IF (NAME = 'JONES')
```

This expression evaluates as true if the value of the NAME field on the screen or in the record equals JONES. If it equals SMITH or any other value, the evaluation is false. Let's return to the EMPADDR example we used for the RECORD definition record we created. Suppose we want to define an operation to check to see if the value of the NAME field equals William Johnson, the company president. Here is how to create the EXPRESSN record:

```
set field(exp)
 FIELD
compile *
 ACF6D101 EXPRESSION COMPILER ENTERED

 . expressn(empname)
 . if (name ne 'william johnson')
 .
 ACF6D110 TOTAL RECORD LENGTH= 250 BYTES, 6 PERCENT UTILIZED
 FIELD
store
 ACF6D026 COMPILED RECORD ******** / EMPNAME REPLACED
 FIELD
end
```

Notice that we specified not equal (NE) in the expression. When we write the resource rule, we want users to have access to all records except William Johnson's. Therefore, we specify NE in the EXPRESSN record.

Now that we have defined the record we want to validate and the operation we want eTrust CA-ACF2 to perform to make its decision, we must put this information in a resource rule.

You can use the ACF subcommand RECKEY to maintain this record.

Write Resource Rule to Perform Validation

Now that we have created RECORD definition and EXPRESSN records to define the layout of the record and the comparison we want eTrust CA-ACF2 to perform, we must create a resource rule that tells eTrust CA-ACF2 what action to take when the comparison evaluates as true.

```
$KEY(EMPADDR) TYPE(CFC)
$RECNAME(EMPADDR)
 UID(PERCLERK) ALLOW RECCHECK(EMPNAME)
 UID(PRESIDNT) ALLOW
```

The $RECNAME control statement identifies the RECORD definition record checked during the validation. The RECCHECK parameter identifies the name of the EXPRESSN record checked during validation.

Here is how eTrust CA-ACF2 validates access to the personnel information record.

- Only users whose UIDs begin with PERCLERK or PRESIDNT can access the EMPADDR record.
- The rule specifies an environment check when the user tries to access the EMPNAME field. Since the EMPNAME EXPRESSN record evaluates as true when NAME does not equal WILLIAM JOHNSON, PERCLERKs can access everyone's records but William Johnson's. When they try to access his record, the operation described in the EXPRESSN record is false so eTrust CA-ACF2 validation processing considers this to not be a matching rule and validation continues with the next rule line.
- The rule does not specify an environment check for UIDs that begin with PRESIDNT. If William Johnson's UID is PRESIDNT, he can access all EMPADDR records.

Skipping Records

Suppose a PERCLERK is asked to print the record for Mary Johnssen, but is not sure how to spell Mary's last name. The PERCLERK does a browse for all employees whose last name begins with JOHN. Since the user performing the job is not William Johnson and the browse includes the WILLIAM JOHNSON record, eTrust CA-ACF2 abends the task based on the RECORD definition record, the EXPRESSN record, and the resource rule.

We want to prevent the PERCLERKs from accessing William Johnson's record, not to prevent them from using applications to improve their efficiency. To allow PERCLERKs to use the browse application, but prevent them from accessing William Johnson's record, use the SKIP field of the RECORD definition record.

The SKIP field tells eTrust CA-ACF2 whether to bypass the record. You can specify one of the following:

SKIP(YES)
eTrust CA-ACF2 performs the browse, but does not let the PERCLERK access the WILLIAM JOHNSON record.

SKIP(NO)
eTrust CA-ACF2 performs the browse and abends if the user performing the browse does not have permission to access a record. NO is the default.

To change the RECORD definition record to allow the PERCLERKs to use the browse application, you must compile the RECORD definition record again as follows:

```
set field(rec)
 FIELD
compile *
 ACF6D100 RECORD COMPILER ENTERED

 . recname(empaddr) skip(yes)
 . fldname(rectype) offset(1) length(1) type(char)
 . fldname(name) offset(2) length(20) type(char)
 . fldname(addr1) offset(22) length(20) type(char)
 . fldname(addr2) offset(42) length(20) type(char)
 . fldname(addr3) offset(62) length(20) type(char)
 . fldname(emercont) offset(82) length(20) type(char)
 .
 ACF6D110 TOTAL RECORD LENGTH= 644 BYTES, 15 PERCENT UTILIZED
 FIELD
store
 ACF6D026 COMPILED RECORD ******** / EMPADDR REPLACED FIELD
end
```

How Violations Appear in the ACFRPTRV Report

Suppose that you want to validate access to the payroll records based on the contents of the SALARY field. For example, payroll clerks can access payroll records if the salary is less than $100,000.

Here is the EXPRESSN record (PAYM) that tests this requirement:

```
EXPRESSN(PAYM)
IF (SALARY GT '100000')
```

Assuming that the RECORD definition record that describes the layout of the SALARY field is named PAYROLL, the following rule validates access to the PAYM transaction based on the contents of the PAYROLL record:

```
$KEY(PAYM) TYPE(CKC)
$RECNAME(PAYROLL)
 UID(CLRK) RECHECK(PAYM) PREVENT
```

Suppose that CLRK01 tries to access the payroll record using PAYM for Richard Wilson, whose salary is $125,000. eTrust CA-ACF2 denies CLRK01 access to the record and creates an SMF violation record.

The violation appears like this in the ACFRPTRV report:

```
eTrust CA-ACF2 UTILITY LIBRARY - ACFRPTRV - GENERALIZED RESOURCE LOG - PAGE    1
DATE 09/03/04 (04.246) TIME 17.41
REQUESTED RESOURCE                                REC  LOOKUP KEY
UID                      SOURCE   CPU  MODULE   DISP     DSP-MOD  KEY-MOD  SERV
    DATE     TIME  JNAME    LID      NAME                 PRE RMC INT PST FIN
R-CKC-PAYM                                       *VIO  R-CKC-PAYM
     CLRK01            TERM89     PRD1          RULE         -        -     READ
04.246 09/03 17.36 PAY1     CLRK01   JONES                  0   0   255 0  16
RLP RECID:  PAYROLL
RLP EXPR:  PAYM
```

Two new fields, RLP RECID and RLP EXPR identify the name of the RECORD definition record and EXPRESSN record identified in the rule. In this case, the test in the EXPRESSN record evaluated as true and the rule specified PREVENT so a violation occurred. If a violation occurs because one or more expressions evaluate as false, the false EXPRESSNs are not identified in the ACFRPTRV report entry for the violation. Only if the rule line that contains an explicit deny caused the violation to occur also contains an EXPRESSN that evaluated as true will the RECID and EXPRESSN names appear in the ACFRPTRV report entry.

For more information about the ACFRPTRV report, see the "ACFPRTRV-Resource Event Log" chapter in the *Reports and Utilities Guide*.

How eTrust CA-ACF2 Sorts Rules

eTrust CA-ACF2 uses RECCHECK conditions as environmental checks that help select the rule that is used. RECCHECK conditions do not directly determine access.

eTrust CA-ACF2 sorts resource rules using the hierarchy described in the following:

1. eTrust CA-ACF2 sorts rule entries from most specific to most general.
2. eTrust CA-ACF2 sorts by UID first. Most specific UIDs come before general ones. For example, UID(PAY) sorts **after** UID(PAYCLRK1).
3. For rule entries that are equally specific, eTrust CA-ACF2 next sorts alphabetically.
4. eTrust CA-ACF2 next sorts by environment controls, such as SHIFT, SOURCE, and UNTIL. RECCHECK is one type of environment control. For example, the first rule entry would sort before the second rule entry based on alphabetical sorting:

   ```
   UID(ABCUSER01) SHIFT(NORMAL) RECCHECK(AAA)
   UID(ABCUSER01) SHIFT(NORMAL) RECCHECK(BBB)
   ```

5. eTrust CA-ACF2 selects the first rule entry that matches. An EXPRESSN record specified in a RECCHECK parameter must evaluate as true for eTrust CA-ACF2 to select it. If it evaluates as false, eTrust CA-ACF2 looks for another rule that matches. Returning to an earlier example, if the following expression evaluates as true, eTrust CA-ACF2 stops searching:

   ```
   UID(ABCUSER01) SHIFT(NORMAL) RECCHECK(AAA)
   ```

   If it evaluates as false, then eTrust CA-ACF2 checks the next rule:

   ```
   UID(ABCUSER01) SHIFT(NORMAL) RECCHECK(BBB)
   ```

Boolean Expression Records (EXPRESSN)

An EXPRESSN record specifies a Boolean expression that you want eTrust CA-ACF2 to use to validate access to a record. eTrust CA-ACF2 compares the data in the field with the expression to make its determination. Unlike some structured infostorage records, EXPRESSN records do not have a eTrust CA-ACF2-defined record name. Instead, you use the COMPILE subcommand and specify the expression name.

Record Format

The full syntax for an EXPRESSN record is defined as follows:

```
EXPRESSN(expname) IF (EXPRESSION)
```

Where EXPRESSION is:

```
([NOT] simple-exp [{AND|OR} [NOT] simple-exp ...])
```

Where *simple-exp* is:

```
(left-hand side relational-operator right-hand side)
```

The fields for the EXPRESSN record are described in the following:

EXPRESSN(*expname*)
Specifies the one to 32-character name of the expression. This is also the value that you specify in the RECCHECK parameter in a resource rule for the record. You cannot specify blanks or special characters.

IF
Signals the beginning of the expression. You must specify IF.

[NOT]
Specifies a negative relationship. For example, the following evaluates as true when the value being tested is neither SMITH nor JONES:

```
IF (NOT((NAME EQ 'SMITH') OR (NAME EQ 'JONES')))
```

simple-exp

Specifies a simple expression, which must contain a left-hand side, a relational operator, and a right-hand side. You must specify at least one simple expression in an EXPRESSN record.

Left-hand side

Identifies the left-hand side entry that you want eTrust CA-ACF2 to compare. You can specify one to 44 characters. eTrust CA-ACF2 accepts the following types of left-hand side entries:

- *fieldname*—specifies the name of a field in a record or screen field.
- R*n*C*n*—specifies the row and column where the field begins. Use this method to specify a field when the screen field is not defined in a RECORD definition record.
- *reserved-word* – specifies a special value that you want eTrust CA-ACF2 to compare. Valid values are:
  - **AID**—specifies the attention ID associated with the entry of the screen being processed. The constant value can name other right-hand side words or can name valid character or hexadecimal literals.
  - **PFKEY**—specifies the PF key number.
  - **PAKEY**—specifies the PA key number.

Relational-operator

Specifies the test that you want eTrust CA-ACF2 to perform between the left- and the right-hand sides. The result is true or false. Use these operators to join a left- and right-hand sides similar to the way you instruct the ACFRPTSL report to select records for report output. You can specify the following kinds of operators:

- **GT or >**—greater than. The left-hand side must be greater than the right-hand side.
- **GE or >=**—greater than or equal to. The left-hand side must be greater than or equal to the right-hand side.
- **LT or <**—less than. The left-hand side must be less than the right-hand side.
- **LE or <=**—less than or equal to. The left-hand side must be less than or equal to the right-hand side.
- **EQ or =**—equal to. The left-hand side must equal the right-hand side.
- **NE or ¬=**—not equal to. The left-hand side cannot equal the right-hand side.

Right-hand side
Specifies the value that you want eTrust CA-ACF2 to compare against the contents of the left-hand side. You can specify 1 to 24 characters. eTrust CA-ACF2 accepts the following types of right-hand side entries:

- *constant*—specifies the value that you want eTrust CA-ACF2 to compare against the contents of the field you specify in fieldname. You must specify single quotes around the constant, such as:

  ```
  IF (SALARY LE '100000')
  ```

  A constant can be one of the following:

  - *fieldname*—specifies that you want to compare a field name from the left-hand side using the logical operator against a field name on the right-hand side. The right-hand side field must be defined in a RECORD definition record using the VALUE field. You cannot specify a field name that references an actual field in a record or field on the screen. That is, you cannot compare one field of a record to another field of the same record.
  - *literal*—specifies that you want to compare a field name from the left-hand side using the logical operator against a literal value on the right-hand side. The right-hand side field must be defined in a RECORD definition record using the VALUE field. The valid forms for literals are:

    'CCCC'—from 1 to 44 characters of data.

    X'xxxx'—from 1 to 44 hexadecimal characters of data. You must specify an even number of hexadecimal characters

    number—a number to be compared against a left-hand side number value.

- *Reserved-word*
  A reserved word to be compared against a left-hand side value. The valid values are:

  - **CLEAR**—specifies the CLEAR key. Specify CLEAR if you specified AID for the left-hand side.
  - **ENTER**—specifies the ENTER key. Specify ENTER if you specified AID for the left-hand side.
  - **PFKEY*n***—specifies the PF key number. Specify if you specified PFKEY for the left-hand side.
  - **PAKEY*n***—specifies the PA key number. Specify if you specified PAKEY for the left-hand side.

- *Number*—specifies a binary number, for example, 110011.

{AND|OR}

Specifies how you want to link two or more simple expressions. For example, when you use AND to link two simple expressions, such as the following, both simple expressions must evaluate as true for the expression to be true:

```
IF ((simple-exp) AND (simple-exp))
```

When you use OR to link two simple expressions, the first simple expression or the second simple expression must evaluate as true for the expression to be true:

```
IF ((simple-exp1) OR (simple-exp2))
```

Usage Notes

Each EXPRESSN record defines a unique test that you want eTrust CA-ACF2 to perform on a field in a record or a field on a screen. To make the expressions available to eTrust CA-ACF2 when the user attempts to enter a value into the field, you must specify the name of the EXPRESSN record in the RECCHECK parameter of the resource rule that applies to that particular file.

eTrust CA-ACF2 lists EXPRESSN records in a format that might be different from the way the user entered them. For example, eTrust CA-ACF2 always uses character names for logical operators such as, EQ for =. Also, eTrust CA-ACF2 always encloses expressions in parentheses. Although the format might appear different, the functionality remains the same.

Sample Resource Rule

Here is a sample resource rule that requests eTrust CA-ACF2 to validate whether a user can access the PAYROLL record based on the contents of the CLKS field:

```
$KEY(INFO) TYPE(CFC)
$RECNAME(PAYROLL)
 UID(PAYMGR) ALLOW
 UID(PAYCLK1) ALLOW RECCHECK(CLKS)
```

When PAYCLK1 accesses the INFO record, eTrust CA-ACF2 performs a test using the CLKS EXPRESSN record. The CLKS record might look like the following example:

```
IF (R2C2 = 'PAYROLL')
```

When combined with the resource rule, this expression lets PAYCLK1 access the INFO record if the screen field that begins at row 2, column 2 is PAYROLL.

Defining Complex Boolean Expressions

You can also use AND, OR, IF, and NOT logic. AND, OR, and NOT let you group expressions. For example:

```
IF ((FIELDA EQ 'ABCD') AND (FIELDB EQ 'EFGH'))
```

This expression is true when both conditions are met. Using OR lets an expression evaluate as true if either condition is met:

```
IF ((FIELDA EQ 'ABCD') OR (FIELDB EQ 'EFGH'))
```

This expression is true when either condition is met. Using NOT lets you include all but one value to match a true condition:

```
IF (NOT(FIELDA = 'ABCD'))
```

This condition is true when FIELDA is any four characters except ABCD.

Using Parentheses in EXPRESSN Records

Use parentheses in complex expressions to indicate how you want eTrust CA-ACF2 to evaluate the expression. Parentheses group the left- and right-hand sides of an expression and they express order, that is, which part of a complex expression is the left-hand side and which side is the right-hand side.

Consider the following example:

```
IF CODE = '3A' OR CODE = '3B' AND SALARY LT '25000'
```

How can you tell what is the left-hand side and what is the right-hand side when there are three parts? The following example shows one way to define the test you want to evaluate using parentheses.

If you want eTrust CA-ACF2 to find a true condition when CODE equals 3A, or CODE equals 3B and SALARY is less than $25,000, then you must use parentheses as shown in the following sample:

```
IF ((CODE = '3A') OR ((CODE = '3B') AND (SALARY LT '25000')))
```

The previous expression evaluates as true when CODE equals 3A, or CODE equals 3B and SALARY is less than $25,000.

```
IF (((CODE = '3A') OR (CODE = '3B')) AND (SALARY LT '25000'))
```

The previous expression evaluates as true when CODE equals 3A and SALARY is less than $25,000 or when CODE equals 3B and salary is less than $25,000. The parentheses clearly define the left- and right-hand sides.

You must specify parentheses when you use NOT in an expression. For examples that use NOT, see Record-level Protection Examples later in this chapter.

Record Definition Records (RECORD)

The RECORD definition record defines the format of the record that you want to protect. You can specify approximately 49 field names in a RECORD definition record. The total size of a RECORD definition record cannot exceed 4K.

Unlike most structured infostorage records, RECORD definition records do not have a defined record name. Instead, you use the COMPILE subcommand and specify the record name.

The record format and field descriptions follow:

```
RECNAME(recid) SKIP(NO|YES)
FLDNAME(fieldname) [ROW(row) COL(col) TYPE(type) [LENGTH(length)]]
                   [OFFSET(offset) TYPE(type) [LENGTH(length)]]
                   [VALUE(value) TYPE(type)]
```

Field Descriptions

RECNAME(*recid*)
Specifies the one to 24-character ID of the record you want to define. This ID must be specified in the $RECNAME control statement in a resource rule. You cannot specify blanks or special characters. This field is required.

SKIP(NO|YES)
Specifies the action that eTrust CA-ACF2 takes during a browse. The default is NO.

- **NO**—specifies that eTrust CA-ACF2 does not bypass a record contained in a browse. eTrust CA-ACF2 abends the browse task and create a logging or violation record.
- **YES**—specifies that eTrust CA-ACF2 bypasses a record contained in a browse that the user is prevented from viewing. eTrust CA-ACF2 does not abend the task and does not create a logging or violation.

FLDNAME(*fieldname*)
Specifies the one to 24-character name of the field. You cannot use blank or special characters. FLDNAME is required. You can specify any number of FLDNAMEs in a RECORD definition record. However, you cannot specify duplicate values for FLDNAME in a RECORD record.

ROW(*row*)
Specifies the vertical row where the screen field begins. The minimum value that you can specify is 1 and the maximum is 43. If you specify a value for ROW, you must also specify a value for COL. You cannot specify ROW with OFFSET or VALUE.

COL(*col*)
Specifies the horizontal column where the screen field begins. The minimum value that you can specify is 1 and the maximum is 133. If you specify a value for COL, you must also specify a value for ROW. You cannot specify COL with OFFSET or VALUE.

TYPE(*type*)
Specifies the type of data contained in the field. The valid types are:

- **BIN**—indicates that the field contains a number stored in binary format. Valid values are from one to four bytes or from 0 to 2,147,483,648.
- **CHAR**—indicates that the field contains character data from 1 to 256 characters in length.
- **HEX**—indicates that the field contains hexadecimal data from 1 to 256 characters in length.
- **PACKED**—indicates that the field contains packed decimal data.
- **ZONED**—indicates that the field contains zoned decimal data.

LENGTH(*length*)
Specifies the length of the data stored in the field.

- Character fields can be compared to character, hexadecimal, binary, packed, or zoned fields. The left-hand and right-hand side lengths do not have to match. If the left-hand side is larger than the right-hand side, eTrust CA-ACF2 pads to the right with blanks.
- Hexadecimal fields can be compared to character, hexadecimal, binary, packed, or zoned fields. The left-hand and right-hand side lengths must match.
- Binary fields can be compared to character, hexadecimal, binary, packed, or zoned fields. The left-hand and right-hand side lengths do not have to match. If the left-hand side is larger than the right-hand side, eTrust CA-ACF2 pads to the left with blanks.

OFFSET(*offset*)
Specifies the starting location in the file record of the field specified in the FLDNAME field. The minimum value that you can specify is one and the maximum is 32,768. You cannot specify this field and the VALUE, COL, or ROW fields. If you specify a value for OFFSET that extends past the end of the record, an error results.

VALUE(*value*)
Specifies a one to 44-character value that you want eTrust CA-ACF2 to compare with to make an access decision. For example, you might specify VALUE for variables such as DEPARTMENT, NAME, and SALARY. You cannot specify VALUE with OFFSET, ROW, or COL.

eTrust CA-ACF2 stores FLDNAME definitions in the Infostorage database. When you list a RECORD record, eTrust CA-ACF2 lists the fields as they were entered. We recommend that you enter the fields as they appear in the record when you create the RECORD definition.

Using the ISPF Panels

Select option 12 FIELD from the eTrust CA-ACF2 ISPF Option Selection Menu.

```
---------------- eTrust CA-ACF2 ISPF OPTION SELECTION MENU ------------------
OPTION  ===>

   1  RULES     - PROCESS eTrust CA-ACF2 ACCESS AND GENERALIZED RESOURCE RULES
   2  LOGONIDS  - eTrust CA-ACF2 LOGONID CREATION/MAINTENANCE FACILITY
   3  SYSTEM    - eTrust CA-ACF2 SHOW COMMANDS
   4  REPORTS   - eTrust CA-ACF2 REPORT PROGRAM PROCESSOR
   5  UTILITIES - PROCESS eTrust CA-ACF2 UTILITIES
   6  GSO       - GLOBAL SYSTEM OPTIONS SERVICES
   7  NET       - NETWORKING SYSTEM OPTIONS SERVICES
   8  CAC       - MVS DATABASE CACHE RECORD SERVICES
   9  XREF      - SOURCE/RESOURCE AND GROUP RECORD SERVICES
  10  MAC       - MANDATORY ACCESS CONTROL ADMINISTRATION
  11  CPF       - COMMAND PROPAGATION FACILITY SERVICES
  12  FIELD     - RECORD LEVEL PROTECTION CONTROLS
  13  TARGETS   - SET CPF TARGET NODES, DEFAULTS IN USE
  14  PROFILE   - PROCESS PROFILE INFORMATION RECORDS
  15  SMS       - PROCESS DFSMS SUPPORT RECORDS
  16  ENTRY     - PROCESS ENTRY SOURCE RECORDS
  17  SHIFT     - PROCESS SHIFT/ZONE RECORDS
  18  RACDCERT  - PROCESS KEYRING/CERTIFICATE COMMANDS
  19  C-CIC     - PROCESS C-CIC CICS INITIALIZATION RECORDS
  20  LDS       - PROCESS LDAP DIRECTORY SERVICES
```

The Record Level Protection Maintenance panel is displayed:

```
------------------ RECORD LEVEL PROTECTION MAINTENANCE ----------------------
OPTION  ===>


   1  FLDOREC - ADD/MAINTAIN FIELD CLASS RECORDS

   2  FLDLREC - DECOMPILE/LIST FIELD CLASS RECORDS
```

Panel Field Descriptions

FLDOREC
Lets you create, display, or delete EXPRESSN or RECORD definition records.

FLDLREC
Lets you list EXPRESSN or RECORD definition records.

Select which of these tasks you want to perform. The sections that follow describe the panels for each of these options.

Creating and Maintaining EXPRESSN and RECORD Records

Select option 1 FLDOREC from the Record Level Protection Maintenance panel to create, display, or delete an EXPRESSN or RECORD definition record. The Record Level Protection Maintenance panel is displayed:

```
---------- RECORD LEVEL PROTECTION MAINTENANCE -------------------------------

COMMAND ===>

RECORD DEFINITION NAME (24 CHARS MAXIMUM)
   NAME    ===>
FIELD RECORD TYPE:
   TYPE    ===>           (REC OR EXP)

DECOMPILE INTO OPTION OR PDS EDIT OF EXISTING RULESET IN A PDS:
   DECOMP RULE PRIOR TO EDIT ===> YES   YES OR NO
ISPF LIBRARY DATA SET (DO NOT INCLUDE MEMBER NAME):
   PROJECT ===>
   LIBRARY ===>
   TYPE    ===>

OTHER PARTITIONED DATA SET (DO NOT INCLUDE MEMBER NAME):
   DATA SET NAME  ===>

PROCESSING OPTIONS:
   PURGE ABOVE RULE SET ===> NO    YES OR NO
   FORCE RULE REPLACE   ===> YES   YES OR NO
```

Panel Field Descriptions

NAME
Enter the name of the EXPRESSN or RECORD definition record that you want to process.

TYPE
Enter REC to process a RECORD definition record or EXP to process an EXPRESSN record.

DECOMP RULE PRIOR TO EDIT
Specify YES to decompile the record before you are placed in edit mode. YES is the default.

ISPF LIBRARY DATA SET (DO NOT INCLUDE MEMBER NAME)
Enter the PROJECT, LIBRARY, and TYPE fields. For example, if the EXPRESSN record is stored in a partitioned data set (PDS) named ACF2.EXPRESSN.PROD(PAYROLL), enter ACF2 for PROJECT, EXPRESSN for LIBRARY, and PROD for TYPE. You do not need to enter the member name. eTrust CA-ACF2 lets you select from a list.

OTHER PARTITIONED DATA SET (DO NOT INCLUDE MEMBER NAME)
Enter the fully-qualified data set name in single quotes, for example 'ADMIN01.ACF2.EXPRESSN'.

PURGE ABOVE RULE SET
Specify YES to delete the EXPRESSN or RECORD definition record. NO is the default.

FORCE RULE REPLACE
Specify YES to replace an existing EXPRESSN or RECORD definition record with the one you create. YES is the default.

Displaying EXPRESSN or RECORD Definition Records

Select option 2 FLDLREC from the Record Level Protection Maintenance panel to decompile a rule at the terminal to place it in a PDS. The Record Level Protection Decompile Processor panel is displayed:

```
---------- RECORD LEVEL PROTECTION DECOMPILE PROCESSOR -----------------------

COMMAND ===>

 eTrust CA-ACF2 COMMAND MODE:
   DISPLAY ===>                VERBOSE/TERSE DISPLAY  (VERBOSE DISPLAY)
   TYPE    ===>                RECORD TYPE (REC, EXP)

 FIELD RECORD NAME:
   $KEY    ===>
   LIKEKEY ===>
   DELETE  ===> NO   YES OR NO (NO)
```

Panel Field Descriptions

DISPLAY
Specify VERBOSE to view the entire record. Specify TERSE to display only the user who stored the record and the date that it was stored.

TYPE
Specify REC to display a RECORD definition record or EXP to display an EXPRESSN record.

$KEY
Specify the one to eight-character name of the EXPRESSN or RECORD definition record that you want to view.

LIKEKEY
Specify a mask to view a group of EXPRESSN or RECORD definition records.

DELETE
Specify YES to erase the rules from the database after you view them. NO is the default.

Using the ACF Command

You can process field records after establishing the FIELD setting of the ACF command:

```
acf
 ACF
set field(exp|rec)
 FIELD
```

After you establish the ACF command setting, you can issue any of the following ACF subcommands:

- ACCESS
- CHKCERT
- *COMPILE
- CONNECT
- *DECOMP (or LIST)
- *DELETE
- END
- EXPORT
- GENCERT
- GENREQ
- HELP
- MLSLABEL
- MLWRITE
- REKEY
- RECKEY
- REMOVE
- ROLLOVER
- SET or T
- SHOW
- SN
- *STORE
- SYNCH

Note: The TEST subcommand is not available for the FIELD setting.

The common subcommands ACCESS, CHKCERT, CONNECT, EXPORT, END, GENCERT, GENREQ, HELP, MLSLABEL, MLWRITE, REKEY, REMOVE, ROLLOVER, SET (or T), SHOW, SN, and SYNCH operate under all settings. The following text describes the function, syntax, and parameters of the other subcommands under the FIELD setting. The subcommands marked with an asterisk (*) work in the same manner under the ACF setting. See the "Overview of eTrust CA-ACF2" chapter for a summary of ACF subcommands available.

SET Subcommand

The SET subcommand lets you maintain FIELD records. The syntax for the SET subcommand is described in the following:

```
SEt or T    [Field(EXP|REC)]
```

After you establish the FIELD setting, you can begin processing EXPRESSN or RECORD definition records.

COMPILE Subcommand

The COMPILE subcommand lets you create an EXPRESSN or RECORD definition record. The syntax of this subcommand is as follows:

```
COMpile   [*]
          [dsn]
          [List|NOList]
          [Store|NOStore]
          [Force|NOForce]
          [ALL]
```

eTrust CA-ACF2 provides two ways to compile rule sets:

1. Directly at the terminal.
2. From a partitioned data set (PDS). FORCE is not the default in this instance; FORCE is the default when compiling all members of a PDS.

These methods were discussed in the "Maintaining Access Rules" chapter under Creating Access Rule Sets.

Parameter Descriptions

Use the following parameters with the COMPILE subcommand:

*** (asterisk)**
Indicates that the text that follows is input to the compiler. This process is referred to as an online compile. In an online environment, the system prompts you to enter the EXPRESSN or RECORD definition record text directly from the terminal. In batch, the statements following the COMPILE statement are assumed to be input.

(no parameters)
Indicates that the text that follows is input to the compiler. This is the same as specifying an asterisk.

dsn
Specifies a PDS and member name that contains the EXPRESSN or RECORD definition record text to be compiled. The PDS name follows TSO conventions. Your high-level index is assumed unless you specify the entire PDS name and enclose it in single quotes. For example, 'PAYROLL.EMPLOYEE.DATA(RULE)' would be specified.

If you do not specify a member name, eTrust CA-ACF2 prompts you for one. To compile input from all PDS members, specify the ALL parameter. When you specify ALL, eTrust CA-ACF2 does an automatic store. eTrust CA-ACF2 does not compile multiple rule sets from a single PDS member.

LIST|NOLIST
Causes the input to the compiler to be displayed on your screen or printed on your listing during compilation of record. NOLIST causes no such display or printed list. LIST is the default; however, the LIST parameter of the compiler is ignored when compiling online. In the case of an online compile, NOLIST is always in effect.

STORE|NOSTORE
Causes the record to be stored at compilation time. NOSTORE means that you must issue the STORE subcommand to store the record. STORE is the default if you are using the ALL parameter to compile all members of a PDS. Otherwise, NOSTORE is the default.

FORCE|NOFORCE
Lets the record be stored regardless of whether it currently exists. NOFORCE lets the field record be stored only if it does not already exist. FORCE is the default.

When used as parameters of the COMPILE subcommand, STORE, NOSTORE, FORCE, and NOFORCE apply only to the current compilation of this particular FIELD record. When used as a parameter of the SET subcommand, FORCE or NOFORCE is in effect until it is changed or until you end ACF command processing.

ALL
Compiles and stores the field records from all the members of a specified partitioned data set (PDS). If any members of a PDS do not contain field records, do not specify this parameter.

For example:

```
acf
 ACF
set field(exp)
 FIELD
compile 'payroll.employee.data(rule)'
```

STORE Subcommand

The STORE subcommand lets you store the previously compiled rule set.

The syntax of the STORE subcommand follows:

```
STore
```

This subcommand accepts no parameters.

If SET NOFORCE has been previously issued, the rule set is stored only if it does not already exist. You receive a message if an access rule is not stored. The NOFORCE parameter of the SET subcommand is described in the "Overview of eTrust CA-ACF2" chapter.

The STORE operation can be rejected if the user does not have the SECURITY privilege.

DECOMP Subcommand

The DECOMP subcommand decompiles a field record that has been previously compiled and stored. Use this subcommand to examine, update, or change records. You can decompile a record at the terminal or into a member of a partitioned data set (PDS). You can also use the LIST subcommand to accomplish the same function as DECOMP.

The syntax of the DECOMP subcommand is:

```
DEComp or List   {*|recid|Like(recidmask)}
                 [Into(dsn)]
```

Parameter Descriptions

The DECOMP subcommand takes the following parameters:

*** (asterisk)**
Indicates you want to decompile the last record that you processed since establishing the FIELD setting.

recid
Specifies the name of field record that you want to decompile or list. If the record name ends with a dash (–), enclose the record ID in single quotes.

Like(*recidmask*)
Specifies a group of records that you want to decompile or list.

Into(*dsn*)
Specifies the data set where you want the record to be decompiled. This data set must be a PDS. No other types qualify. When you specify a fully qualified data set name (that is, including the high-level index), you must enclose that name in single quotes.

```
decomp pamast into('payrpk.work.rule')
```

If a data set is not allocated, eTrust CA-ACF2 allocates that data set as a PDS and decompiles the rules into the PDS. If you specify no member name, eTrust CA-ACF2 uses the $KEY of the rule set. However, if the member name is invalid, eTrust CA-ACF2 automatically generates a member name. For more information about the automatic generation of this member name, see the description of the SET MEMBER subcommand in the "Overview of eTrust CA-ACF2" chapter.

DELETE Subcommand

The DELETE subcommand lets you delete field records. Only users with the SECURITY privilege level can delete field records. You can restrict this authority through the use of scopes.

The syntax of the DELETE subcommand is:

```
DELete    {*|recid|Rule(recidmask)}
```

For example, the following subcommand deletes the field record set PAY7777:

```
delete pay7777
 DELETED
```

After you issue the DELETE command, the message DELETED is returned.

Parameter Descriptions

The DELETE subcommand takes the following parameters:

*** (asterisk)**
Indicates that you want to delete the last field record that you referenced.

recid
Specifies a name of the record that you want to delete. If the record ID ends with a dash (–), enclose the record name in single quotes.

Rule(*recidmask*)
Specifies a name of the field records that you want to delete. If the record ID ends with a dash (–), enclose the record name in single quotes.

RECKEY Subcommand

The RECKEY subcommand assists security administrators in maintaining rule sets and compiled infostorage rule records. This subcommand lets the user decompile, add or delete a record entry, recompile, and store the updated record on one command. This command can be used in any ACF mode that handles compile records and it executes on other CPF-defined nodes.

The syntax of the RECKEY subcommand is:

```
RECKEY recordid {ADD(record-data)|DELETE(record-data)
```

The following example shows the sequence of using the RECKEY command in FIELD mode to add (or modify) a RECORD definition record. Use of RECKEY with FIELD(RECORD) records lets you insert new records or add new FLDNAME definitions to existing records. This is similar to use of RECKEY with compiled rule records; if the rule record is new, it will be inserted, while if the rule record already exists, it will be decompiled first, modified to include the new data, and then will replace the existing rule record.

```
?  t f(rec)
?  reckey testpay add(fldname(name) type(char) offset(20) length(20))
ACF6D100 RECORD COMPILER ENTERED

RECNAME(TESTPAY)
FLDNAME(NAME) TYPE(CHAR) OFFSET(20) LENGTH(20)
ACF6D110 TOTAL RECORD LENGTH= 244 BYTES, 5 PERCENT UTILIZED
ACF60207 RULE F REC ********TESTPAY INSERTED
```

Processing works slightly different for EXPRESSN records within FIELD mode. This type of record cannot be modified (for example, you cannot add to or update an IF statement within an existing EXPRESSN record). You can only use RECKEY to insert new EXPRESSN records while under FIELD mode. You cannot use RECKEY to modify an existing EXPRESSN record. The EXPRESSN record consists of two statements: an EXPRESSN(name) statement that identifies the name of the record to be compiled, and a single IF statement that defines the Boolean logic expression to be performed as part of Record Level Protection (RLP) processing. There can be only one IF statement per EXPRESSN record.

The following example shows use of the RECKEY command to insert a new EXPRESSN record:

```
?  t f(exp)
?  reckey payroll add(if name eq 'david smith')
ACF6D101 EXPRESSION COMPILER ENTERED

EXPRESSN(PAYROLL)
IF NAME EQ 'DAVID SMITH'
ACF6D110 TOTAL RECORD LENGTH= 250 BYTES, 6 PERCENT UTILIZED
ACF60207 RULE F REC ********PAYROLL INSERTED
```

Activating EXPRESSN and RECORD Records

To make EXPRESSN and RECORD definition records available for eTrust CA-ACF2 processing, you must build an infostorage directory for the records, perform a refresh, and rebuild the resident directories. Afterwards, if you make changes to existing field setting records, you must rebuild the resident directory to make the changes take effect.

Use the following procedure to perform these tasks:

1. Add the F-REC and F-EXP infostorage record classes to a GSO INFODIR record as follows:

```
acf
 ACF
set control(gso)
 CONTROL
change infodir types(r-frec,r-fexp)
```

2. Reinitialize the eTrust CA-ACF2 address space or issue the following command:

```
F ACF2,REFRESH(INFODIR)
```

3. Rebuild the resident directories for the F-EXP and F-REC type codes by issuing the following commands:

```
F ACF2,REBUILD(REC),CLASS(F)
F ACF2,REBUILD(EXP),CLASS(F)
```

Defining CICSKEYs and USERKEYs

The CICSKEY and USERKEY definitions tell eTrust CA-ACF2 which types of CICS resources you want to protect. Ensure that you have the following CICSKEY definitions for CICS files, programs, and transactions if you plan to use record-level protection:

```
CICSKEY RESOURCE=FILE,OPTION=VALIDATE,TYPE=CFC
CICSKEY RESOURCE=PROGRAM,OPTION=VALIDATE,TYPE=CPC
CICSKEY RESOURCE=TRANS,OPTION=VALIDATE,TYPE=CKC
```

If you plan to use record-level protection for resources with different type codes, be sure to include the CICSKEY or USERKEY definitions in the CICS job that initializes eTrust CA-ACF2.

Record-Level Protection Examples

This section presents examples of different situations where you can use record-level protection. For each of the examples, we describe the situation and the protection to be enforced, the EXPRESSN record you can define, and the resource rule you can compile.

Controls for Terminal Input

You might want to provide record-level protection based on what a terminal operator enters. For example, you can define expressions to test the following:

- Data a terminal operator keys in to execute a transaction
- Which key a terminal operator uses to process a transaction
- Data a terminal operator enters to start a subsequent transaction.

Note: These examples pertain to the CICS environment with eTrust CA-ACF2 CICS active. See the *CICS Support Guide* for more information.

The eTrust CA-ACF2 CICS implementation of this support performs the screen input checks only on the initial transaction and program invokations.

Validating PFKEY Entry

Site A has two CICS regions: production (PRD1) and test (TST1). A single transaction (MANT) performs maintenance on both systems. To process the transaction on PRD1, the terminal operator enters MANT and presses PF5. To process the transaction on TST1, the terminal operator enters MANT and presses PF6.

Suppose the site wants to let application programmers process the MANT transaction on the test system and allow only the operations department to process the transaction on the production system. To do this, the site must create two EXPRESSN records: one that evaluates as true when a terminal operator presses PF5 and one that evaluates as true when a terminal operator presses PF6.

Here is how the EXPRESSN record would look to test for PF5:

```
IF (PFKEY EQ 5)
```

The name for this EXPRESSN record is MAINT1.

To test for the entry of PF6, the site must create the following EXPRESSN record:

```
IF (PFKEY EQ 6)
```

The name for this EXPRESSN record is MAINT2. Assuming the RECORD definition record is called MAINT, the following rule lets the application programmers process the MANT transaction on TST1 and lets the operations staff process the MANT transaction on PRD1:

```
$KEY(MANT) TYPE(CKC)
$RECNAME(MAINT)
 UID(APPLPRG) RECCHECK(MAINT1) ALLOW
 UID(OPER) RECCHECK(MAINT2) ALLOW
```

Validating Data Entry

The first screen personnel clerks see when they have to enter or change employee information asks for the employee's department. Only personnel supervisors can view, enter, or change information for the Executive department, which has a department code of 01. Personnel clerks can view, enter, or change the information of employees in any other department.

The RECORD definition record named DEPT for the screen defines the fields on this screen. There is only one field, DEPTCODE:

```
                    UPDATE AN EMPLOYEE FILE


                     01-Executive
                     02-Research
                     03-Finance
                     04-Accounting
                     05-Personnel
                     06-Operations

Enter Department Code __
```

After the clerk enters the code, the screen invokes TRN1, which is a screen where the clerk updates the employee information.

Here are some of the possible EXPRESSN records the site could create to test the value entered in the field:

```
IF (DEPTCODE NE '01')

IF (DEPTCODE GT '01')

IF (NOT (DEPTCODE EQ '01'))
```

Each of these expressions evaluates as true when a personnel clerk enters a department code other than 01. The name of this EXPRESSN record is DEPT2.

Because the personnel manager can process all codes, no expression is required for the rule entry that applies to him. The following rule lets the personnel clerks process TRN1 if the department code is not 01, and lets personnel supervisors process TRN1.

```
$KEY(TRN1) TYPE(CKC)
$RECNAME(DEPT)
 UID(PERCLRK) RECCHECK(DEPT2) SERVICE(READ,UPDATE,DELETE) ALLOW
 UID(PERSUPR) SERVICE(READ,UPDATE,DELETE) ALLOW
```

Validating Pseudo-Conversational Entry

Suppose clerks in the Payroll department must enter PAYROLL, the name of the payroll transaction, and then make a selection from a menu of the task that they want to perform. For example:

```
                    PAYROLL TRANSACTIONS

     01 Update employee payroll data
     02 Browse employee payroll data
     03 Delete an employee
     04 Print employee payroll data
     05 Back up payroll database
     06 Exit

Enter selection __
```

The site wants to let all payroll clerks process transactions 02, 04, 05, and 06. However, only PAYCLRK05 and PAYMGR can process transactions 01 and 03.

Here are some of the possible tests for the PAYCLRKs:

```
IF ((PAYROLL EQ ‘’) OR (PAYROLL EQ ‘02’) OR (PAYROLL GE ‘04’)
(PAYROLL EQ ‘05’) OR (PAYROLL EQ ‘06’))

IF ((PAYROLL EQ ‘’) OR (NOT ((PAYROLL EQ ‘01’) OR (PAYROLL EQ ‘03’))))

IF ((PAYROLL EQ ‘’) OR ((PAYROLL NE ‘01’) AND (PAYROLL NE ‘03’)))
```

The site must include the PAYROLL EQ “” clause because the clerk must enter PAYROLL on a blank screen before the menu is displayed. After that, the value he enters in the field is tested. The name of this EXPRESSN record is PAYROLL1. No EXPRESSN record is required to test the PAYCLRK05 and PAYMGR. The following rule allows the personnel clerks to process the PAY1 transaction if they select 02, 04, 05, or 06 and allows PAYCLRK05 and PAYMGR to process the PAY1 transaction no matter what code they select:

```
$KEY(PAY1) TYPE(CKC)
$RECNAME(PAYROLL)
 UID(PAYCLRK) RECCHECK(PAYROLL1) ALLOW
 UID(PAYCLRK05) ALLOW
 UID(PAYMGR) ALLOW
```

As you can see, PAYCLRK05 matches two rule entries. eTrust CA-ACF2 selects the entry for PAYCLRK05 because it is more specific than the rule entry for all PAYCLRKs.

Controls for Fields

You might want to provide record-level protection based on the contents of a field.

Validating Field Contents

Suppose a site wants to control access to payroll records based on the contents of the SALARY field of the PAYROLL RECORD definition record. It could define the following PAY1 EXPRESSN record:

```
IF (SALARY LE '10000')
```

The resource rule for the PAYM transaction might allow clerks access to records if the expression tests true and allow managers access to records where the contents of the field is greater than $10,000.

```
$KEY(PAYM) TYPE(CKC)
$RECNAME(PAYROLL)
 UID(CLRK) RECCHECK(PAY1) ALLOW
 UID(MGR) ALLOW
```

Validating Multiple Fields Contents

Suppose that you want to validate access to the payroll records, based on the contents of the SALARY field and the FUNCTION field. For example, payroll clerks can access payroll records for all job functions except vice presidents, if the salary is less than $100,000.

Here are some expressions to test these requirements:

```
IF ((FUNCTION NE 'VP') AND (SALARY LE '100000'))

IF (NOT ((FUNCTION EQ 'VP') AND (SALARY GT '100000')))
```

The rule to validate access to the PAYM transaction based on the contents of the PAYROLL record is displayed in the following:

```
$KEY(PAYM) TYPE(CKC)
$RECNAME(PAYROLL)
 UID(CLRK) RECCHECK(PAYTEST) ALLOW
```

Validating Field Contents and Data Input

Suppose the site wants to design a test based on the contents of two fields and the data the clerks enter. For example, to view the payroll record, the clerk must type the name of the employee in the NAME field and press the ENTER key. Clerks can view the salaries of all employees except the company president, John Smith, and the CEO, Judy Jones. They cannot view records of employees who earn more than $100,000 or are vice presidents.

The following EXPRESSN records illustrate how to test for these conditions:

```
IF ((NAME NE 'JOHN SMITH') OR (NAME NE 'JUDY JONES') OR
(FUNCTION NE 'VP') OR (SALARY LT '100000'))

IF (NOT (((NAME EQ 'JOHN SMITH') OR (NAME EQ 'JUDY JONES')) OR
(FUNCTION EQ 'VP') OR (SALARY GE '100000')))
```

As you can see, you can build a great deal of complexity into the expressions that you test.

The resource rule to perform the validation would look like the following example:

```
$KEY(PAYM) TYPE(CKC)
$RECNAME(PAYROLL)
 UID(CLRK) RECCHECK(PAYROLL) ALLOW
```

Using Row and Column Position in an EXPRESSN

So far, we have presented expressions that defined tests based on the name of a field. In some cases, RECORD definition records might define fields based on their row and column position. Here is a RECORD definition record that defines fields by their row and column position:

```
RECNAME(PAYMAST)
FLDNAME(NAME) ROW(2) COL(3) TYPE(CHAR) LENGTH(40)
FLDNAME(FUNCTION) ROW(4) COL(3) TYPE(CHAR) LENGTH(8)
FLDNAME(SSN) ROW(4) COL(15) TYPE(PACKED) LENGTH(11)
FLDNAME(SALARY) ROW(6) COL(3) TYPE(PACKED) LENGTH(8)
FLDNAME(HIREDATE) ROW(6) COL(15) TYPE(PACKED) LENGTH(8)
FLDNAME(ADDR1) ROW(8) COL(3) TYPE(CHAR) LENGTH(50)
FLDNAME(ADDR2) ROW(10) COL(3) TYPE(CHAR) LENGTH(50)
```

You can define the same expressions using the row and column definitions as shown in this example:

```
IF ((R2C3 NE 'JOHN SMITH') OR (R2C3 NE 'JUDY JONES') OR
(R4C3 NE 'VP') OR (R6C3 LT '100000'))

IF (NOT (((R2C3 EQ 'JOHN SMITH') OR (R2C3 EQ 'JUDY JONES')) OR
(R4C3 EQ 'VP') OR (R6C3 GE '100000')))
```

These two expressions test true when the NAME field is not John Smith or Judy Jones or the FUNCTION field is not VP or when the SALARY field is less than $100,000.

Chapter

18

Maintaining Identity Records

Extended user authentication support lets eTrust CA-ACF2 interface with a user processing exit that asks users to supply additional proof that they can access the system. eTrust CA-ACF2 acts as the coordinator between the processing exit and the user who requests access. The authentication device can be a software-generated algorithm or an interface to a hardware device, such as an operator identification card reader. See the *Systems Programmer Guide* for more information.

eTrust CA-ACF2 provides identity records so that you can store the authentication information for each user on the Infostorage database. All identity records have the same type, AUTHSUP or AUT (the short form of the type code), even though individual identity records contain data that is unique for each system user. The logonid is used as part of the key to identify each record. The *System Programmer's Guide* demonstrates the use of extended user authentication using Identity records.

You can define a maximum of eight different applications at your site. You should assign a unique authentication type or division to each application. A division is a synonym for SYSID. Using different divisions lets you store different information in the same record in the Infostorage database.

A GSO APPLDEF record lets you specify the formats for the records in each division of identity records. Your application establishes the need for the identity records, but you must define the identity records in the GSO APPLDEF record. If you do not select the INFOSTG option of the AUTHEXIT record, you do not need to create identity records for that application. See the "Maintaining Global System Options Records" chapter for more information about these records. See the *System Programmer's Guide* for additional info on using INFOSTRG Indentity records with EUA.

See the *System Programmer's Guide* for additional information on using INFOSTG Identity records with EUA.

Identity Record Examples

Here is an example of the subcommands you enter to establish the IDENTITY setting and list a record for a user (LB475A) of a specific extended user authentication exit type. The example presupposes the implementation of eTrust CA-ACF2 Note 9. To establish the IDENTITY setting, enter the following subcommands:

```
acf
 ACF
set identity(aut) division(note9)
 IDENTITY
```

You must enter the type code AUT to place the system in the proper setting. When you specify the division, you tell eTrust CA-ACF2 to restrict the next series of subcommands to identity records of the NOTE9 division. To ensure that you have selected the right division, you can enter the following subcommands:

```
set identity(aut) division(note9)
 IDENTITY
show mode
  MODE:  IDENTITY  TYPE: AUTHSUP  DIVISION: NOTE9
  IDENTITY
```

To list the records for a user (LB475A), enter the LIST subcommand. eTrust CA-ACF2 displays the following information:

```
list lb475a
  NOTE9 LB475A  LAST CHANGED BY LB475 ON 6/26/01-11:13
                  EUA-ACNT(4) EUA-DATE(06/26/01) EUA-SRC(LV424)
                  EUA-TIME(11:13) RETN-KEY(10) SRCGROUP(*VLDALL*)
```

Here is a list of what the items in the display mean:

NOTE9
The division name of the record.

LB475A
The name of the record.

LAST CHANGED BY LB475 ON 6/26/01-11:13
The logonid of the user who last updated the record, the date, and the time that it was updated.

EUA-ACNT(4)
The number of accesses for the user. This user has accessed the system four times through EUA.

EUA-DATE(06/26/01)
The last access date. This user last accessed the system using EUA on June 26, 2001.

EUA-SRC(LV424)
The last access source. This user last accessed the system using EUA from the terminal device LV424.

EUA-TIME(11:13)
The last access time. This user last accessed the system using EUA at 11:13.

RETN-KEY(10)
The key that the user must enter to access the system can only be used ten times for EUA processing. After ten times, the key is automatically changed by EUA processing.

SRCGROUP(*VLDVALL*)
The group of entry sources from which this user can access the system for EUA purposes. *VLDVALL* is a source group mask indicating that all sources are acceptable.

The fields you see in this example were created using eTrust CA-ACF2 Note 9 support that is described in Appendix B of the *Systems Programmer Guide*. See the *Systems Programmer Guide* for more information about extended user authentication.

Using the ISPF Panels

No ISPF panels exist for processing identity records.

Using the ACF Command

You can process user authentication interface records using the IDENTITY(AUT) setting. You must create records that contain the data the authentication application needs for verification with the IDENTITY setting.

The ACF command and subcommands follow the format of the CONTROL setting commands. To differentiate between groups of identity records, enter the division with the SET subcommand or one of the subcommands (INSERT, CHANGE, LIST, or DELETE).

DIVISION follows the same format as SYSID. When you use the IDENTITY setting, DIVISION groups the data for a specific authentication application. You define the division in the APPLDIV field on the GSO APPLDEF record for the IDENTITY setting at your site. Use the same conventions for DIVISION in the IDENTITY setting as you do for SYSID in the CONTROL setting.

The examples in this section show possible entries for identity records. The field names differ depending on the authentication applications you use.

You can process AUT records after you establish the IDENTITY setting of the ACF commands and specify the type code AUT:

```
SET IDENTITY(AUT)
```

After you establish the IDENTITY setting, you can issue any of the following subcommands:

- ACCESS
- *INSERT
- CHKCERT
- CONNECT
- *DELETE
- EXPORT
- GENCERT
- GENREQ
- REKEY
- REMOVE
- ROLLOVER
- *SET or T
- *CHANGE
- END or QUIT
- SHOW
- *LIST
- HELP
- SN

The common subcommands ACCESS, CHKCERT, CONNECT, EXPORT, END, GENCERT, GENREQ, REKEY, REMOVE, ROLLOVER, HELP, SHOW, and SN operate under all settings. The other subcommands, marked by asterisks (*), are described in the following text.

SET Subcommand

The SET subcommand lets you establish the IDENTITY(AUT) setting for the ACF command. It also lets you establish the active division plus other parameters that affect how certain ACF subcommands operate.

The SET subcommand under the IDENTITY setting has the following syntax:

```
SEt or T     Identity(Aut)
             [DIVision(appldiv)|SYSid(sysid)]
             [MDIVision(divmask)|MSYSid(sysidmask)]
```

Only the DIVISION and MDIV parameters are unique to the IDENTITY(AUT) setting, and are described in the following. The other parameters of the SET subcommand operate as described in "Overview of eTrust CA-ACF2."

Parameter Descriptions

DIVision(*appldiv*)|SYSid(*sysid*)
Specifies the division of the identity records to be processed under the IDENTITY(AUT) setting. This division remains active for the duration of the IDENTITY(AUT) setting. You can override it with the DIVISION parameter of individual ACF subcommands.

If you do not specify the DIVISION parameter when you establish the IDENTITY(AUT) setting, eTrust CA-ACF2 uses the system default. You define the name of the division when you specify the APPLDIV field of the GSO APPLDEF record. You can mask the APPLDIV field; this allows the use of any division name that matches the mask.

DIV is an acceptable abbreviation for the parameter name DIVISION. SYSID is a synonym for DIVISION.

MDIVision(*divmask*)|MSYSid(*sysidmask*)
Specifies a mask to indicate the divisions of the identity records to be processed under the IDENTITY(AUT) setting. These division masks remain active for the duration of the IDENTITY(AUT) setting. You can override them with the DIVISION parameter of individual ACF subcommands. MSYSID is a synonym for MDIV.

INSERT Subcommand

The INSERT subcommand lets you create an identity record. Syntax for the INSERT subcommand follows:

```
Insert   {*|recid|USING(modelrecid) newrecid}
         [DIVision(?|appldiv)|SYSid(?|sysid)]
         [MDIVision(divmask)|MSYSid(sysidmask)]
         [field1,...,fieldn]
         [ADD|REP|DEL]
```

For the sake of this illustration, assume that the site has an extended authentication routine that requires specific users to enter a second password after normal TSO logon. The site named its division AUTRTN1 to represent its first authentication application. Both the AUTRTN1 division and the routine's record structure block (RSB) are named in the GSO APPLDEF record. The GSO AUTHEXIT record contains the logonid attribute name and associated processing routine for this application. Since the routine uses records for each user, the eTrust CA-ACF2 Infostorage database identity records can be created for each user as follows:

```
acf
 ACF
set identity(aut)  division(autrtn1)
 IDENTITY
insert user1 password(abcdefg)
  USER1  CREATED BY USER2 ON 08/28/01-15:32
```

This example of the INSERT subcommand shows:

- An identity record ID of USER1 has been created for the site-defined AUTRTN1 division.
- The USER1 logonid is required to enter a second password after normal TSO logon through the AUTRTN1 routine. The routine chosen by the site stores the contents of the PASSWORD field entry in encrypted format in the eTrust CA-ACF2 Infostorage database.
- DIVISION can be entered with the SET subcommand or on the same line as the INSERT subcommand of the ACF command.

Parameter Descriptions

The INSERT subcommand supports the following parameters:

*** (*asterisk*)**
Specifies that you want to add the last record you processed since you established the IDENTITY setting. (The asterisk does not work with multiple records or masking.)

recid
Specifies the name of the record to be inserted. Depending on the definitions specified in the APPLDEF record, this name can be from one to forty characters.

USING(*modelrecid*) *newrecid*
Identifies the name of a model identity record to be copied to create the new identity record. eTrust CA-ACF2 inserts all values from the model record into the new record unless suppressed when the field is defined with an @CFDE macro. Any other settings and values you specify in the INSERT subcommand add to or replace the settings and values in the new record.

DIVision(?|*appldiv*)|SYSid(?|*sysid*)
Specifies the division name to which this inserted record applies. The question mark (?) indicates that the division default value for this system should be used. If you do not specify this operand, eTrust CA-ACF2 uses the DIVISION value you specified when you established the IDENTITY setting. If you did not specify this on the SET command, eTrust CA-ACF2 uses the default routine referenced in the APPLDEF record to set the DIVISION name. Acceptable values for the division name are defined on the APPLDIV field in the GSO APPLDEF record. DIV is an acceptable abbreviation for the DIVISION parameter name. SYSID is a synonym for DIVISION.

field1,...,fieldn
Specifies the fields and any values to be added to the new record (or to replace fields copied from a model record). For the specific field names in each record, use the SHOW FIELDS(*recid*) command.

These general rules apply to field names:

- Turn on bit fields by stating the field name. Turn them off by prefixing the field name with NO.
- Activate fields with variable or character values by entering the desired value in parentheses after the field name. For example, TIME(12:30).
- Enter fields that can contain multiple values by enclosing those values in parentheses. For example, PASSWORD(123 132 145). A space or comma is a valid delimiter within the parentheses.

The following parameters apply only to multi-value fields in an identity record.

ADD
Indicates that the specified field values are added to those copied from the model record. The new value is added. Any existing values remain. ADD is the default value.

REP
Indicates that any field value specified is replaced by the value of that field as indicated.

DEL
Indicates that the specified field value is deleted from the record.

CHANGE Subcommand

The CHANGE subcommand, under the IDENTITY(AUT) setting, lets you change an existing identity record.

The CHANGE subcommand has the following syntax:

```
CHAnge    {*|recid|LIKE(recidmask)}
          [DIVision(?|appldiv)|SYSid(?|sysid)]
          [MDIVision(divmask)|MSYSid(sysidmask)]
          [field1,...,fieldn]
          [ADD|REP|DEL]
```

The following CHANGE subcommand alters the user's authentication record by adding more DEVICE information:

```
set identity(aut) division(appldiv1)
 IDENTITY
change user1 device(34567890)
```

ADD is the default of this command if the field is a multi-value field.

Parameter Descriptions

The CHANGE subcommand takes the following parameters:

*** (asterisk)**
Specifies the ID of the record to be changed. An asterisk (*) refers to the last record you processed since you established the IDENTITY setting. (The asterisk does not work with multiple records or masking.)

recid
Specifies the name of the identity record to be changed. Depending on the definitions specified in the APPLDEF record, this name can be from one to forty characters.

LIKE(*recidmask*)
Specifies a mask for the IDs of the identity records to be changed. This masking follows the same conventions that apply to logonids, as described in the "Maintaining Logonid Records" chapter.

DIVision(?|appldiv)|SYSid(?|sysid)
Specifies the division ID of the identity record to be changed. If you specify a question mark (?) as the DIVISION value, eTrust CA-ACF2 uses the established setting for the session. DIV is an acceptable abbreviation for DIVISION.

Any value you specify with the DIVISION parameter can contain asterisks; however, these characters are treated as part of the DIVISION itself and not as an indication of masking.

If you specify the DIVISION parameter, you cannot specify the MDIV parameter. SYSID is a synonym for DIVISION.

MDIVision(*divmask*)|MSYSid(*sysidmask*)
Specifies a mask to indicate the division for which the specified identity record is changed. This must contain at least one asterisk or a trailing dash. The command changes any Identity records having the same record ID and a division values that matches the mask specified in the MDIV operand. MSYSID is a synonym for MDIV.

field1,...,fieldn
Specifies the field value to be changed. For the specific field names in each record, issue the SHOW FIELDS(recid) command.

These general rules apply to field names:

- Turn on bit fields by stating the field name. Turn them off by prefixing the field name with NO.
- Activate fields with variable or character values by entering the desired value in parentheses after the field name. For example, TIME(12:30).
- Enter fields that can contain multiple values by enclosing those values in parentheses. For example, PASSWORD(123 132 145). A space or comma is a valid delimiter within the parentheses.

The following parameters apply only to multi-value fields in an identity record:

ADD
Indicates that any fields and values you specify with this CHANGE subcommand are added to the existing field values in the specified AUT record. ADD is the default value.

REP
Indicates that any fields you specify with this CHANGE subcommand completely replace the corresponding fields in the specified identity record.

DEL
Indicates that any field value you specify with this CHANGE subcommand is deleted from the specified IDENTITY record. If any specified value does not exist in a field, the field remains unchanged.

LIST Subcommand

The LIST subcommand, under the IDENTITY setting, lets you list the contents of an identity record.

The LIST subcommand has the following syntax:

```
List     {*|recid|LIKE(recidmask)}
         [DIVision(?|appldiv)|SYSid(?|sysid)]
         [MDIVision(divmask)|MSYSid(sysidmask)]
```

Parameter Descriptions

The LIST subcommand takes the following parameters:

*** (asterisk)**
Refers to the name of the last identity record you processed since you established the IDENTITY setting. The asterisk is not effective if you specified multiple records by masking.

recid
Specifies the name of the identity record to be listed. Depending on the definitions specified in the APPLDEF record, this name can be from one to forty characters.

LIKE(*recidmask*)
Specifies a mask for the IDs of the identity records to be listed. The LIST command processes any records whose name matches the mask specified. This masking follows the same conventions that apply to logonids, as described in the "Maintaining Logonid Records" chapter. You cannot abbreviate the LIKE parameter.

DIVision(?|*appldiv*)|SYSid(?|*sysid*)
Specifies the division of the identity record to be listed. If you specify a question mark (?) as the DIVISION value, eTrust CA-ACF2 uses the default value for the system. DIV is an acceptable abbreviation for DIVISION. SYSID is a synonym for DIVISION.

Any value you specify with the DIVISION parameter can contain asterisks; however, these characters are treated as part of the DIVISION itself and not as an indication of masking.

If you specify the DIVISION parameter, you cannot specify the MDIV parameter.

MDIVision(*divmask*)|MSYSid(*sysidmask*)
Specifies a mask to indicate the divisions for which the specified identity record is listed. This division mask must contain at least one asterisk or a trailing dash. Identity records listed with this parameter have the same record ID (RECID) as divisions that match the mask specified. MSYSID is a synonym for MDIV.

DELETE Subcommand

The DELETE subcommand, under the IDENTITY setting, lets you delete the specified identity record.

The DELETE subcommand has the following syntax:

```
DELete      {*|recid|LIKE(recidmask)}
            [DIVision(?|appldiv)|SYSid(?|sysid)]
            [MDIVision(divmask)|MSYSid(sysidmask)]
```

Parameter Descriptions

The DELETE subcommand takes the following parameters:

*** (asterisk)**
Specifies the use of the last identity record you processed since you established the IDENTITY setting. The asterisk is not effective if you specified multiple records by masking.

recid
Specifies the name of the identity record to be deleted. Depending on the definitions specified in the APPLDEF record, this name can be from one to forty characters.

LIKE(*recidmask*)
Specifies a mask for the IDs of the identity records to be deleted. The delete command removes any record whose name matches the mask specified. This masking follows the same conventions that apply to logonids, as described in the chapter, "Maintaining Logonid Records." Use extreme care when you delete multiple identity records with this parameter.

When you enter any online command that deletes multiple records from the eTrust CA-ACF2 Infostorage database, eTrust CA-ACF2 prompts you to confirm the DELETE LIKE request. If you respond **Y,** eTrust CA-ACF2 executes the command. If you respond **N** or any other response, eTrust CA-ACF2 ignores the command and prompts you for the next command.

DIVision(?|*appldiv*)|SYSid(?|*sysid*)
Specifies the division of the identity record to be deleted. If you specify a question mark (?) as the DIVISION value, eTrust CA-ACF2 uses the established setting for the session. DIV is an acceptable abbreviation for DIVISION. SYSID is a synonym for DIVISION.

Any value you specify with the DIVISION parameter can contain asterisks; however, these characters are treated as part of the division itself and not as an indication of masking.

If you specify the DIVISION parameter, you cannot specify the MDIV parameter.

MDIVision(*divmask*)|MSYSid(*sysidmask*)
Specifies a mask to indicate the divisions for which the specified identity record are deleted. This mask must contain at least one asterisk or a trailing dash. Use extreme care when deleting multiple AUT records with this parameter. MSYSID is a synonym for MDIV.

Chapter 19

Special eTrust CA-ACF2 Procedures

This chapter describes the following special procedures:

- Implementing TSO full-screen logon support
- Backing up eTrust CA-ACF2
- Using TSO command limiting
- Accessing the system when eTrust CA-ACF2 is inactive

Implementing TSO Full-screen Logon Support

This section describes the following procedures for implementing TSO full-screen logon support:

- Authorizing full-screen logon
- Retaining logon values from session to session
- Accommodating the User Attribute Data Set (UADS)

Authorizing Full-screen Operand Values

To authorize users for the full-screen logon feature, set the TSOFSCRN field in their logonid records. To authorize a user to change information displayed on the logon screen, set the logonid record fields described in the following table:

Parameter/Option	Field in Logonid Record
PROCEDURE	LGN-PROC, PMT-PROC, TSOPROC, VLD-PROC
ACCT NMBR	LGN-ACCT, PMT-ACCT, VLD-ACCT
SIZE	LGN-SIZE, TSOSIZE
PERFORM	LGN-PERF, TSO-PERF
UNIT	LGN-UNIT, TSOUNIT
TIME	LGN-TIME, TSOTIME

Parameter/Option	Field in Logonid Record
NOMAIL	MAIL
NONOTICE	NOTICES
RECOVER	LGN-RCVR, RECOVER

Notes:

- Field names that begin with LGN indicate permission to change a particular operand at logon time. For instance, if you specify the LGN-PROC field of the logonid record, the user has permission to specify a TSO logon procedure name at logon time.
- Field names that begin with PMT determine whether the user is forced to specify a value for the corresponding operand during each logon. For instance, if you turn on the PMT-PROC field, eTrust CA-ACF2 prompts the user for a procedure name at logon time.
- Field names that begin with TSO or other characters might provide default values to be displayed when a value for an option is not saved from the previous session. For instance, the TSOPROC field contains the user's default TSO logon procedure name.
- Only ACCOUNT and PROC can be treated as resources. The VLD prefix indicates resource validation. See the "Maintaining Resource Rules" chapter for more information about resource rules.

For a description of the logonid record fields, see the "Maintaining Logonid Records" chapter. For information about the retention of logon values, see TSO Full-Screen Logon Retention Records in the "Maintaining Global System Options Records" chapter.

Retaining Logon Values between Sessions

A user can retain individual logon values from session to session. You can permit this option by specifying the FSRETAIN field of the GSO TSO record. This retention of logon values is a system-wide option rather than an individual option for each user. For more information about the TSO record, see Displaying TNG Options.

Retention of logon values occurs only if the user has a full-screen attribute. Furthermore, eTrust CA-ACF2 saves only logon values that differ from the default values. This information is stored in the Infostorage database.

Automatic deletion of the logon values retained for a user can occur when the user's logonid record is deleted or all retained values are identical to those already specified in the user's logonid record.

Accommodating UADS

If you decide to use UADS with the eTrust CA-ACF2 Logonid database to maintain TSO users, you **cannot** use the eTrust CA-ACF2 TSO full screen option. See The TSO Logon Interface section in the "User Exits" chapter in the *Systems Programmer Guide* for more information about using UADS with eTrust CA-ACF2.

Backing Up eTrust CA-ACF2

You can dynamically allocate backup work files and backup files. The @DDSN macro specifies a group of dynamic allocation data set names. Data sets are dynamically allocated at startup time and deallocated immediately to ensure allocation can be completed. At backup time, eTrust CA-ACF2 dynamically allocates the files and performs backup processing. You supply the information for the backup files in the GSO BACKUP record. Any preallocated DD statement overrides the GSO BACKUP record or the @DDSN description. The SHOW DDSN subcommand of the ACF command displays the values specified in the @DDSN macro and the data sets in actual use.

You can start eTrust CA-ACF2 with AUTOBACKUP, for dynamic allocation or preallocation, or NOBACKUP, meaning allocation does not take place. At startup time, console messages indicate the status of the allocation:

- PRE—preallocated
- FDR—dynamically allocated
- NOA—not allocated

If you specify NOBACKUP, eTrust CA-ACF2 does not display console messages for backup files.

Work and backup files are dynamically allocated, based on file information in the GSO BACKUP record.

To dynamically allocate files, remove the SYSUT1 DD statement from the eTrust CA-ACF2 startup procedure because it overrides the GSO BACKUP record. For more information about the GSO BACKUP record and work file parameters, see Automatic Backup Options (BACKUP) in the "Maintaining Global System Options Records" chapter.

At backup time, if eTrust CA-ACF2 cannot allocate a work file or a backup file, eTrust CA-ACF2 sends a warning message to the console. The backup stops. eTrust CA-ACF2 processing continues, and the next cluster is processed. If allocation is successful, eTrust CA-ACF2 performs the backup and frees the backup data sets.

The backup only backs up the primary database to the backup files. The GSO BACKUP record lets an additional job or STC start using the STRING command that can then REPRO the backup files into the alternate database. We recommend this full backup procedure.

Using TSO Command Limiting

Through the eTrust CA-ACF2 TSO command limiting function, you can define a list of available TSO commands for an individual user or your entire site. This command limiting applies to TSO commands entered under READY mode or under ISPF.

To activate this feature for an individual, use the TSOCMDS field of the logonid record; to activate it for your entire site, use the CMDLIST field of the GSO record named TSO. If you do not specify a system-wide default and leave the TSOCMDS field blank, TSO operates without eTrust CA-ACF2 command limiting.

Command Limiting Fields

If you use command limiting, you can use the following fields in each user's logonid record:

TSOCMDS(*module-name*)
Specifies the name of a module that contains a list of valid commands for the user. See the ACF$CMDS in CAI.CAIMAC for a sample list.

ALLCMDS
Indicates permission for the user to bypass command limiting. You can bypass command limiting by prefixing the command name with the character specified in the BYPASS field of the GSO TSO record.

CMD-LONG
Requires the user to enter the complete command name or alias name if a command limiting list is active for the user.

Each TSOCMDS field or CMDLIST value specifies a load module in the link pack area or system linklist libraries that is loaded at logon time and stays resident throughout the life of the TSO session. You specify this module in eTrust CA-ACF2 macros and assemble and link-edit it into one of these libraries. The input to the assembler consists of a list of valid command names and aliases.

With the command list, the Terminal Monitor Program (TMP) does not automatically issue an ATTACH for the command name. Invalid command names and implicit CLISTs do not cause a search of directories of all system linklist libraries, saving a log of channel time. IBM recommends that you enter implicit CLISTs with a percent sign (%) preceding the name to avoid this search.

Consider using eTrust CA-ACF2 command lists for users with unlimited privileges. The ALLCMDS attribute can be assigned to these users and, when necessary, they can bypass eTrust CA-ACF2 command limiting by using an optional escape character. You can specify this character in the BYPASS field of the GSO TSO record.

Accessing the System When eTrust CA-ACF2 is Inactive

Normally, if eTrust CA-ACF2 has been stopped or has not been started initially, new jobs cannot enter the system. However, you can access the system without eTrust CA-ACF2 active. For example, if a hardware or software failure destroyed the eTrust CA-ACF2 databases, the eTrust CA-ACF2 recovery jobs that recreate these databases must run without eTrust CA-ACF2 active.

Operator Intervention

When eTrust CA-ACF2 is not active, all jobs that are executing, plus new jobs that enter the system, require operator intervention. Exactly when operator intervention is required depends on the following conditions:

- eTrust CA-ACF2 was not started
- eTrust CA-ACF2 was started, then stopped
- eTrust CA-ACF2 started after a job

eTrust CA-ACF2 Was Not Started

In this case, the operator receives a query at each step initiation. The operator can instruct the system to:

- Let the step continue
- Cancel the step and, therefore, the entire job

New jobs are permitted to enter the system.

eTrust CA-ACF2 Started Then Stopped

If the P ACF2 command was entered to stop eTrust CA-ACF2, the operator receives a query when the job begins execution, at each step initiation, each time a job attempts to access a data set, and whenever a started task attempts to access a data set (if the STC field is specified in the GSO OPTS record). **No new jobs can enter the system**. The operator can take one of the following actions:

- Let the job (or started task) continue.
- Cancel the job.
- Instruct the job to wait for eTrust CA-ACF2 to start. The operator should enter the wait response only if eTrust CA-ACF2 is to be restarted.

eTrust CA-ACF2 Has Not Been Started

Once eTrust CA-ACF2 has been started, the FORCE parameter cannot be used.

eTrust CA-ACF2 Started After a Job Started

After eTrust CA-ACF2 has been started, no further operator intervention is needed. eTrust CA-ACF2 flushes jobs from the system when they enter by way of the reader when eTrust CA-ACF2 is not active.

Whenever a job or task is permitted to access the system without eTrust CA-ACF2 active, automatic data set or resource checking cannot be performed.

Chapter

20 JES Security Overview

This chapter gives the security administrator an overview of JES2 and JES3 security from the perspective of eTrust CA-ACF2. eTrust CA-ACF2 specific facilities and JES exploitable facilities are discussed. This overview consolidates eTrust CA-ACF2 JES-specific information found elsewhere in this guide.

The topics covered include the extensions that eTrust CA-ACF2 brings to JES, NJE processing options, and the additional controls available through use of the System Authorization Facility (SAF).

JCL Extensions

//*LOGONID and //*PASSWORD

eTrust CA-ACF2 provides the ability to specify userid and password information on JCL statements that follow the JOB statement. A //*LOGONID statement is equivalent to the USER= keyword on the JOB statement. A //*PASSWORD statement is equivalent to the PASSWORD= keyword on the JOB statement.

By supporting these statements, interface programs can easily insert security information into a job without parsing and modifying the JOB statement itself.

The USER= and PASSWORD= keywords on the JOB statement override any subsequent //*LOGONID or //*PASSWORD statement. The first occurrence of a //*LOGONID or //*PASSWORD statement overrides any subsequent statements found.

Note: eTrust CA-ACF2 scans for a /*LOGONID JECL card or a //*LOGONID JCL card. Either card can be used to provide the logonid and password for the job.

//*JOBFROM

This statement lets a job run under a userid different than that of the submitter, without the submitter providing a password.

For example, suppose a job is submitted from a CICS region. Without any userid information, the job inherits the CICS region userid, which is not usually desirable. A //*LOGONID statement representing the CICS user could be used, but then the user's password must be present. The //*JOBFROM statement gives you controls with which to allow a userid to be specified without the need for a password.

In addition to a logonid, a source can be specified on the //*JOBFROM statement. This lets a meaningful source be associated with the userid under which the job is running. For example, the statement //*JOBFROM USERA/TERM01 propagates both the CICS user's logonid and terminal source to the job. Without the JOBFROM source, the region's source, usually STCINRDR or INTRDR, is used.

To control use of //*JOBFROM, the logonid of the job or address space from which the jobs are submitted must have the JOBFROM privilege. If the submitting address space does not have this privilege, the statement is ignored. The statement is also ignored if any USER= or //*LOGONID statements are present in the job's JCL.

The following lists the hierarchy from which the job's security information is obtained:

- USER= from the JOB statement.
- //*LOGONID statement.
- //*JOBFROM statement.
- Inherit the submitter's logonid.
- The GSO OPTS batch default.
- The job fails.

Surrogate Processing

Another way to submit jobs that run under a different user than the submitter, without the need to know that user's password, is called surrogate processing. In this case, a user supplies a USER= or //*LOGONID statement to the JCL without any corresponding password. eTrust CA-ACF2 detects that a password should be required. Before raising an error condition, eTrust CA-ACF2 checks to see if the submitter is authorized to run a job under that userid without providing a password.

See Security Classes later in this chapter for more information.

Submission Controls

Inheritance

eTrust CA-ACF2 automatically inherits the logonid and source information of the submitter unless the submitter is a started task. Users need not specify userid and password information on their batch jobs unless they want the job to run under a different security environment than their own. When a job is submitted by a started task, the job must contain logonid information; otherwise, the job runs under the batch default. If no batch default logonid exists, the job submission fails. The batch default is specified in the Global Systems Options (GSO) OPTS record. See eTrust CA-ACF2 Option Specifications (OPTS) in the "Maintaining Global System Options Records" chapter for more information.

RESTRICT

The RESTRICT privilege defines a logonid for which a password is not required. These logonids are used in production batch jobs. These logonids cannot be used to access on-line systems such as TSO or CICS. Normally, a production job scheduler assigns a restricted logonid when submitting a production job. By not requiring passwords and not allowing online access, use of these logonids is restricted to the batch environment. Use of SUBAUTH and PROGRAM can be used to control use of restrict logonids.

SUBAUTH

The SUBAUTH logonid privilege can only be used on a logonid with the RESTRICT attribute and is used to control use of RESTRICT logonids. When SUBAUTH is specified in a RESTRICT logonid, any job that specifies that logonid must be submitted by a program out of an APF-authorized environment. Most job scheduler programs run APF-authorized and would be able to submit jobs that contain RESTRICT and SUBAUTH logonids. TSO users and normal batch jobs would be prevented from submitting jobs with these logonids. We recommend that SUBAUTH be used with PROGRAM to provide complete control over use of restricted logonids.**Note:** SUBAUTH is not checked for SAF processing.

PROGRAM

The PROGRAM keyword in a RESTRICT logonid lets you specify a name or mask that must match the submitting program name for the job to run under that RESTRICT logonid. For example, if your job scheduler program that submits production jobs is named PRODSUB, specify PROGRAM(PRODSUB) on the restricted logonids that the job scheduler assigns to the jobs. The restricted logonid with program cannot be submitted from any other program. We recommend that PROGRAM be used in conjunction with SUBAUTH to provide complete control over use of restricted logonids. **Note:** PROGRAM is not checked for SAF processing.

SOURCE

Source restriction limits a user's access to a specific terminal or group of terminals. During entry validation, eTrust CA-ACF2 processing checks the user's logonid record SOURCE field. If it contains a value, eTrust CA-ACF2 ensures that the user is entering the system from one of the defined source devices. For example, you can restrict a group of users to a group of terminals in a building or department. See the "Maintaining Entry Source and Source Group Records" chapter for more information.

SHIFT

Shift restriction limits a user's entry to the system to a specific time range. You can also restrict access based on the day of the week. During entry validation, eTrust CA-ACF2 processing checks the user's logonid record SHIFT field. If it contains a value, eTrust CA-ACF2 ensures that the user is entering the system during the defined period. Use the logonid LOGSHIFT indicator to let a user access the system outside of the defined SHIFT period while logging the access to SMF. Such accesses are reported using the ACFRPTPW report and can be reviewed by the security administrator. See the "Maintaining Shift and Zone Records" chapter for more information.

JOB

This logonid indicator is used to let the logonid be run as a batch job. If the GSO OPTS record indicates JOBCK, all batch logonids must have JOB specified. If you have the need to prevent certain on-line users from executing batch jobs, you can specify JOBCK in GSO OPTS and specify JOB only in those logonids representing users authorized to execute batch. See Logonid Record Fields in the "Maintaining Logonid Records" chapter for more information.

JCL

This logonid indicator is used to let the logonid user use the TSO SUBMIT, STATUS, OUTPUT, and CANCEL commands. Without this privilege, a TSO user would not be able to submit batch jobs. See Logonid Record Fields in the "Maintaining Logonid Records" chapter for more information.

JESJOBS

Using JESJOBS security validation, you can control who can submit or cancel batch jobs based on node name, jobname or userid. See Security Classes later in this chapter for more information.

NJE Options

JES Network Job Entry (NJE) work has special considerations. NJE work can be batch jobs, sysout, or commands that originate from or travel to other systems. eTrust CA-ACF2 provides you with the ability to define how you want security performed for this type of work. eTrust CA-ACF2 also exploits JES security calls for further protection of your resources.

You define eTrust CA-ACF2 NJE protection options in the GSO NJE record. You can define your options as relevant to a specific NJE node or group of nodes. That is, you can specify different eTrust CA-ACF2 NJE options for different NJE nodes. See Network Job Entry Validation Options (NJE) in the "Maintaining Global System Options Records" chapter for specific information. A brief description of the options follows.

For Incoming NJE Jobs

For incoming NJE jobs, the node name in the GSO NJE record represents the originating node name.

VALIN=YES|ONLY
Specify YES if jobs from the specified node should always be fully validated. Specify ONLY if the jobs should not be validated when eTrust CA-ACF2 performed a full validation on the sending node. Regardless of the option, the logonid specified must exist on the execution node.

Your selection of this option could be based upon the characteristics of the sending and receiving node. For example, a production system might want to specify YES for jobs coming from test systems to ensure full integrity of the incoming work, and ONLY for jobs from other trusted production systems. See VALOUT for an example.

INHERIT|NOINHERIT
Specify INHERIT if jobs from the specified node should be able to inherit the identity of the job's submitter. Specify NOINHERIT if you want to require logonid and password information to be specified in each incoming job from that node. What you specify for this option depends on the level of integrity you require on the receiving node and the trustworthiness of the sending node.

For more control, you can exclude specific logonids from the NJE INHERIT process through use of the NO-INH logonid attribute. NJE jobs that will execute under a logonid that has the NO-INH attribute on the receiving node must provide password information to run. You might want to specify NO-INH on logonids with SECURITY or NON-CNCL privileges to prevent misuse.

Default lid
When a batch job does not provide logonid information, it runs under the batch default as specified in the GSO OPTS record. NJE jobs can also be missing logonid information. Before these jobs are assigned the batch default, eTrust CA-ACF2 checks the GSO NJE record associated with the sending node to see if a NJE DEFAULT lid is specified. If so, the NJE job runs with that logonid. If the GSO NJE record does not specify a default logonid, the batch default is assigned. If a batch default is not specified, the NJE job fails.

This option lets a site run without a general batch default logonid while permitting NJE jobs from specified nodes to run under a default logonid. It also provides accountability as a different default NJE logonid can be specified for each sending node.

Default sysout lid

An NJE job can execute at one node and route its sysout output to another NJE node. Use of the GSO NJE default sysout logonid lets you assign default ownership of undefined sysout.

An example of undefined sysout would be a job for which the logonid exists at the execution node but not at the sysout destination node. Through use of the eTrust CA-ACF2 NJE record, each node can have a different execution default logonid and default sysout logonid.

If a default sysout logonid is not specified, the NJE default logonid is assigned. If the NJE default logonid is not specified, the batch default logonid is assigned. If the batch default logonid is not specified, the owner ID of the sysout is ++++++++.

For Outbound NJE Jobs

For outbound NJE jobs, the node name in the GSO NJE record represents the destination node name.

VALOUT | NOVALOUT

eTrust CA-ACF2 is unique in that it lets a site perform NJE job validation at the sending node. When a NJE job is to execute at a node that is defined in the GSO NJE record as VALOUT, full user validation is performed before the job is sent. On the receiving node, validation is determined by the VALIN option. Different combinations of VALOUT/NOVALOUT and VALIN(YES)/VALIN(ONLY) give differing degrees of security for NJE jobs.

For example, a job with a logonid and password is submitted; the sending node specifies VALOUT, so the logonid and password are validated. An encrypted version of the password is sent to the remote node. If the remote node specifies VALIN(YES), the logonid and password are also validated there. If VALIN(ONLY) is specified at the remote node, the password is not validated because eTrust CA-ACF2 detects that it was validated at the sending node.

ENCRYPT | NOENCRYPT

The encryption option is set depending upon the security product running at the receiving node. If eTrust CA-ACF2 is at the receiving node, specify ENCRYPT; any password specified within the job is then sent in an encrypted format. If the receiving node is not secured by eTrust CA-ACF2, specify NOENCRYPT. The password is then presented to JES in an unencrypted format and is recognized on the receiving node.

WRITER

To prevent certain users from submitting NJE jobs to specific nodes, you can enable the SAF WRITER class validation. See Security Classes later in this chapter for more information.

NJE Group

There are NJE considerations when GROUP validation is used during system entry. The GROUP associated with the logonid during execution is also used when sysout output is validated. For the ownership of the sysout to be propagated, the logonid must have access to that group at the sysout destination node. If group validation fails at the sysout destination node, the NJE sysout default logonid is assigned.

For example, a job is submitted to run on RUNNODE. RUNNODE has implemented group processing and the execution logonid has group ADMIN assigned to it. When the job terminates, the sysout output is returned to the submitting node, SUBNODE. SUBNODE does not use group processing. When the job arrives at SUBNODE, JES validates the logonid with the group ADMIN. eTrust CA-ACF2 group validation sees if a rule exists to allow the logonid to use ADMIN as a group. If the validation fails, the sysout is assigned the NJE sysout default logonid.

See Implementing the Group Logon Parameter in the chapter entitled, "Controlling System Entry," for more information about group processing.

NJE Commands

Operator commands can originate on one node and use NJE facilities to execute on another node. When such a request is received, JES attempts to sign on the originating node name as the execution userid. If the node name cannot be found as a logonid, the NJE default logonid is assigned. Command authorization is performed against the logonid assigned. If you expect NJE operator commands to be used, ensure that logonids representing the possible sending nodes are defined with appropriate operator command authority. See the OPERCMDS section in this chapter for more information.

For RJE Jobs

Workstation ID

JES security attempts to authorize Remote Job Entry (RJE) work using the workstation name as the userid. The RJE access fails with the following message if the userid is not defined to eTrust CA-ACF2:

```
ACF01004 - LOGONID XXXXXXXX NOT FOUND
```

You must define each RJE workstation as a userid or tell eTrust CA-ACF2 to ignore these sign-on validations. If you choose to ignore the validation, insert a SAFDEF record as follows:

```
set c(gso)
 CONTROL
insert safdef.verxrje id(verxrje) mode(ignore)
racroute(request=verifyx,envir=create,session=rjeoper) rep
```

Security Classes

Beyond the standard eTrust CA-ACF2 JES security interface, there are other security calls that you can enable. These security calls are made using the System Authorization Facility (SAF). Through the use of resource rules and the eTrust CA-ACF2 SAF interface, you can secure the following areas of JES processing. For more information, see the IBM *JES Initialization and Tuning Guide*.

Resource rule service attributes can be used. See your JES documentation for descriptions of access levels. SAF access levels are mapped to eTrust CA-ACF2 service levels as follows:

SAF Access Level	eTrust CA-ACF2 Service Level
Read	Read
Update	Update
Control	Delete
Alter	Add

JESJOBS

JESJOBS validation controls both job submit and job cancel activity. The resource name format is:

```
SUBMIT.nodename.jobname.userid
CANCEL.nodename.userid.jobname
```

You can see from the resource names that submit and cancel activity can be controlled in many ways. *Nodename* can control the ability to submit or cancel jobs on a particular system. *Jobname* can control the ability to submit or cancel jobs or a certain name. With *userid* you can control who can submit or cancel jobs.

Users can always cancel jobs they own.

The following example shows a rule to allow only CA7 to submit production jobs on the production node (production jobs always start with the letter P). Anyone can submit production jobs on the test node, but they are logged. The "catch-all" rule lets users submit their own jobs.

```
$KEY(SUBMIT) T(job)
 prodnode.p-.-  UID(ca7lid)  ALLOW
 prodnode.p-.-  UID(*)       PREVENT
 testnode.p-.-  UID(*)       LOG
 -.-            UID(*)       ALLOW
```

To enable JESJOBS validation, perform the following:

- The default resource type for JESJOBS is SAF. If you want to use a different type code, insert a GSO CLASMAP record as follows:

  ```
  INSERT CLASMAP.jjobs RESOURCE(JESJOBS) RSRCTYPE(job)
  ```

  Where *job* is the type code you select.

- Write resource rules for SUBMIT and CANCEL.
- Insert a GSO SAFDEF record to enable the SAF calls:

  ```
  INSERT SAFDEF.jjobs ID(jjobs) RACROUTE(REQUEST=AUTH,CLASS=JESJOBS)
  ```

- Issue the F ACF2,REFRESH(ALL) console command to activate the changes.

JESSPOOL

Validation of the JESSPOOL resource class provides the ability to protect the JES2 and JES3 data sets that are generated when a job runs. JESSPOOL data sets include those created by JES such as joblog, JCL and allocation messages data sets, and user-created data sets such as SYSIN and SYSOUT.

The security administrator can create rules to restrict access of sensitive data to authorized individuals.

The JESSPOOL resource name consists of a node name, userid, job name, job number, data set ID, and data set name. The data set name is specified in the JCL DD statement or defaults to a question mark. For example, if the following is specified in a job:

```
SYSOUT=A,DSN=&&OUTPUT
```

the JESSPOOL name might look like:

```
NODE1.USER1.USER1A.JOB00045.D0000104.OUTPUT
```

When DSN= is not specified, the JESSPOOL name might look like:

```
NODE1.USER1.USER1A.JOB00045.D0000104.?
```

See your IBM *JES Initialization and Tuning Guide* for more information about JESSPOOL data set names.

The eTrust CA-ACF2 administrative steps required to implement JESSPOOL validation are:

- Determine a resource type code to use for JESSPOOL resources
- Specify that type code in a GSO CLASMAP record
- Write resource rules
- Activate validation through GSO SAFDEF

Assuming you select type code **SPL** for JESSPOOL, insert the following GSO CLASMAP record:

```
SET C(GSO)

INSERT CLASMAP.spool RESOURCE(JESSPOOL) RSRCTYPE(SPL)
```

Before you write rules, consider the following:

- No validation takes place when the requesting userid matches the userid in the JESSPOOL resource name. In other words, users can access their own JES data sets; rules are not needed for every user on the system.
- When the SAF validation includes the receiver (RECVR) keyword, access is granted even though a rule does not exist. The receiver keyword specifies a userid that is allowed access when no matching rule is found.

  For example, suppose that a data set is transmitted between systems by use of the TSO XMIT command and that the JESSPOOL resource name contains the userid of the sender. When it is received, the JESSPOOL validation includes RECVR, specifying the userid of the person to whom the data set was sent. During validation, eTrust CA-ACF2 searches for a rule written for the sender. If found, normal validation takes place. If not found, the userid of the receiver is compared with that specified by RECVR. If it matches, access is allowed.

  You can see that without RECVR, rules would have to be written for all remote userids that use XMIT.
- Although logging is performed when a rule specifies LOG or when an access is allowed through NON-CNCL or security privileges, logging is suppressed when a JESSPOOL access violation occurs. You can override this by specifying LOG in the JESSPOOL CLASMAP record. Since a job can contain many JES data sets, and because JESSPOOL validation occurs for each of these data sets, excessive loggings and an inflated violation count would occur without suppression.
- For better performance, add the SPL resource type code to GSO INFODIR so that the rules are made resident in storage. This avoids the overhead of retrieving rules from the database.

To create a JESSPOOL resource rule, issue the following commands:

```
SET R(SPL)

COMPILE * STORE

$KEY(NODE1) TYPE(SPL)
 prodid.prodjob.- UID(prodcntl) ALLOW
 prodid.prodjob.- UID(-) PREVENT
 prodid.- UID(-) LOG
```

This rule lets production control personnel access JES data sets created by the production userid under the production job; all other users are prevented access. Everyone is allowed access to JES data sets created by the production userid when not running under the production job, but access is logged.

Note: Although not explicitly indicated in the previous rule set, all users are allowed full access to their own data sets. Access is not allowed to any data set not under the production ID when the RECVR keyword is used by the SAF caller.

After rules are written, enable JESSPOOL validation by inserting a GSO SAFDEF record as follows:

```
SET C(GSO)

INSERT SAFDEF.spool ID(JESPOOLA) MODE(global)
       RACROUTE(REQUEST=AUTH,CLASS=JESSPOOL) REP
```

To activate this new record, remember to refresh your GSO records by issuing this command:

```
F ACF2,REFRESH(SAFDEF),SYSID(systemid),CLASS(C),TYPE(GSO)
```

For more information about operator commands, see the *Systems Programmer Guide*.

JESSPOOL Performance Hints

Following are some JESSPOOL performance hints:

- JES2 issues many SAF validation calls to secure the environment, the bulk of the calls validating the JESSPOOL resource class. If you do not want to secure JESSPOOL or if you want to limit the types of JESSPOOL validations, you can implement JES2 exit 36 to suppress the SAF calls.

  Documentation for JES2 exit 36 is found in the *IBM JES2 Installation Exits*. A sample exit 36 is found in SYS1.SAMPLIB member HASX36A. Routine The NOCREDEL routine suppresses the JESSPOOL validation calls associated with SYSIN and SYSOUT creation.

- Add an additional check to the sample code to suppress all JESSPOOL validations made during the job startup. This additional check suppresses the validation calls made for creation of a job's JES system data sets, such as job log and JCL. Adding the following two lines just prior to the check for $SEADEL also improves performance in the JES2 address space:

  ```
  CLI    XPLIND,$SEASSOC   Was the call for system data set create?
  BE     BYPASS            Yes, then bypass
  ```

SURROGAT

Surrogate processing provides the ability for a user to submit a job that runs under another person's logonid without the submitter knowing the execution logonid's password. The SURROGAT class also allows an address space (like a CICS region) to use a logonid without knowing the password.

Resource rules control the ability to do this. The resource created by the SAF call is the userid being surrogated, followed by the environment being used. An example for JES would be:

```
$KEY(BETA) TYPE(SUR)
 SUBMIT UID(uid_for_ALPHA) ALLOW
```

In other environments (like CICS), the address space might need access to various logonids it uses without passwords. The resource constructed for this validation could be: userid.DFHSTART or userid.DFHINSTL. Since more than a single logonid can be involved, you could address the access by writing a single masked rule as follows:

```
$KEY(********) TYPE(SUR)
 DFH- UID(uid_for_CICS_region) ALLOW
```

You can also be more specific as follows:

```
$KEY(********) TYPE(SUR)
 DFHSTART UID(uid_for_CICS_region) ALLOW
 DFHINSTL UID(uid_for_CICS_region) ALLOW
```

If the CICS region can also use the BETA logonid under SURROGAT processing, the BETA rule must reflect this. For example:

```
$KEY(BETA) TYPE(SUR)
 SUBMIT UID(uid_for_ALPHA) ALLOW
 DFH- UID(uid_for_CICS_region) ALLOW
```

All resource rule facilities, such as SHIFT, SOURCE and UNTIL, are available for SURROGAT rules.

If a rule allowing access does not exist, the job is canceled with message ACF01007 indicating that a password is required.

NON-CNCL and SECURITY violation overrides are not performed for surrogate processing. That is, a rule must exist to allow a user to submit a job with a surrogate userid.

The SURROGAT class is enabled by default. No surrogate action is allowed until a resource rule is written.

OPERCMDS

System operator commands can be controlled through use of OPERCMDS validation. Users issuing commands can be signed-on consoles, SDSF users, TSO/E Extended MCS consoles, or NJE commands.

The resource name format is:

```
subsys.command.operand
```

Examples of subsys names are MVS, JES2, JES3 and JESA. The resource names used for JES commands are documented in the IBM *JES Initialization and Tuning Guide*. For example, the resource name associated with the JES2 $SI command is:

```
JES2.START.INITIATOR
```

An example of a rule to allow anyone to issue JES2 display commands but limit other commands to systems people would be:

```
$KEY(JES2) T(SAF)
 DISPLAY.-  UID(*)  ALLOW
 -          UID(systems) ALLOW
```

To enable OPERCMDS validation, perform the following:

- The default resource type for OPERCMDS is SAF. If you want to use a different type code, insert a GSO CLASMAP record as follows:

  ```
  INSERT CLASMAP.opercmd RESOURCE(OPERCMDS) RSRCTYPE(opr)
  ```

  Where *opr* is the type code you have selected.

- Write resource rules to protect JES commands. Remember that z/OS commands are also protected by OPERCMDS. If you do not wish to protect z/OS commands, you must create a rule that allows access similar to the following:

  ```
  $KEY(MVS) T(opr)
   -  UID(*)  ALLOW
  ```

- We recommend that OPERCMDS rules be kept storage resident. Add the resource types to the GSO INFODIR record as resident:

  ```
  CHANGE INFODIR TYPES(R-Ropr)
  ```

  The GROUP and USER profile records must be made resident by adding them to the GSO INFODIR record as follows:

  ```
  INSERT INFODIR TYPES(R-PGRP)
  INSERT INFODIR TYPES(R-PUSR)
  ```

  Then issue the REFRESH and REBUILD console commands:

  ```
  F ACF2,REFRESH(INFODIR)
  F ACF2,REBUILD(opr)
  ```

- Insert a GSO SAFDEF record to enable the SAF calls:

  ```
  INSERT SAFDEF.opercmd ID(oper) RACROUTE(REQUEST=AUTH,CLASS=OPERCMDS)
  ```

- Issue the F ACF2,REFRESH(ALL) console command to activate the changes.

WRITER

With WRITER validation, you can control where data is sent. This includes output to a printer, or jobs sent through NJE to another node. Examples of resource names are:

```
subsys.LOCAL.device
subsys.NJE.nodename
```

An example of a rule protecting certain printers and NJE destinations follows:

```
$KEY(JES2) T(SAF)
 LOCAL.printer1  UID(systems)  ALLOW
 LOCAL.-         UID(*)  ALLOW
 NJE.prodnode    UID(produid)  ALLOW
 NJE.testnode    UID(*)  ALLOW
 -               UID(*)  LOG
```

To enable WRITER validation, perform the following:

- The default resource type for WRITER is SAF. If you want to use a different type code, insert a GSO CLASMAP record as follows:

  ```
  INSERT CLASMAP.writer RESOURCE(WRITER) RSRCTYPE(wtr)
  ```

  Where *writer* is the type code you select.

- Write resource rules.
- Insert a GSO SAFDEF record to enable the SAF calls:

  ```
  INSERT SAFDEF.writer ID(writer) RACROUTE(REQUEST=AUTH,CLASS=WRITER)
  ```

- Issue the F ACF2,REFRESH(ALL) console command to activate the changes.

Chapter 21

z/OS Unix System Services Support

This chapter explains how to implement the basic UNIX System Services (OMVS) functions in an eTrust CA-ACF2 environment. Additional information regarding the administrative commands used in this chapter can also be found in the *Administrator Guide*.

Implementing eTrust CA-ACF2 in a z/OS Environment

In environments where users move across multiple hardware platforms and operating systems to access numerous applications, security is a major concern. Sites need the same control over data and resources accessed in an open system as they have in their mainframe environment. eTrust CA-ACF2 offers security for such open environments by supporting z/OS Unix System Services and the standards developed for a Portable Operating System Interface (POSIX). Specifically, eTrust CA-ACF2 supports these services in a z/OS Unix System Services environment:

- Callable Services
- Hierarchical File System (HFS)
- A SAF Router and Interface
- Userid (UID) and Groupid (GID) definitions
- Home and Path definitions
- Audit Records for z/OS Unix System Services
- z/OS Unix System Services Security Trace Facility
- Digital Certificate Support

The following sections explain:

- Controlling access to z/OS Unix System Services
- Creating eTrust CA-ACF2 logonid and profile records for z/OS Unix System Services

For general information about profile records, see the *Administrator Guide*. For operator commands, see the *Systems Programmer Guide*. For information on the reporting facilities available, see the *Reports and Utilities Guide*. For information on how to implement MLS in a UNIX System Services environment using eTrust CA-ACF2, see the *Multilevel Security Planning Guide*.

Starting eTrust CA-ACF2 in a z/OS Unix System Services Environment

When z/OS starts up the address spaces associated with z/OS Unix System Services, there exists a potential for various external security calls. If the external security manager is not active, the results of these calls can lead to various error messages or initialization failure. Because of this, we highly recommend that eTrust CA-ACF2 be started using the CAISEC00 member in SYS1.PARMLIB. For more information see the "Installing eTrust CA-ACF2 " chapter of the *Getting Started*.

Controlling Access to z/OS Unix System Services

When a user attempts to enter the z/OS Unix System Services shell, eTrust CA-ACF2 verifies that the user is a z/OS Unix System Services user before initializing the shell. It also verifies that the user associated with a program attempting to access z/OS Unix System Services resources is a z/OS Unix System Services user before allowing access to the requested resource.

To define a z/OS Unix System Services user, you must:

- Define the user to eTrust CA-ACF2
- Assign a z/OS Unix System Services GID to the group

Defining z/OS Unix System Services Users

z/OS Unix System Services recognizes users by their assigned user identification (UID) and group identification (GID) numbers. (The z/OS Unix System Services UID is not the same as eTrust CA-ACF2 UID.) UIDs and GIDs can have numeric values of zero to 2,147,483,647. A GID and UID should be unique to the user or group that it represents. Generally, you should reserve the lower number range for system use. The UID is defined in a profile record. This profile record defines a user's UID, the user's home directory, and the initial program that the user will run. The initial program is generally the shell program that the user invokes.

For more information about creating profile records, see Creating eTrust CA-ACF2 Logonid Records for z/OS Unix System Services later in this chapter.

Superusers

A *superuser* is a special user under z/OS Unix System Services. The superuser is a trusted user who can maintain the z/OS Unix System Services system and administer security in the Hierarchical File System (HFS). A superuser's UID has a value of zero. Use caution when assigning users the superuser authority. A superuser passes all security checks and can access any file in the file system. This type of authority is similar to that of an unscoped security officer, although it does not give the user any added authority outside of z/OS Unix System Services.

z/OS Unix System Services provides controls such that users need not be assigned a UID of zero. They can still retain the ability to switch to superuser status when this is required. See Controlling Access to Superuser Status later in this chapter for more information.

Defining z/OS Unix System Services Groups

z/OS Unix System Services security is based on user and group ownership of files and processes. eTrust CA-ACF2 uses the GROUP field of the logonid record to assign the user to a z/OS Unix System Services group. Users can also specify the group they want to be associated with at system entry time. A group profile record assigns the GID to the group. For more information about group profile records, see Creating eTrust CA-ACF2 Logonid Records for z/OS Unix System Services MVS later in this chapter.

We recommend that you assign a unique GID to each group. If you assign the same GID value to multiple groups, the groups share ownership of and access to the same files. This could cause unreliable results. For example, if you assign multiple groups the same GID and a getgrgid() service request is made, only one group name is returned in response to the request. eTrust CA-ACF2 searches its cross-reference tables and returns the first group that matches the GID; it does not return all the groups associated with that GID, nor can it distinguish the specific group for which you intended the request to be made.

Supplemental Groups

Under z/OS Unix System Services and eTrust CA-ACF2 , a user is a member of the group defined in the GROUP field of his logonid, and a member of any other group that he has access to through a resource rule. These groups are called *supplemental groups* and a list of the allowed groups is built for each signon. When group access checks are performed for HFS file access, eTrust CA-ACF2 compares the GID of the file to the GID of the group defined in the logonid. If those GIDs do not match, eTrust CA-ACF2 checks to see if the file's GID matches the GID of any of the supplemental groups. If it matches, then eTrust CA-ACF2 uses the GROUP permissions to determine the user's access to the file.

To grant a user access to a supplemental group, you must create a TYPE(TGR) resource rule. The $KEY of the rule identifies the one to eight character group name. This field is maskable. The following is an example of a resource rule that grants access to group OMVSGRP to users in groupa and groupb.

```
$KEY(OMVSGRP) TYPE(TGR)
 UID(groupa) ALLOW
 UID(groupb) ALLOW
```

It is required that the TGR rules be resident, so you must create an entry for the TGR type code in the GSO INFODIR record and rebuild the resident directory. For details, see the "Maintaining Global System Options Records" chapter. You do not have to specify the GROUP field as part of the eTrust CA-ACF2 UID string for it to be used with z/OS Unix System Services. For more information about setting the owner, group, and other permissions for a file, see the IBM *z/OS UNIX System Services User's Guide*.

Note: The list of groups associated with a specific user's ACEE is pointed to by the ACEECGRP

The UNIXOPTS GSO record NGROUPS parameter sets the maximum size of the supplemental group list created for each signon. The number of entries is set within the range of 0 through 8192 with 300 being the default.

Controlling Applications that Invoke the R_ticketserver Callable Service

Authorized applications, such as servers, can invoke the R_ticketserv callable service to extract principal names from a GSS-API context token. This enables an application server to determine the client principal who originated an application-specific request, when the request includes a GSS-API context token and the intended receipt is the application server. For detailed information about invoking the R_ticketserv callable service, see the IBM's *z/OS SecureWay Security Server RACF Callable Services* guide.

Applications that run in system key or in supervisor state do not require eTrust CA-ACF2 authorization to use the R_ticketserv callable service. Applications that do not run in system key or in supervisor state require READ access from eTrust CA-ACF2 to the IRR.RTICKETSERV Facility class resource. For example:

```
Set Resource(FAC)

Compile
 .$KEY(IRR) TYPE(FAC)
 .RTICKETSERV UID(***USER1) SERVICE(READ) ALLOW

Store
```

In order for this rule to take effect you must rebuild the Facility class rules by issuing:

```
F ACF2,REBUILD(FAC)
```

Controlling Applications that Invoke the R_audix Callable Service

Authorized components can invoke the R_auditx callable service to log security events to SMF. Applications that run in system key or in supervisor state do not require eTrust CA-ACF2 authorization to use the R_auditx callable service. Applications that do not run in system key or in supervisor state require READ access to the IRR.RAUDITX resource in the FACILITY class.

```
Set Resource(FAC)
Compile
.$KEY(IRR) TYPE(FAC)
RAUDITX UID(***USER1) SERVICE(READ) ALLOW
Store
```

In order for this rule to take effect you must rebuild the Facility class rules by issuing :

```
F ACF2,REBUILD(FAC)
```

Controlling Applications that Invoke the R_dcekey Callable Service

Authorized components can invoke the R_dcekey callable service to set or retrieve a DCE password or retrieve an LDAP bind password. Applications that run in system key or in supervisor state do not require eTrust CA-ACF2 authorization to use the R_dcekey callable service. Applications that do not run in system key or in supervisor state require READ access to the BPX.SERVER or IRR.RDCEKEY resource in the FACILITY class.

```
Set Resource(FAC)
Compile
.$KEY(IRR) TYPE(FAC)
RDCEKEY UID(***USER1) SERVICE(READ) ALLOW
Store
```

In order for this rule to take effect you must rebuild the Facility class rules by issuing :

```
F ACF2,REBUILD(FAC)
```

Controlling Applications that Invoke the R_Getinfo Callable Service

Authorized servers can invoke the R_Getinfo callable service to retrieve a subset of security server information. Applications that run in system key or in supervisor state do not require eTrust CA-ACF2 authorization to use the R_Getinfo callable service. Applications that do not run in system key or in supervisor state require READ access to the BPX.SERVER resource in the FACILITY class. If Function_code 1 is specified, callers with read access to the IRR.RGETINFO.EIM resource in the FACILITY class will be allowed to use R_Getinfo.

```
Set Resource(FAC)
Compile
.$KEY(IRR) TYPE(FAC)
RGETINFO.EIM UID(***USER1) SERVICE(READ) ALLOW
Store
```

In order for this rule to take effect you must rebuild the Facility class rules by issuing :

```
F ACF2,REBUILD(FAC)
```

Controlling Applications that Invoke the PassTicket Subfunction of the R_ticketserv/R_GenSec Callable Service

Authorized and unauthorized applications can invoke the PassTicket subfunction of the R_ticketserv or R_GenSec callable service to generate or evaluate a PassTicket. For all callers, regardless of authorization, the use of the R_ticketserv or R_GenSec callable service to use PassTickets is authorized by resources in the PTKTDATA class. These resources are based on the application ID and target userid that were used in the PassTicket function. See the following table to determine what access is required to generate or evaluate a PassTicket using these callable services.

Operation	Resource Name	Access Required
Generate PassTicket	IRRPTAUTH.application.target-userid	UPDATE
Evaluate PassTicket	IRRPTAUTH.application.target-userid	READ

If the PTKTDATA class is not active or the resource rules are not defined, any PassTicket request made through the callable services will fail. All callers regardless of the PSW key or state must pass the authorization check.

The following is a sample resource rule allowing all users to generate or evaluate a PassTicket for USER1 on application TSOXE69 using the R_ticketserv or R_GenSec callable service.

```
 ACF
SET RESOURCE(FACF)
COMPILE
.$KEY(IRRPTAUTH) TYPE(PTK)
TSOXE69.USER1 UID(*) SERVICE(UPDATE,READ) ALLOW
STORE
```

Creating eTrust CA-ACF2 Logonid Records for z/OS Unix System Services

During the installation of z/OS Unix System Services, you must create a logonid for the OMVS started task and a logonid with z/OS Unix System Services superuser authority for one or more security administrators. In addition, any user accessing z/OS Unix System Services also must be defined. Use the steps outlined below to create these logonids.

Logonids Needed to Install z/OS Unix System Services

ServerPac and SystemPac Security Requirements

ServerPac and SystemPac Installation must be performed from a user ID that:

- Is a superuser (UID=0). Notice that you must be a superuser; just having access to the BPX.SUPERUSER facility class is not sufficient. This is because the pax utility is used to unload the ServerPac HFS and this utility does not use the BPX.SUPERUSER facility class to establish superuser identification.

  The following example shows how to define user OMVSUSR as a superuser. Since HOME and OMVSPGM are not explicitly specified, the defaults are taken for these fields.

  ```
  SET PROFILE(USER) DIV(OMVS)
  INSERT OMVSUSR UID(0)
  ```

- Is permitted read access to facility classes BPX.FILEATTR.APF and BPX.FILEATTR.PROGCTL.

  The following BPX Facility class rule can be coded to accomplish this:

  ```
  $KEY(BPX) TYPE(FAC)
   FILEATTR.- UID(user's_ uid) SERVICE(READ) ALLOW
   SUPERUSER UID(user's_uid) ALLOW
  ```

  Note: If the FACILITY class BPX rule already exists, simply add the rule entries to the existing rule set.

Security Requirements for z/OS Installs Using the CBPDO Method

When you use the CBPDO method of installing z/OS, you install in four stages called waves. This section describes the driving system requirements for each wave. In Wave 1 you establish an activated OMVS address space with z/OS UNIX kernel services operating in full function mode. This installation should be performed from a user ID that:

- Is a superuser (UID=0) or has access to the BPX.SUPERUSER facility resource.
- Is permitted read access to facility classes BPX.FILEATTR.APF and BPX.FILEATTR.PROGCTL.

  The following BPX Facility class rule can be coded to accomplish this:

```
$KEY(BPX) TYPE(FAC)
 FILEATTR.- UID(user's_ uid) SERVICE(READ) ALLOW
 SUPERUSER UID(user's_uid) ALLOW
```

  Note: If the FACILITY class BPX rule already exists, simply add the rule entries to the existing rule set.

Defining the OMVS Started Task Logonid

First, make sure you are running with STC set in the GSO OPTS record. Then, use the following steps to create the OMVS started task logonid. For detailed information about the administrative tasks of inserting logonids or checking GSO record settings, see the "Maintaining Logonid Records" chapter.

1. Create the z/OS Unix System Services kernel started task logonid and profile records by issuing the following eTrust CA-ACF2 subcommands:

```
SET LID
INSERT OMVS NAME(USS ID) GROUP(OMVSGRP) STC UID(0) HOME(/) OMVSPGM(/bin/sh)
NOMAXVIO
```

   This example shows logonid OMVS created as a started task (STC is specified) and assigned to a group called OMVSGRP. Remember that the OMVS started task, as with any address space, needs access to the resources that it uses. Ensure that the OMVS started task has access to any resource or data set that it needs. In accordance with z/OS Unix System Services requirements, giving logonid OMVS a UID of zero designates it as a superuser. Also, the NOMAXVIO attribute prevents eTrust CA-ACF2 from canceling the OMVS address space due to violations it might encounter in processing.

2. Create a group profile record to define a GID for the group to which the logonid belongs by issuing the following eTrust CA-ACF2 subcommands:

```
SET PROFILE(GROUP) DIV(OMVS)
INSERT OMVSGRP GID(1)
```

   In this example, the group OMVSGRP is given a GID value of one.

3. A logonid and profile record must be defined for the BPXOINIT started task as follows:

```
SET LID
INSERT BPXOINIT NAME(BPXOINIT STCID) STC GROUP(OMVSGRP) UID(0) HOME(/)
OMVSPGM(/bin/sh)
```

4. A logonid must be defined for the BPXAS started task as follows:

```
SET LID
INSERT BPXAS NAME(BPXAS STCID) STC GROUP(OMVSGRP)
```

5. The HFS file system and the files contained within this system are defined to z/OS as data sets. The OMVS started task must be given WRITE and ALLOCATE access to the data sets that make up the HFS mountable files. Be aware that your naming conventions for the data sets containing HFS mountable files affect how you write the eTrust CA-ACF2 access rules giving the OMVS started task access to these data sets.

   For example, if you mount the data set USER1.HFS.FILE to the /u/USER1 mount point, you must add an entry in the USER1 access rule allowing OMVS access to the data set. However, if you use a single high level index, such as OMVS, you only need one rule set to allow access to all of your mountable files. For example, if the data set name is OMVS.USER1.HFS.FILE, you could write a generic rule entry to allow access to multiple data sets as follows:

```
$KEY(OMVS)
 - UID(omvs) R(A) W(A) A(A)
```

   For more information about mountable files, see the IBM document entitled *z/OS UNIX System Services Planning*.

6. The use of profile records requires that eTrust CA-ACF2 build resident directories for these records. To implement this, add the following to the GSO INFODIR record:

```
SET CONTROL(GSO)
CHANGE INFODIR TYPES(R-PUSR,R-PGRP)
```

   To activate the new records, issue the following operator commands:

```
F ACF2,REFRESH(INFODIR)
F ACF2,REBUILD(USR),CLASS(P)
F ACF2,REBUILD(GRP),CLASS(P)
F ACF2,OMVS
```

   Note: This process is required anytime profile records are created or changed to place the changes into effect. This process is automatic when eTrust CA-ACF2 is started.

7. If access is denied, then eTrust CA-ACF2 performs the permission bit checking maintaining an audit trail of violations.

You must create a FAC type rule with a $KEY of BPX as BPX is the high level for a number of facility checks related to z/OS Unix System Services processing authorization. To initially set up this rule, you must create the following control card and compile the rule:

```
$KEY(BPX) TYPE(FAC)
```

Then add the following rule entry to the BPX rule to turn off the FASTPATH processing.

```
SAFFASTPATH UID(*) PREVENT
```

Sites that add the NON-CNCL privilege to the OMVS logonid will always return an allow condition to the BPX.SAFFASTPATH resource check. This will always result in a bypass of the audit trail as described above. To avoid this problem and to disable FASTPATH processing you must insert the following SAFDEF record:

```
INSERT SAFDEF.OEFSTART FUNCRET(4) ID(OEFSTAUT) JOBNAME(OMVS) MODE(IGNORE) –
    RB(BPX-) RACROUTE(REQUEST=AUTH CLASS=FACILITY ENTITY=BPX.SAFFASTPATH) -
    REP
```

Defining Additional Started Task Logonids

If your site is using TCP/IP, the INETD daemon, or a started task that monitors OMVS, such as RMFGAT, you must create started task logonids for them and define these IDs to OMVS.

1. Create the logonid and user profile records by issuing the following subcommands:

```
SET LID
INSERT TCPIP NAME(TCPIP STCID) STC GROUP(OMVSGRP) UID(0) HOME(/)
OMVSPGM(/bin/sh)

INSERT INETD NAME(INETD STCID) STC GROUP(OMVSGRP) UID(0) HOME(/)
OMVSPGM(/bin/sh)

INSERT RMFGAT NAME(RMFGAT STCID) STC GROUP(OMVSGRP) UID(0) HOME(/)
OMVSPGM(/bin/sh)

INSERT RESOLVER NAME(OMVS RESOLVER) GROUP(OMVSGRP) STC UID(42) HOME(/)
```

2. Create an additional TTY group profile record. This is a default group used by OMVS and needs to be assigned a GID:

```
SET PROFILE(GROUP) DIV(OMVS)
INSERT TTY GID(nn)
```

Replace the *nn* with an appropriate GID number.

3. Define additional started tasks logonids and their associated OMVS profile records:

```
INSERT OMPROUTE NAME(OMPROUTE STC Logonid) STC GROUP(OMVSGRP) UID(0) HOME(/)
INSERT OROUTED NAME(OROUTED STC Logonid) STC GROUP(OMVSGRP) UID(0) HOME(/)
INSERT OSNMPD NAME(OSNMPD STC Logonid) STC GROUP(OMVSGRP) UID(0) HOME(/)
INSERT PAGENT NAME(PAGENT STC Logonid) STC GROUP(OMVSGRP) UID(0) HOME(/)
INSERT PAGTSNMP NAME(PAGTSNMP STC Logonid) STC GROUP(OMVSGRP) UID(0) HOME(/)
INSERT PORTMAP NAME(PORTMAP STC Logonid) STC GROUP(OMVSGRP) UID(0)
INSERT SENDMAIL NAME(SENDMAIL STC Logonid) STC GROUP(OMVSGRP) UID(0) HOME(/)
INSERT SNMPQE NAME(SNMPQE STC Logonid) STC GROUP(OMVSGRP) UID(0) HOME(/)
INSERT SYSLOGD NAME(SYSLOGD STC Logonid) STC GROUP(OMVSGRP) UID(0) HOME(/)
INSERT TRAPFWD NAME(TRAPFWD STC Logonid) STC GROUP(OMVSGRP) UID(0) HOME(/)
INSERT NAMED NAME(NAMED STC Logonid) STC GROUP(OMVSGRP) UID(0) HOME(/)
INSERT DCAS NAME(DCAS STC Logonid) STC GROUP(OMVSGRP) UID(0) HOME(/)
```

4. The SYSLOGD logonid needs access to the facility class resource BPX.SMF so that SYSLOGD can write records to SMF. Add the following entry to the facility class BPX resource rule:

```
UID(syslogd_uid)
```

5. Various servers also require access to the facility class resource BPX.DAEMON. Add the following rule entries to the facility class BPX resource rule:

```
DAEMON UID(tcpip_uid) ALLOW
DAEMON UID(ftpd_uid) ALLOW
DAEMON UID(osnmpd_uid) ALLOW
DAEMON UID(syslogd_uid) ALLOW
```

Controlling Access to Superuser Status

The superuser command lets users under OMVS switch to the identity of another user. If no ID is specified, the user switches to a superuser, UID(0). You might want to use this facility as an alternative to assigning a user a UID of zero. The ability to switch to superuser status is controlled through the FACILITY class resource BPX.SUPERUSER. To control which users have the ability to use the su command, add rule entries to the BPX FACILITY resources rule, similar to this one, as appropriate:

```
SUPERUSER UID(user_uid) ALLOW
```

Controlling Access to Daemons

Daemons are long-running OMVS tasks that perform tasks on behalf of the users that use the daemons. They often use z/OS Unix System Services processes such as setuid(), seteuid(), setreuid() and spawn(). These services allow the daemon to change the caller's z/OS user identity. The FACILITY class resource BPX.DAEMON regulates who can use these powerful services. The OMVS kernel, OMVS, requires daemon authority. Daemons always run as superusers so they require a UID of 0. To allow daemon processes to invoke setuid() for superusers, define a superuser with a userid of BPXROOT on all systems.

Note: BPXROOT is the default userid used in the z/OS documentation. If your site has changed this default userid, substitute that userid for BPXROOT.

Give daemon authority to the OMVS kernel by adding the following rule entry to the BPX FACILITY resource rule:

```
DAEMON UID(omvs)  SERVICE(READ) ALLOW
```

Use the following commands to define the BPXROOT logonid and user profile records:

```
SET LID
INSERT BPXROOT GROUP(OMVSGRP) UID(0) HOME(/) OMVSPGM(/bin/sh)
```

Note: The main difference between the OMVS logonid and the BPXROOT logonid is that BPXROOT does not have access to the FACILITY class resource BPX.DAEMON. BPXROOT is used when a daemon process invokes setuid() to change the UID to 0 and the user name has not been previously identified by getpwnam() or by the _passwd() function. This prevents giving daemon authority to a superuser who is not defined to the BPX.DAEMON resource.

For more information about granting access to daemons, see the IBM document entitled *z/OS UNIX System Services Planning*.

Defining Servers to Use Thread-Level Security

The FACILITY class resource BPX.SERVER controls access to the pthread_security_np service. This service lets the server establish a security context (task level ACEE) similar to an eTrust CA-ACF2 MUSASS environment. This process is called *surrogation* and it validates access to resources using the eTrust CA-ACF2 logonid of the user (client) instead of the server's own logonid. This resource check establishes users for whom the server can act as a surrogate. BPX.SERVER is also used to determine z/OS resources that the server can access while acting as a surrogate for its clients.

When a server is given UPDATE access to the FACILITY class resource BPX.SERVER, this server can act as a surrogate for any client. The identity of the thread associated with the request from the server's client runs with the z/OS logonid of the client. All access control decisions to resources that are accessed by the client's thread are made using the eTrust CA-ACF2 logonid of the client. To define this level of security to eTrust CA-ACF2, add the necessary rule entry for each server to the BPX FACILITY resource rule:

```
SERVER UID(server1) SERVICE(READ,UPDATE) ALLOW
```

If you give the server only READ access to the FACILITY class resource BPX.SERVER, the thread associated with the request from the server's client runs with the z/OS logonid of the client. However, access control decisions to resources that are accessed by the client's thread depend on whether a password is supplied by the client. If the application obtains the client's password and supplies this password when using the security service, the task level ACEE created is for an authenticated client. In this case, the server is capable of acting as a surrogate for the client. Access control decisions made by the client use the eTrust CA-ACF2 logonid of the client. If a password is not supplied, the task level ACEE created is for an unauthenticated client. In this case, both the eTrust CA-ACF2 logonid of the client and the eTrust CA-ACF2 logonid of the server are used when making access control decisions to z/OS resources. When a password is not supplied, resource rules must be written for BPX.SRV.userid in the SURROGAT class to allow the server to act as surrogate for clients that do not supply passwords.

The following example gives READ access to BPX.SERVER for the SERVER1 application. Add the following rule entry to the BPX FACILITY resource rule:

```
SERVER UID(server1) SERVICE(READ) ALLOW
```

The following example allows SERVER1 to act as a surrogate for the unauthenticated client USER1:

```
SET RESOURCE(SUR)
COMPILE
$KEY(BPX) TYPE(SUR)
SRV.USER1 UID(server1) ALLOW
```

Note: If the SURROGAT class BPX resource rule already exists, add the appropriate rule entries to the existing resource rule.

TSO ISHELL Support

IBM provides a TSO REXX exec, BPXWIRAC, which is used to interface between the OMVS ISHELL and RACF. eTrust CA-ACF2 provides a substitute for this exec in the eTrust CA-ACF2 CAIEXEC data set. This file contains a member with the name BPXWIRAC. To install this, ensure that the eTrust CA-ACF2 CAIEXEC data set is added to the desired TSO PROC and ensure that this data set is concatenated ahead of the data set containing the IBM version of BPXWIRAC.

The BPXISEC1 CLIST also contains the initial commands to set up required z/OS Unix System Services definitions. An eTrust CA-ACF2 version of this is supplied in CAI.SAMPJCL member ACFISEC.

Note: The logonid being used to process these execs requires the appropriate eTrust CA-ACF2 and TSO authorities to execute the commands contained within the exec.

Creating an Administrator ID

The z/OS Unix System Services shell and utilities installation process creates directories in the Hierarchical File System (HFS). To perform the installation steps, the user must have superuser authority.

To create a superuser administrator logonid and give it the authority it needs, follow these steps (logonid assumed to already exist):

1. Define the logonid as a superuser by issuing the following eTrust CA-ACF2 subcommands:

   ```
   SET LID
   INSERT sysadmin-lid UID(0) HOME(/) OMVSPGM(/bin/sh)
   ```

   Logonid SYSADMIN is defined as a superuser by setting the UID value to zero.

2. Define the existing logonid as a member of a group by issuing these eTrust CA-ACF2 subcommands:

   ```
   SET LID
   CHANGE sysadmin-lid GROUP(sysadmin-group)
   ```

   The example shows a logonid changed to give it a group value so that this user can sign on and be validated as a member of that group. The members of this group would be a special subset of users who perform system-related tasks.

3 Assign the group a GID value by issuing these eTrust CA-ACF2 subcommands:

```
SET PROFILE(GROUP) DIV(OMVS)
INSERT sysadmin-group GID(19)
```

In this example, the selected group is assigned a GID of 19, defining it for use in z/OS Unix System Services.

4. Set up the GROUP and USER profile records for residency by adding them to the GSO INFODIR record as follows:

```
CHANGE INFODIR TYPES(R-PGRP)
CHANGE INFODIR TYPES(R-PUSR)
```

5. Refresh the INFODIR record to make the change effective immediately by issuing this command:

```
F ACF2,REFRESH(INFODIR)
```

6. Rebuild the user and group profile directories to pull the changes into storage and then rebuild the OMVS table by issuing the following commands:

```
F ACF2,REBUILD(GRP),CLASS(P)
F ACF2,REBUILD(USR),CLASS(P)
F ACF2,OMVS
```

USER Profile Records

z/OS Unix System Services UIDs are defined to eTrust CA-ACF2 in user profile records in the eTrust CA-ACF2 Infostorage database. Specifically, you define the UID information in the OMVS segment of the user profile record. This segment contains three basic fields: UID, HOME, and OMVSPGM and six override fields..

See Profile support for user limit overrides for details on the six override fields.

- UID is a numeric field that accepts values from zero to 2,147,483,647. A UID defined with a value of zero indicates that this user is a superuser. For additional information on superusers, see the IBM *z/OS UNIX System Services User's Guide*. This field does not have to be unique, but we recommend that you make it unique; otherwise, individual accountability and control are lost. You should generally reserve the lower number range for system use. **This field is required.**

  Note: Do not include commas when entering a number in profiles.

- The HOME field defines the initial directory pathname. This is the initial directory used when a user enters the OMVS command or enters the ISPF shell. The HOME field accepts values that are from one to 1023 characters in length. Both upper and lower case characters are allowed. If HOME is not defined, z/OS Unix System Services sets the initial directory for the user to the root directory. This field is optional.

- The OMVSPGM field defines the pathname of the user's z/OS Unix System Services shell program, which is the first program started when the OMVS command is entered or when a z/OS Unix System Services batch job is started using the BPXBATCH program. The OMVSPGM field accepts values that are from one to 1023 characters in length. Both upper and lower case characters are allowed. If a value is not entered in the omvspgm field, z/OS Unix System Services gives control to the default shell program. This field is optional.

The following example shows how to define user OMVSUSR as a superuser. Since HOME and OMVSPGM are not explicitly specified, the defaults are taken for these fields.

```
SET PROFILE(USER) DIV(OMVS)
INSERT OMVSUSR UID(0)
```

This example shows how to define user OMVSU2 as a regular user. The HOME and OMVSPGM fields are defined.

```
SET PROFILE(USER) DIV(OMVS)
INSERT OMVSU2 UID(199) HOME(/u/omvsu2) OMVSPGM(/bin/sh)
```

After inserting or changing any user profile records, rebuild the user profile directory as documented in the Operator Commands for z/OS Unix System Services section later in this chapter. This directory must be rebuilt before the new or changed logonid attempts to access z/OS Unix System Services resources; otherwise, the change is not recognized and access is denied. Also, remember that eTrust CA-ACF2 requires that profile user records be resident in storage. Before doing any rebuilds, ensure that these records are designated resident. To accomplish this, add them to the GSO INFODIR record with the R parameter as follows:

```
CHANGE INFODIR TYPES(R-PUSR)
```

Profile Support for User Limit Overrides

The user profile record contains six fields that are used to support user limit overrides. These fields supply values needed by UNIX System Services to verify a user's access. These fields are:

CPUTIME
This field overrides for this user the MAXCPUTIME parameter in the BPXPRMxx member of parmlib. The value can be from 7 to 2,147,483,647. This field is not set when the record is inserted. If the field is not set, a RACROUTE EXTRACT will return a length of 4 and a value of X'FFFFFFFF'. UNIX System Services will take the system defaults set in BPXPRMxx when this field is not set. You can remove this field from the record by changing it to a null value (change user CPUTIME()).

ASSIZE

This field overrides for this user the MAXASSIZE parameter in the BPXPRMxx member of parmlib. The value can be from 10,485,760 to 2,147,483,647. This field is not set when the record is inserted. If the field is not set, a RACROUTE EXTRACT will return a length of 4 and a value of X'FFFFFFFF'. UNIX System Services will take the system defaults set in BPXPRMxx when this field is not set. You can remove this field from the record by changing it to a null value (change user ASSIZE()).

FILEPROC

This field overrides for this user the MAXFILEPROC parameter in the BPXPRMxx member of parmlib. The value can be from 3 to 65,535. This field is not set when the record is inserted. If the field is not set, a RACROUTE EXTRACT will return a length of 4 and a value of X'FFFFFFFF'. UNIX System Services will take the system defaults set in BPXPRMxx when this field is not set. You can remove this field from the record by changing it to a null value (change user FILEPROC()).

MEMLIMIT

This field specifies the maximum number of bytes of non-shared memory space that the user can allocate. The value can be from 0 to 16,777,215 followed by a multiplier that is used to calculate the total number of bytes (M for megabyte, G for gigabyte, T for terabyte, and P for petabyte). The maximum value is 16383P. This field is not set when the record is inserted. If the filed is not set, a RACROUTE EXTRACT will return a length of 4 and a value of X'FFFFFFFF'. You can remove this field from the record by changing it to a null value (MEMLIMIT()). MEMLIMIT is only valid at z/OS 1.6 and above.

PROCUSER

This field overrides for this user the MAXPROCUSER parameter in the BPXPRMxx member of parmlib. The value can be from 3 to 32,767. This field is not set when the record is inserted. If the field is not set, a RACROUTE EXTRACT will return a length of 4 and a value of X'FFFFFFFF'. UNIX System Services will take the system defaults set in BPXPRMxx when this field is not set. You can remove this field from the record by changing it to a null value (change user PROCUSER()).

SHMEMMAX

This field specifies the maximum number of bytes of shared memory space that the user can allocate. The value can be from 1 to 16,777,215 followed by a multiplier that is used to calculate the total number of bytes (M for megabyte, G for gigabyte, T for terabyte, and P for petabyte). The maximum value is 16383P. This field is not set when the record is inserted. If the field is not set, a RACROUTE EXTRACT will return a length of 4 and a value of X'FFFFFFFF'. UNIX System Services will take the value set in the IPCSHMNSEGS parameter in the BPXPRMxx when this field is not set. You can remove this field from the record by changing it to a null value (SHMEMMAX()). SHMEMMAX is only valid at z/OS 1.6 and above.

The multiplier table for MEMLIMIT and SHMEMMAX describes the multiplier value used to calculate the total number of bytes.

Multiplier	Decimal	Binary	Hex
M=Megabyte	1,048,576	2**20	00000000 00100000
G=Gigabyte	1,073,741,824	2**30	00000000 40000000
T=Terabyte	1,099,511,627,776	2**40	00000010 00000000
P=Petabyte	1,125,899,906,842,624	2**50	00040000 00000000

The following example allows OMVSU2 to allocate a maximum of 18,014,397,740,160 bytes of shared memory space.

```
CHANGE OMVSU2 SHMEMMAX(16777215G)

OMVS/OMVSU2 LAST CHANGED BY TLC250 ON 11/2/04-17:30
        FILEPROC(3) HOME(/u/omvsu2) MMAPAREA(10)
        OMVSPGM(/bin/sh) PROCUSER(3) SHMEMMAX(16777215G) UID(199)
```

The following example allows OMVSU2 to allocate a maximum of 1,099,511,627,776 bytes of non-shared memory space.

```
CHANGE OMVSU2 MEMLIMIT(1T)

OMVS/OMVSU2 LAST CHANGED BY TLC250 ON 11/2/04-17:38
         FILEPROC(3) HOME(/u/omvsu2) MEMLIMT(1T) MMAPAREA(1)
         OMVSPGM(/bin/sh) PROCUSER(3) SHMEMMAX(1T) UID(199)
```

THREADS

This field overrides for this user the MAXTHREADS parameter in the BPXPRMxx member of parmlib. The value can be from 0 to 100,000. This field is not set when the record is inserted. If the field is not set, a RACROUTE EXTRACT will return a length of 4 and a value of X'FFFFFFFF'. UNIX System Services will take the system defaults set in BPXPRMxx when this field is not set. You can remove this field from the record by changing it to a null value (change user THREADS()).

MMAPAREA

This field overrides for this user the MAXMMAPAREA parameter in the BPXPRMxx member of parmlib. The value can be from 1 to 16,777,216. This field is not set when the record is inserted. If the field is not set, a RACROUTE EXTRACT will return a length of 4 and a value of X'FFFFFFFF'. UNIX System Services will take the system defaults set in BPXPRMxx when this field is not set. You can remove this field from the record by changing it to a null value (change user MMAPAREA()).

```
SET PROFILE(USER) DIV(OMVS)
CHANGE OMVSU2 CPUTIME(200) ASSIZE(10485760) FILEPROC(3) –
       PROCUSER(3) THREADS(10) MMAPAREA(1)

OMVS / OMVSU2 LAST CHANGED BY TLC250 ON 05/17/99 – 13:38
       ASSIZE(10,485,760) CPUTIME(200) FILEPROC(3) HOME(/u/omvsu2)
       MMAPAREA(1) PROCUSER(3) OMVSPGM(/bin/sh) THREADS(10) UID(199)
```

Normally, you cannot remove values from binary fields in eTrust CA-ACF2. However, the override fields in the user profile record have been designed to allow this. You can remove values by specifying a null value for the field. This results in eTrust CA-ACF2 ignoring the field, and causes UNIX System Services to pick up the default value specified in the BPXPRMxx member. The following example shows how to remove a value from a field.

```
SET PROFILE(USER) DIV(OMVS)
CHANGE OMVSU2 CPUTIME() THREADS() ASSIZE()

OMVS / OMVSU2 LAST CHANGED BY TLC250 ON 05/21/99 – 10:48
       FILEPROC(3) HOME(/u/omvsu2) MMAPAREA(1) PROCUSER(3)
       OMVSPGM(/bin/sh) UID(199)
```

GROUP Profile Records

z/OS Unix System Services groups are defined to eTrust CA-ACF2 as group profile records in the eTrust CA-ACF2 Infostorage database. Specifically, you define the group in the OMVS segment, which contains one field, the GID field.

GID is a numeric field that accepts values from zero to 2,147,483,647. This value does not need to be unique, but we recommend that you make the GID unique; otherwise, control over a particular group is lost.

Note: Do not include commas when entering numbers in eTrust CA-ACF2 profiles.

This example shows how to define an OMVS group profile record for a group called OMVSGRP and assign it a GID of 20.

```
SET PROFILE(GROUP) DIV(OMVS)

INSERT OMVSGRP GID(20)
```

After inserting or changing any group profile records, rebuild the group profile directory as documented in the Operator Commands for z/OS Unix System Services section later in this chapter. This directory must be rebuilt before the new or changed logonid attempts to access z/OS Unix System Services resources; otherwise, the change is not recognized and access is denied. Also, remember that eTrust CA-ACF2 requires that profile GROUP records be resident in storage. Before doing any rebuilds, ensure that the records are designated resident. Add them to the GSO INFODIR record with the R parameter as follows:

```
CHANGE INFODIR TYPES(R-PGRP)
```

Assigning Users to Groups under eTrust CA-ACF2

You assign a user's default group by setting the GROUP field in that user's eTrust CA-ACF2 logonid. The example below shows you how to assign logonid OMVSU2 to group OMVSGRP:

```
SET LID
CHANGE OMVSU2 GROUP(OMVSGRP)
```

Note: Users can change their group by specifying GROUP(*groupname*) with their logonid and password when they log on to TSO. Resource rules control the users' access to groups not specified in their logonids.

Displaying UID and GID Numbers

SHOW OMVS[ALL | GROUPS(mmmm[-nnnn]) | SUPERUSERS | USERS(mmmm[-nnnn])][Duplicates]

The SHOW OMVS displays z/OS Unix System Services users and/or groups.

- ALL—displays all defined UIDs and GIDs along with their associated userids.
- GROUPS(mmmm[-nnnn])—displays a range of GID values along with their associated userids.
- SUPERUSERS—displays all superusers (UID of ero(0)) along with their associated userids.
- USERS(mmmm[-nnnn])—displays a range of UID values along with their associated userids.
- Duplicates—shows only the UID and GID values that belong to more than one user or group. The DUPLICATES keyword can be used together with another keyword. Example: SHOW OMVS USERS(1-2000) DUPLICATES will show only duplicate UID values that are in the range 1 to 2000.

ALL is the default.

```
SHOW OMVS ALL
---------- OPENEDITION MVS DISPLAY -----------
               -- OMVS USERS --

             UID               NAME
             ================  ========
             0                 BPXAS
             0                 BPXOINIT
             0                 BPXROOT
             7                 GUEST3
             101               TEST
             8,888,888         OMVSC
```

```
  -- OMVS GROUPS --

GID               NAME
================  ========
0                 NULLGRP
0                 ZEROGRP
11                LDAPGRP
44,444            OMVSG
99,999,999        OMVSDGRP
```

Defining a Default OMVS UID and GID

z/OS Unix System Services requires that users attempting to signon be defined with a valid UID and GID. If they do not have both of these, access is denied. If desired, a default OMVS userid and group can be defined allowing users without a UID or GID to access z/OS Unix System Services. The DFTUSER and DFTGROUP fields of the GSO UNIXOPTS record provide this capability. These defaults require that eTrust CA-ACF2 profile records be defined for them. In addition, z/OS Unix System Services issues a SAF EXTRACT call for the entity BPX.DEFAULT.USER. No rules are needed for this, but, in order for eTrust CA-ACF2 to satisfy this call, the default OMVS logonid must exist.

Following is an example of setting up defaults for OMVS:

```
SET LID
INSERT OMVSUSER NAME(OMVS DEFAULT USERID) UID(99999999) HOME(/) OMVSPGM(/bin/sh)

SET CONTROL(GSO)
CHANGE UNIXOPTS DFTUSER(OMVSUSER) DFTGROUP(OMVSDGRP)

SET PROFILE(GROUP) DIV(OMVS)
INSERT OMVSDGRP GID(99999999)
```

Note: If the GSO PSWD record has the password required (PSWDREQ) set, the default logonid needs the RESTRICT bit set for it also.

In cases where the default user is defined to grant access to facilities such as FTP, but where access to the rest of z/OS Unix System Services is not desired, you can define the program value for the UID profile as /bin/echo. This will allow the default to function for use with FTP, but disallow access to the OMVS shell (access is allowed to the ISHELL through TSO). It will also stop the OMVS command, telneting using OTELNET, RLOGIN connections, the OSHELL command, BPXBATCH using the sh parm and a command send to the shell via ISHELL using ISHELL's Run Shell command facility.

When a user that does not have an OMVS USER PROFILE record attempts to access z/OS Unix System Services, eTrust CA-ACF2 uses the userid defined in the DFTUSER field of the UNIXOPTS record.

If a user attempting to access z/OS Unix System Services does not have a value in his logonid GROUP field or the value in the GROUP field is not defined to z/OS Unix System Services, eTrust CA-ACF2 uses the group value defined in the DFTGROUP field of the UNIXOPTS record.

The NO-OMVS logonid attribute lets a site specify that the logonid is not eligible to use OMVS. A logonid with the NO-OMVS attribute set is denied access at signon.

Controlling Superuser Functions

OMVS requires that users performing certain functions have a UID(0) or superuser status. Once a user is given superuser status, they have complete access to the system. The UNIXPRIV class allows specific control of the individual functions usually performed by a user with superuser authority. This is referred to as superuser granularity. The following table lists the new resources and what access or function is controlled by that resource:

Note: See RACF Attribute Translation in the appendix "RACF to eTrust CA-ACF2 Translation," for more information about the access levels allowed for the resources.

Resource name	Access Given	Affected Functions
SUPERUSER.FILESYS (READ access)	Allows a user to read any HFS file and read or search any HFS directory	OPEN (for read or search) OPENDIR, READLINK, STAT REALPATH, LSTAT, EACCESS (for read) ACCESS (for read); if real, effective and saved UID match)
SUPERUSER.FILESYS (UPDATE access)	Allows a user to write to any existing HFS file.	OPEN (for write), EACCESS (for write), ACCESS (for write; if real, effective and saved UID match)
SUPERUSER.FILESYS (DELETE access)	Allows a user to write to any HFS directory.	LINK, MKDIR, RENAME, RMDIR, SYMLINK, UNLINK

Resource name	Access Given	Affected Functions
SUPERUSER.FILESYS (Control access)	Allows a user to write to any HFS directory.	Link(), mkdir(), rename(), rmdir(), symlink(), unlink()
SUPERUSER.FILESYS.ACLOVERRIDE (READ access)	Allows a user with access to SUPERUSER.FILESYS to override an ACL that denies access	OPEN (for read or search), OPENDIR, READLINK, STAT, REALPATH, LSTAT, EACCESS (for read), ACCESS(for read; if real, effective and saved UID match)
SUPERUSER.FILESYS.ACLOVERRIDE (UPDATE access)	Allows a user with access to SUPERUSER.FILESYS to override an ACL that denies access.	OPEN (for write), EACCESS (for write), ACCESS (for write; if real, effective and saved UID match)
SUPERUSER.FILESYS.ACLOVERRIDE (DELETE access)	Allows a user with access to SUPERUSER.FILESYS to override an ACL that denies access.	LINK, MKDIR, RENAME, RMDIR, SYMLINK, UNLINK
SUPERUSER.FILESYS.CHANGEPERMS (READ access or higher)	Allows a user to change the access mode of any file.	chmod(), setfacl()
SUPERUSER.FILESYS.CHOWN	Allows a user to change ownership of any file.	chown()
SUPERUSER.FILESYS.MOUNT	Allows a user to issue mount, unmount, quiesce, and unquiesce requests.	Mount(), unmount(), quiesce(), unquiesce()
SUPERUSER.FILESYS.PFSCTL	Allows a user to call pfsctl().	Pfsctl()
SUPERUSER.FILESYS.VREGISTER	Allows a user to issue vregister() to register as a vfs file server.	Vregister()
SUPERUSER.IPC.RMID	Allows a user to do ipcrm calls to clean up leftover IPC mechanisms.	Ipcrm command use of IPC_RMID for msgctl(), semctl(), shmctl()
SUPERUSER.PROCESS.GETPSENT	Allows a user to see all processes.	Getpsent() – ps command
SUPERUSER.PROCESS.KILL	Allows a user to send signals to any process.	Kill()

Resource name	Access Given	Affected Functions
SUPERUSER.PROCESS.PTRACE	Allows a user to use dbx to trace any process.	Dbx
SUPERUSER.SETPRIORITY	Allows a user to increase his priority.	Setpriority(), nice()

Using the UNIXPRIV class means that a user does not need superuser authority to perform an individual function from the above table. When a user attempts to perform the function without a UID(0) or superuser authority, eTrust CA-ACF2 issues a resource check to see if that user is allowed to perform the function. If the resource rule allows access to the resource associated with the function, the user is allowed to perform the function even though they do not have UID(0).

The following example shows a rule that allows USERA to read all HFS files, change the ownership of any file, and see all processes using the ps command:

```
SET RESOURCE(UNI)
LIST SUPERUSER

ACF75052 RESOURCE RULE SUPERUSER STORED BY USER01 ON 05/04/99 – 12:26
$KEY(SUPERUSER) TYPE(UNI)
 FILESYS UID(usera) SERVICE(READ) ALLOW
 FILESYS.CHOWN UID(usera) ALLOW
 PROCESS.GETPSENT UID(usera) ALLOW
ACF75051 TOTAL RECORD LENGTH = 268 BYTES, 6 PERCENT UTILIZED
```

The UNIXPRIV class uses FASTAUTH calls so the type code used for the UNIXPRIV class must be added to the GSO INFODIR record and rules must be made resident:

```
SET CONTROL(GSO)
CHANGE INFODIR TYPES(R-RUNI)
```

Once the INFODIR record has been updated, issue the following commands to activate the changes:

```
F ACF2,REFRESH(INFODIR)
F ACF2,REBUILD(UNI),CLASS(R)
```

Operator Commands for z/OS Unix System Services

Rebuilding Unix System Services Cross-Reference Tables

Any time you create, change, or delete a user or group profile record, you must rebuild the directories for the profile records so that the changed are recognized. Also, remember that the profile records must be designated resident in the INFODIR record **before** you rebuild the user or group profile directory.

To rebuild the directory for user profile records, issue this command:

```
F ACF2,REBUILD(USR),CLASS(P)
```

To rebuild the directory for group profile records, issue this command:

```
F ACF2,REBUILD(GRP),CLASS(P)
```

During initialization eTrust CA-ACF2 also builds cross-reference tables to associate UIDs with their eTrust CA-ACF2 logonids and GIDs with the eTrust CA-ACF2 group name. If you add, change, or delete entries in user or group profile records, these tables must be rebuilt so that the associations are accurate and current. Issue this operator command to rebuild these tables after you have rebuilt the directories:

```
F ACF2,OMVS(table,table,.,table|ALL)
```

See the *System Programmers Guide* for more information about the F ACF2,OMVS command.

Tracing z/OS Unix System Services SAF Requests

The SECTRACE facility, used to trace SAF requests in the eTrust CA-ACF2 environment, is also available to trace SAF requests made by OMVS.

For complete details on the SECTRACE syntax see the Special Usage Considerations chapter in the *System Programmers Guide*.

To start SECTRACE for OMVS, issue the following command:

```
ST SET,TYPE=OMVS,ID=aaaa,FUNC=xxxx,end
```

Note: FUNC= is a required parameter.

ID names the SECTRACE trap and can be from one to eight characters. FUNC can be one of seven values. Each of the functions traces a set of related OMVS services. The seven functions and the services that they trace are:

ALL
Traces all OMVS services.

CHANGE
Traces R_chown, R_chaudit, and R_chmod.

CHECK
Traces ck_access, ck_priv, ck_process_owner, ck_file_owner, R_ptrace, ck_IPC_access, ck_owner_two_files, R_IPC_ctl, and R_dceauth.

GET
Traces getUMAP, getGMAP, R_getgroups, R_getgroupsbyname, get_uid_gid_supgrps, R_datalib, R_dceinfo, R_dcekey, R_dceruid, R_kerbinfo, R_ticketserv, and R_usermap.

INIT
Traces initACEE, initUSP, deleteUSP, and R_fork.

MAKE
Traces makeFSP, makeISP, and make_root_FSP.

MISC
Traces audit, query_file_security_options, query_system_security_options, R_cachserv, R_proxyserv, R_PKIServ, R_writepriv.

SET
Traces R_umask, R_setegid, R_seteuid, R_setfacl, R_setgid, R_setuid, R_exec, clear_setid, and R_admin.

The OMVS SECTRACE will accept other parameters on the ST SET statement, but will ignore them. Output from the OMVS SECTRACE can only be routed to the console.

The OMVS services are documented in the IBM *z/OS Security Services Callable Services* guide. You should only use the OMVS SECTRACE when instructed to by eTrust CA-ACF2 Technical Support due to the large volume of trace entries possible in the OMVS environment. It is usually easier to debug an OMVS problem using the ACFRPTOM report, because it shows more information than the trace. All of the OMVS services write SMF records when the service returns with a non-zero return code. The services can be easily found using the ERROR parameter of the ACFRPTOM report.

To stop the SECTRACE for OMVS, issue the following command, where *xxxx* is the identifier assigned to the SECTRACE:

```
ST DEL,ID=xxxx,end
```

GSO UNIXOPTS Record

The UNIXOPTS record defines the system options related to UNIX System Services.

Record ID	Fields
UNIXOPTS	CHOWNRES | NOCHOWNRES DFTGROUP(*defaultgroup*) DFTUSER(*defaultuser*) DIRACC | NODIRACC DIRSRCH | NODIRSRCH FSOBJ | NOFSOBJ FSSEC | NOFSSEC GOSETGID | NOGOSETGID HFSACL | NOHFSACL HFSSEC | NOHFSSEC IPCOBJ | NOIPCOBJ NGROUPS(NGROUP_MAX) PROCACT | NOPROCACT PROCESS | NOPROCESS

Field Descriptions

CHOWNRES | NOCHOWNRES

CHOWNRES implements POSIX CHOWN RESTRICTED which states that only the super user can modify the owner (UID) of a file. NOCHOWNRES implements POSIX CHOWN UNRESTRICTED which allows the current owner to modify the owning UID of file.

DFTGROUP(defaultgroup)

Specifies the name of the default group used by UNIX System Services (OMVS) if a user does not have a valid OMVS group in the logonid record.

DFTUSER(defaultuser)

Specifies the name of the logonid and OMVS user profile record name that defines the defaults for UNIX System Services (OMVS). If a user accesses OMVS services and does not have an OMVS user profile record, the defaults defined in this ID are used. If a user has NO-OMVS defines in his logonid, the user cannot use OMVS services and the default is not used. This is equivalent to specifying NOUID in RACF.

DIRACC|NODIRACC
Specifies if SMF records are to be cut for UNIX system services that control directory searches. Some of the functions that search directories are chmod, chown, chaudit, getcwd, link, mkdir, open, opendir, stat, ttyname and vlink. The Security Server callable service that controls cutting this SMF record is ck_access. Be aware that auditing directory searches will generate an extremely large amount of SMF records in a short period of time.

FSOBJ|NOFSOBJ
Specifies if SMF records are to be cut for UNIX system services that control the auditing of the creation and deletion of system objects. It also cuts SMF records for all access check except directory searches. Some of the functions that will do this are chdir, link, mkdir, open, mount, rename, rmdir, symlink, vmakedir, and vcreate. The Security Server callable services that control cutting of this SMF record are ck_access, ck_owner_2_files, makeFSP, make_root_FSP, makeISP, and R_audit.

FSSEC|NOFSSEC
Specifies if SMF records are to be cut for UNIX system services that control the auditing of changes to the security data (FSP) for file system objects. Some of the functions that modify the FSP are chaudit, chmod, chown, chattr, write, fachaudit, and fchmod. The Security Server callable services that control cutting of this SMF record are R_chaudit, R_chown, R_chmod, and clear_setid.

GOSETGID|NOGOSETGID
This option alters the way the makeFSP SAF callable service works. If GOSETGID (Group Owner SETGID) is set and a new directory is being created, the new directory will inherit the S_ISGID setting from the parent directory. Otherwise, the bit is set to zero. When a file or directory is being created, the owning GID of the new file is normally set to that of the parent directory. If GOSETGID is set and the parent's set-gid bit is off, then the owning GID of the new file or directory is set to the effective GID of the process.

HFSACL|NOHFSACL
When HFSACL is specified, Access Control Lists are used in the z/OS UNIX security access validation process in addition to the checking of file permission bits and superuser status. When NOHFSACL is specified, normal z/OS UNIX security access validation is done, including the checking of file permission bits and superuser status. Access Control Lists (ACL's) are supported in z/OS release 1.3 and above. If HFSSEC is also specified, ACL's are not used regardless of the setting of this field. See "Controlling Access to the Hierarchical File System" for more information about Access Control Lists.

HFSSEC|NOHFSSEC

When HFSSEC is specified, CA SAF HFS security is activated. Normal z/OS UNIX security access validation is bypassed. This includes checking of file permission bits, superuser status and normal z/OS UNIX Security Services.

When NOHFSSEC is specified, CA SAF HFS security is not active. Normal z/OS UNIX sercurity access validation is enabled. This includes checking of file permission bits, superuser status and normal z/OS UNIX Security Services. NOHFSSEC is the default.

See Controlling Access to the Hierarchical File System for more information on CA SAF HFS security.

CA SAF HFS security can also be controlled using the F ACF2,HFS (STATUS| ENABLE|DISABLE) command. See the *Systems Programmer's Guide* for more information.

IPCOBJ|NOIPCOBJ

Specifies if SMF records are to be cut for UNIX system services that control the auditing of the access control, IPC object changes and the creation and deletion of IPC objects. Some of the functions that will do this are msgctl, msggest, msgsnd, semctl, semget, semop, shmat, shmget and shmctl. The Security Server callable services that control cutting of this SMF record are ck_IPC_access, R_PIC_ctl, and makeISP.

NGROUPS(NGROUPS_MAX)

The maximum size of the supplemental group list created for each signon. Supplemental groups are currently used with UNIX System Services support. The value for NGROUPS is a number in the range of 0 to 8192. The default is 300.

PROCACT|NOPROCACT

Specifies if SMF records are to be cut for UNIX system services that control the auditing of services that look at data from or effect other processes. Some of the functions that affect other processes are getpsent, kill, ptrace, recv, recvmsg and sendmsg. The Security Server callable services that control cutting of this SMF record are ck_process_owner and R_ptrace.

PROCESS|NOPROCESS

Specifies if SMF records are to be cut for UNIX system services that control the dubbing and undubbing of processes, changes to the UIDs and GIDs of processes, and changes to the thread limits and other privileged options. Some of the functions that dub processes or change process values are exec, setuid, setgid, seteuid, setegid, dub, undub, and vregister. The Security Server callable services that control cutting of this SMF record are R_exec, R_setuid, R_setgid, R_seteuid, R_setegid, ck_priv, initUSP, initACEE, and deleteUSP.

Displaying the UNIXOPTS Record

SHOW UNIXOPTS

Displays UNIX options on the system.

```
show unixopts

-- UNIXOPT OPENEDITION/MVS/UNIX/SYSTEM SERVICES (USS) SUMMARY --
OMVS DEFAULT USER:  OMVSDFLT
OMVS DEFAULT GROUP:  OMVSUSRG
MAX NUMBER OF OMVS GROUPS: 300
HFS SECURITY ACTIVE: YES
HFSACL ACTIVE: YES
FILE.GROUPOWNER.SETGID ACTIVE:  NO

-- AUDIT FLAG STATUS --
CHOWN _RESTRICTED:  NO
DIRACC_ACTIVE:      NO
DIRSRCH_ACTIVE:     NO
FSOBJ_ACTIVE:       NO
FSSEC_ACTIVE:       NO
IPCOBJ_ACTIVE:      NO
PROCACT_ACTIVE:     NO
PROCESS_ACTIVE:     NO
```

z/OS Unix System Services Security Calls

To monitor user activity in a z/OS Unix System Services environment, eTrust CA-ACF2 logs security events under z/OS Unix System Services to SMF using the standard eTrust CA-ACF2 SMF record. Log records are written for any security event that denies the user access to a z/OS Unix System Services facility. These records can assist you in determining the UID and GID of the user involved in the attempted access.

Setting Attributes

Turning on the user or auditor logging or audit options in an HFS file can also cause logging. The owner of the file can set the user audit attribute of the file. The auditor option can also set an audit attribute, but requires the eTrust CA-ACF2 AUDIT or SECURITY privilege. Each of these attributes is set based on the access being attempted to the file. If these AUDIT attributes or flags are turned on in a file for the type of file access, eTrust CA-ACF2 logs that access by writing an SMF record. These SMF records can be viewed on the ACFRPTOM report.

The ACFRPTOM Report

The eTrust CA-ACF2 report ACFRPTOM uses standard eTrust CA-ACF2 report JCL like the following for batch submission:

```
//ACFRPTOM JOB 1,'USS RPT',MSGCLASS=A,TYPRUN=HOLD
//*
//REPORT  EXEC PGM=ACFRPTOM,PARM='TITLE(USS EVENTS)'
//*
//SYSPRINT DD  SYSOUT=*
//SYSUDUMP DD  SYSOUT=*
//RECMAN1  DD  DSN=SYS1.MAN1,DISP=SHR
//SYSIN    DD  DUMMY
//
```

See the *Reports and Utilities Guide* for more information about the ACFRPTOM report.

Implementing z/OS Unix System Services Applications in an eTrust CA-ACF2 Environment

The remainder of this chapter presents information regarding the implementation of various applications that run under z/OS Unix System Services. The details show what is necessary to set up these applications in an eTrust CA-ACF2 protected environment. Additional information regarding the administrative commands used in this chapter can also be found in other sections of the *Administrator Guide*.

The required set up usually follows three basic steps:

1. Define the required logonids and associated profile information.
2. Define any required group profiles.
3. Allow access to any of the protected resources used by the application.

Any additional information or set up requirements are noted in the respective section following the basic set up steps. Each section below covers a separate application.

TCP/IP

TCP/IP (Transmission Control Protocol/Internet Protocol) is a telecommunications protocol commonly used for communicating among distributed systems. TCP/IP allows computers to talk to each other and is well established on the UNIX and PC platforms.

Communications Server IP for z/OS (TCP/IP)

If you are running Communications Server IP for z/OS, appropriate OMVS authorities are required for the logonids under which the TCP/IP servers run. Create the TCP/IP logonid and the needed OMVS credentials by entering the following commands:

```
SET LID
INSERT TCPIP NAME(TCP/IP STC LOGONID) STC MUSASS GROUP(OMVSGRP) UID(0) HOME(/)
SET PROFILE(GROUP) DIV(OMVS)
INSERT OMVSGRP GID(nn)
```

Note: Reports have come in indicating that use of the lpr command has required that the user have superuser authority. This has been traced to the RESTRICTLOWPORTS setting in TCP/IP being set to YES. This forces the lpr command to use a port below 1024, requiring superuser authority. Set this option to NO to remove the superuser requirement.

TCP/IP SERVAUTH Class

TCP/IP uses the SERVAUTH class to protect various TCP/IP resources from unauthorized access. There are five functions protected under the SERVAUTH class:

Stack Access
Controls which users can access TCP/IP stack using a constructed resource name similar to EZB.STACKACCESS.sysname.tcpipid.

Net Access
Controls which users can access individual networks using a constructed resource name similar to EZB.NETACCESS.sysname.tcpipid.netname.

Port Access
Controls which users can use TCP and UPD ports using a constructed resource name similar to EZB.PORTACCESS.sysname.tcpipid.portname.

Netstat Access
Controls access to Netstat command output from TSO and z/OS Unix System Services shell environments using a constructed resource name similar to EZB.NETSTAT.sysname.tcpname.netstatoption.

TN3270

Controls which users can use secured ports using a constructed resource name similar to EZB.TN3270.sysname.tcpipid.PORTnnnn.

where sysname is the name of the system:

- tcpipid is the name of the TCPI/IP started task.
- netname is the network named defined in the PROFILE.TCPIP file.
- portname is the port name defined in the PROFILE.TCPIP file
- nnnn is the port number with leading zeros.

STACK ACCESS resources are validated automatically while the other three resource types require activation via settings in the PROFILE.TCPIP file.

See the *z/OS IP Configuration Guide* for additional information.

Users attempting to use TCP/IP via a protected access request that they have read access to the specific resource being validated. To allow access to any or all of these resources, create and store a resource rule with the appropriate rule entries.

For example, the following rule would allow USERA access to all the SERVAUTH class resources:

```
SET RESOURCE(SER)
COMPILE*
$KEY(EZB) TYPE(SER)
NETACCESS.- UID(usera_uid) SERVICE(READ) ALLOW
STACKACCESS.- UID(usera_uid) SERVICE(READ) ALLOW
PORTACCESS.- UID(usera_uid) SERVICE(READ) ALLOW
TN3270.- UID(usera_uid) SERVICE(READ) ALLOW
NETSTAT.- UID(usera_uid) SERVICE(READ) ALLOW

STORE
```

The type code selected is based on the value set up for the default CLASMAP for SERVAUTH. You can change the type code by creating your own local CLASMAP records using the following commands:

```
SET CONTROL(GSO)
INSERT CLASMAP.SERVAUTH RESOURCE(SERVAUTH) RSRCTYPE(type_code) ENTITYLN(64)
```

The SERVAUTH resource rules must be resident. You set this up by defining the type code as resident in the INFODIR GSO record. The following commands are an example of doing this using the SER default type code:

```
SET CONTROL(GSO)
CHANGE INFODIR TYPES(R-RSER)
```

To activate these changes immediately, enter the following commands:

```
F ACF2,REFRESH(ALL)
F ACF2,REBUILD(SER)
```

Securing IPv4 Addresses

Securing an IPv4 address using eTrust CA-ACF2 (or any external security product) requires that the installed TCP/IP product pass the IPv4 address packet. However, not all TCP/IP vendor products pass this information. IBM's TCP/IP product does pass the IPv4 address.

IPv4 address protection is not available if your TCP/IP product does not pass the IPv4 address packet.

The IPv4 packet passed is generated from the user's originating IPv4 address. Thus, these IPv4 packets often have no resemblance to standard LU names. Each node of the IPv4 address is translated into a character representation of the hex value of the node. For example, the IPv4 address 141.202.201.56 would appear as terminal 8DCAC938. The hex value of 141 is 8D, the hex value of 202 is CA, and so forth.

eTrust CA-ACF2 provides two mechanisms to implement security of an IPv4 address. The standard IPv4 address is converted to hex pairs. If you want to restrict a particular user to enter the system only through a specific IPv4 address, use source restriction on the logonid. For example:

```
SET LID
CHANGE usera SOURCE(8DCAC938) equivalent to 141.202.201.56
```

If you want to allow more than one IPv4 address or an IPv4 address range, use a source group restriction. Add each allowable source or use masking. For example, to allow the use of all IPv4 addresses starting with 141.202 and 141.201.201.55, enter:

```
SET XREF(SGP)
INSERT IPGROUP INCLUDE(8DCA**** 8DC9C937)
SET LID
CHANGE usera SOURCE(IPGROUP)
```

To activate changes to the XREF records immediately, issue:

```
F ACF2,NEWXREF,TYPE(SGP)
```

Important! The handling of IPv6 addresses affects the current handling of IPv4 addresses that are passed to eTrust CA- ACF2 as converted hexadecimal pairs. You cannot specify an IPv6 address as a physical input source in logonids, access rules, and resource rules, as you can with an IPv4 address, since, in z/OS V1R5 the TERMID is no longer used as the source ID for users trying to gain entry to the system and resources through IPv6-based ports of entry. Therefore, if you are implementing IPv6 and IPv4 addressing in your system, to ensure that source restrictions continue to function properly in existing logonids, access rules, and resource rules, which specify an IPv4 physical input source in one or more X(SGP) records, you must add the word SERVAUTH to these records. Logonid records that specify a physical input source in the SOURCE field must also be changed to specify the name of a source group record in which the physical input sources and the word SERVAUTH have been defined.

Example

In a system both IPv4 and IPv6 addressing, the word SERVAUTH must be added to existing X(SGP) record, IPV4GTSA:

```
 ACF
set x(sgp)
 XREF
change ipv4gtsa include(servauth) add
  XE43 / IPV4GTSA LAST CHANGED BY M1ADMN 06/29/05-16:33
                      INCLUDE(A69LO907 SERVAUTH) SOURCE
```

The X(SGP) record IPV4GTSA, is specified as the SOURCE in the logonid, IPUSR05. Therefore, User IPUSR05 will be allowed to access the system through IPv4 terminal, A69LO907.

```
 ACF
SET LID
 LID
l ipusr05
  IPUSR05         SHS  IPUSR05  TEST USER ADMIN    X1111
                       COMPANY(S) DEPT() IDNUM() LEVEL(S) LOCATION() OLDLID()
                       POSITION() PROJECT() SITE(H)
  ...

  RESTRICTIONS         PREFIX(IPUSR05) SOURCE(IPV4GTSA)
 LID
```

In addition, the X(SGP) record IPV4GTSA, is specified as the SOURCE in the resource rule entry, TEST1.TEST.RESOURCE.NAME. Therefore, user IPUSR16 is allowed READ or UPDATE access to the resource, TEST1.TEST.RESOURCE.NAME, in the FACILITY class, through IPv4 terminal A69LO907.

```
 ACF
SET RESOURCE(FAC)
 RESOURCE
DECOMP TEST1
 ACF75052 RESOURCE RULE TEST1 STORED BY M1ADMN 06/29/05-16:38
 $KEY(TEST1) TYPE(FAC)
  TEST.RESOURCE.NAME UID(-IPUSR16) SOURCE(IPV4GTSA) SERVICE(READ,UPDATE) ALLOW
ACF75051 TOTAL RECORD LENGTH= 278 BYTES, 6 PERCENT UTILIZED
 RESOURCE
```

Securing IPv6 Addresses

IPv6 is a new version of IP designed as a successor to IPv4. Because IPv6 increases the IP address size from 32 bits to 128 bits, a physical IPv6 address cannot be translated into a hexadecimal-paired format and must be handled differently than an IPv4 address.

The TERMID on a RACROUTE REQUEST=VERIFY call will no longer be used as the source ID when an IPv6-based port of entry is trying to gain access to a system, resource or dataset. Instead, IBM has created a new SESSION type, IP, and a new port-of-entry (POE) class, SERVAUTH. The new SERVAUTH keyword on the VERIFY RACROUTE call specifies the address of the identifier of the server through which a user is trying to gain access to the system. The address points to a 1-byte length field followed by a 64-byte data area, which contains the name of a resource in the SERVAUTH class. This resource name is the network access security zone name that contains the IPv6 address of the user. Security zone mappings are defined in the NETACCESS parameter block in a TCP/IP profile.

The following is an example of a TCP/IP Profile:

```
NETACCESS
    9.24.104.0/24       ZONE1
    9.24.104.119/32     ZONE2
ENDNETACCESS
```

The network access zone name to which the above IP addresses are mapped is in the following format:

```
EZB.NETACCESS.sysname.stackname.zone
```

Allowing system entry and resource access from an IPv6 address using eTrust CA-ACF2 requires, first, creating a resource rule for the network security zone in which the user's IPv6 address is mapped.

The following eTrust CA-ACF2 resource rule would allow USERA to access the system from IPv6 address 9.24.104.119/32 in ZONE2.

```
set resource(ser)
 RESOURCE
compile*
  $key(ezb) type(ser)
   netaccess.-.zone2 uid(usera) service(read) allow
store
```

In addition to creating resource rules, you must also create and activate the following SAFDEF record to process SAF AUTH requests coming from ACF9CSFV.

```
acf
 ACF
set control(gso)
 CONTROL
Insert safdef.acf9csfv id(acf9csfv) mode(global) racroute(reqstor=acf9csfv
 request=auth)
XE69 / SAFDEF.ACF9CSFV LAST CHANGED BY M1ADMN ON 08/01/04-11:14
                       FUNCRET(4) FUNCRSN(0) ID(ACF9CSFV) MODE(GLOBAL)
                       RACROUTE(REQSTOR=ACF9CSFV REQUEST=AUTH RETCODE(4)
 CONTROL
f acf2,refresh(safdef)
 ACF79507 GSO PROCESSING COMPLETED WITHOUT ERROR
 CONTROL
```

WARNING! *If the above SAFDEF record is not created and activated on the eTrust CA-ACF2 system with or without MLS active, the system will not be secured from IPv6-based ports of entry.*

In addition to resource rule validation on the SERVAUTH class, MLS can also be used to further restrict user entry to the system and access from an IPv6 address. An administrator must create an appropriate eTrust CA-ACF2 Security Seclabel Compiled Profile record that classifies the network access zone name and the IPv6 address mapped to it and assigns it a security label. Once this is done, and MLS is active on the system, security label checking will be performed at system entry, to ensure that the user's security label is equivalent to the security label of the SERVAUTH resource. If it is not, the user will be denied access to the system through the server. If the security label check succeeds, rule validation is then performed to ultimately allow or deny the access request.

The following eTrust CA-ACF2 SECLABEL Compiled Profile record would allow USERA to access the system from IPv6 address 9.24.104.119/32 that is in ZONE2 only if the security label with which USERA enters the system is equivalent to LABEL2 (the security label of the network security ZONE2) and resource rule validation also allows the access.

```
set p(seclabel) div(profile)
 PROFILE
compile *
  $key(ezb)
  $rtype(ser)
   netaccess.-.zone2  seclabel(label2)
store
```

For more information about how to secure SERVAUTH resources in an eTrust CA-ACF2 MLS system, see the *Multilevel Security Planning Guide*.

z/OS FTP

Packaged with TCP/IP OE Application Services, FTP is an OMVS application that executes under z/OS Unix System Services to facilitate the file transfer of OMVS HFS files, as well as MVS data sets, throughout a TCP/IP network. FTP under z/OS Unix System Services typically executes under a started task named FTPD.

To complement FTP, TCP/IP OE Application Services includes an optional message-log daemon (called SYSLOG-D) that can be used to log past and present message traffic related to FTP. This optional logging task should be considered since, without it, there is no ongoing log of FTP activity.

Installing FTP with eTrust CA-ACF2

Per IBM documentation, the installation of FTP has considerations for the mainframe security administrator. The following information describes the deviations from and adjustments to IBM documentation when installing FTP with eTrust CA-ACF2:

1. Create a logonid for the FTP started task typically named FTPD.

   The FTPD logonid must be assigned superuser status, UID(0).

   ```
   SET LID
   INSERT FTPD NAME(OE/FTP STCID) STC GROUP(OMVSGRP) UID(0) HOME(/)
   OMVSPGM(/bin/sh)
   ```

2. Give the FTPD logonid access to the required BPX FACILITY class resources.

   The FTPD logonid needs access to the BPX.DAEMON resource. To accomplish this, add the following rule line to the existing BPX FACILITY resource rule:

   ```
   DAEMON UID(ftpd) SERVICE(READ) ALLOW
   ```

HFS Security Considerations

FTP uses a SERVAUTH class resource to protect HFS access by FTP users. Any FTP users requiring access to HFS must be allowed to this resource. The resource name uses the following format:

```
EZB.FTP.sysname.ftpdaemonname.ACCESS.HFS
```

where sysname is the system FTP is running under and ftpdaemonname is the name of the FTP daemon. For example, if FTP is running on a system named SYS1 and the FTP started task is FTPD, the resource name would be constructed as follows:

```
EZP.FTP.SYS1.FTPD.ACCESS.HFS
```

In our example, based on the previous discussions of the SERVAUTH, we would add the following rule entry to the EZP rule as appropriate:

```
FTP.SYS1.FTPD,ACCESS.HFS UID(FTP_user) ALLOW
```

To activate any changes to the SERVAUTH class EZB rule, issue the following command:

```
F ACF2,REBUILD(SER)
```

ANONYMOUS Logon Feature

Users accessing FTP must log on to the service before using it. This requires them to supply their userids and passwords, and for their userids to have access to TCP/IP.

FTP supports anonymous logons if the ANONYMOUS parameter is set in the FTPDATA configuration file in one of the following ways:

```
ANONYMOUS
ANONYMOUS userid
ANONYMOUS userid/password
```

If the ANONYMOUS parameter is specified without a userid and an FTP user specifies anonymous at logon time, a SAF VERIFY is issued for the logonid ANONYMOU. If this logonid is defined to eTrust CA-ACF2, the user is allowed to log on and run under the authority of the ANONYMOU logonid.

If the ANONYMOUS parameter is specified with a userid, the user is prompted for a password and must provide the correct password for this userid. If the ANONYMOUS parameter is specified with both userid and password, both values are used to sign on.

The use of ANONYMOUS logon should be carefully considered. It can be a candidate for auditing.

TELNET

TELNET is a feature of TCP/IP that allows users terminal access to a system over a TCP/IP network. TELNET runs under both native MVS and z/OS Unix System Services. If running under native MVS, users can be forced to log on to TELNET itself. This will occur if the TELNET configuration statements in the TCP/IP profile data set do not specify a DFLTAPPL. Alternatively, DFLTAPPL can be specified, which directs all logons to a session manager such as CA-TPX. The session manager then controls access through its security interface.

Securing TELNET for z/OS Unix System Services

When using TELNET under OMVS, RLOGIN runs under its own logonid until the user successfully signs on. The name of this logonid is specified in the configuration file /etc/inetd.conf. It is delivered with a default ID of OMVSKERN. Consider changing this default to one that conforms to installation standards and is not common knowledge. This userid must be defined with both superuser and daemon authority.

The following commands define such a logonid:

```
SET LID
INSERT omvskern NAME(TELNET OMVS ID) STC GROUP(OMVSGRP) UID(0) HOME(/)
OMVSPGM(/bin/sh)
```

If you have not already defined the OMVSGRP group to OMVS, issue the following commands using an available group number:

```
SET PROFILE(GROUP) DIV(OMVS)
INSERT OMVSGRP GID(nn)
```

If you are using OMVSKERN, it is likely that this logonid was defined as part of your OMVS installation. In addition, if you are securing daemon authority, the TELNET server logonid must have access to the FACILITY class resource BPX.DAEMON resource. Add the following rule entry to the BPX FACILITY resource rule:

```
DAEMON UID(omvskern) SERVICE(READ) ALLOW
```

z/OS Infoprint Server

The z/OS Infoprint Server allows for consolidation of print files from multiple servers to one central server. Control of the resources defined under the Print Server requires the definition of two groups: AOPOPER and AOPADMIN. These are the IBM defaults. AOPOPER is the operator group with authority to start and stop the print interface. AOPADMIN provides authority to administer the printer inventory and controls. If the separation of authority is not necessary, then one group name can be defined for both functions.

Use the following steps to define the security environment for eTrust CA-ACF2:

1. Define logonids for the started tasks used by the Infoprint server using the following commands:

```
SET LID
INSERT AOPSTART NAME(AOPSART PROC) STC
INSERT AOPSTOP NAME(AOPSTOP PROC) STC
```

2. Define group profile records using the following example:

```
SET PROFILE(GROUP) DIV(OMVS)
INSERT AOPOPER GID(nn)
INSERT AOPADMIN GID(nn)
```

   Replace *nn* with an appropriate GID number.

3. The Infoprint Server limits access to the printer configuration using the FACILITY class. To let users access this resource to perform administrative functions, create the following resource rule under the FACILITY type:

```
$KEY(AOPADMIN) TYPE(FAC)
 UID(uid) ALLOW
```

4. To let users access the groups through the supplemental group facility, write TGR resource rules that grant access to the specified groups as appropriate.

```
$KEY(AOPOPER) TYPE(TGR)
 UID(uid) ALLOW

$KEY(AOPADMIN) TYPE(TGR)
 UID(uid) ALLOW)
```

5. The FAC type and TGR type rules should be maintained in a directory so, be sure to issue the rebuild command after compiling and storing the rules:

```
F ACF2,REBUILD(FAC),CLASS(R)
F ACF2,REBUILD(TGR),CLASS(R)
```

6. To correctly perform security checking in the PostScript and PDF to AFP transform requires an ownership change to its executable ps2afpd. As installed, ps2afpd has an owner with a UID of 0. Ownership of this file must be changed to a UID that is not a superuser. The z/OS Infoprint Server Customization Guide details the steps necessary to accomplish this.

   You can use an existing UID and GID or create one specifically for this purpose using the following commands:

```
SET LID
INSERT lid_name NAME(PS2AFPD owner) GROUP(group_name) UID(nn)
SET PROFILE(GROUP) DIV(OMVS)
INSERT group_name GID(nn)
END
```

   Replace nn in the UID and GID operands with an appropriate UID and GID values. To immediately activate these changes, issue the following commands:

```
F ACF2,REBUILD(USR),CLASS(P)
F ACF2,REBUILD(GRP),CLASS(P)
F ACF2,OMVS
```

DOMINO for z/OS (Lotus Notes) Server

DOMINO for z/OS, revered to as The Lotus Notes (Email), runs in a z/OS environment. While DOMINO has its own internal security, which does not interface with external security, there are a number of tasks that must be done to define DOMINO to external security.

1. Define a logonid for the DOMINO with a valid z/OS Unix System Services UID and GID. The z/OS Unix System Services UID should not be UID(0) as suggested in the DOMINO documentation. The following example shows sample commands to accomplish this:

```
ACF
SET LID
INSERT domino NAME(DOMINO server) STC GROUP(notes) UID(nn) OMVSPGM(/bin/sh)
HOME(/u/domino)
SET PROFILE(GROUP) DIV(OMVS)
INSERT notes GID(nn)
```

   Note: The values used are based on the sample defaults in the DOMINO documentation. Be sure to use the appropriate values for your installation.

2. The DOMINO server might need access to various FACILITY resources. If the DOMINO server is going to create SMF records, it needs READ access to the resource of BPX.SMF. If you are going to make the DOMINO server non-swappable, DOMINO needs READ access to the BPX.STOR.SWAP resources. If the server is going to spawn jobs with a name not matching its id, then it needs READ access to the resource BPX.JOBNAME. Add the following rule entries as appropriate to the BPX FACILITY resource rule:

```
SMF UID(domino) SERVICE(READ) ALLOW
STOR.SWAP UID(domino) SERVICE(READ) ALLOW
JOBNAME UID(domino) SERVICE(READ) ALLOW
```

z/OS Console for Domino

DOMINO for z/OS provides a facility (DOMINO console interface) for sending commands from z/OS to start, stop, and manage a DOMINO server running under UNIX System Services. The code provided to implement this support is named DOMCON. To define the environment to eTrust CA-ACF2, do the following:

1. Create a logonid and profile record for DOMINO console user:

```
SET LID
INSERT DOMCNSL NAME(COMINO console user) GROUP(NOTES) UID(0) OMVSPGM(/bin/sh)
HOME(/u/domcnsl)
```

2. The DOMINO console logonid needs access to the FACILITY class resources of BPX. DAEMON and BPX.JOBNAME. Add the following rule entries to the BFX FACILITY rule:

```
DAEMON UID(domcnsl_uid) SERVICE(READ) ALLOW
JOBNAME UID(domcnsl_uid) SERVICE(READ) ALLOW
```

3. Users that are authorized to enter domoe commands will need update access to the FACILITY class resource BPX.SERVER. Add the following rule entry as necessary for the users to the BFX FACILITY rule:

```
SERVER UID(domoe_cmd_user_uid) SERVICE(READ,UPDATE) ALLOW
```

4. These same users also need access to the SURROGAT class resources for each server. The default server name of DOMINO would result in a SURROGAT resource name of BPX.SRV.DOMINO. This would be accomplished using commands similar to the following:

```
SET RESOURCE(SUR)
COMPILE
$KEY(BPX) TYPE(SUR)
SRV.DOMINO UID(domino_cmd_user_uid) SERVICE(READ) ALLOW

STORE
```

Note: If the BPX SURROGAT rule already exists at your site, simply add the appropriate rule entry to it.

5. If the SURROGAT or FACILITY rules were made resident via the GSO INFODIR record, issue the following commands to activate the changes:

```
F ACF2,REBUILD(SUR)
F ACF2,REBUILD(FAC)
```

z/OS and OS/ 390 Console for Domino 5.0.6 and Higher

With DOMINO for z/OS servers release 5.0.6 and higher the set up process has changed. The command process has split into multiple procedures as follows:

- DOMINC – Issues command to the Domino servers.
- DOMINK – Stops the Domino servers.
- DOMINM – Enables and disables the monitor.
- DOMINS – Starts the Domino servers.

To define this environment to eTrust CA-ACF2, do the following:

1. Create logonid and profile records for DOMINO procedures:

```
SET LID
INSERT DOMINC NAME(DOMINC procedure) GROUP(NOTES) UID(0) OMVSPGM(/bin/sh)
HOME(/u/domcnsl)

INSERT DOMINK NAME(DOMINK procedure) GROUP(NOTES) UID(0) OMVSPGM(/bin/sh)
HOME(/u/domcnsl)

INSERT DOMINM NAME(DOMINM procedure) GROUP(NOTES) UID(0) OMVSPGM(/bin/sh)
HOME(/u/domcnsl)

INSERT DOMINS NAME(DOMINS procedure) GROUP(NOTES) UID(0) OMVSPGM(/bin/sh)
HOME(/u/domcnsl)
```

2. The DOMINO console logonids need access to the FACILITY class resource of BPX. DAEMON. Add the following rule entry to the BFX FACILITY resource rule for each procedure:

```
DAEMON UID(procedure_uid) SERVICE(READ) ALLOW
```

3. The logonid used for running the install script for the console support must be UID 0 and have read access to the facility class resources BPX.FILEATTR.PROGCTL and BPX.FILEATTR.APF. This is required to set the extended attributes for console support to APF authorized and program controlled. Add the following rule entry to the BFX FACILITY resource rule:

```
FILEATTR.PROGCTL UID(installing_lid's_uid) SERVICE(READ) ALLOW

FILEATTR.APF UID(installing_lid's_uid) SERVICE(READ) ALLOW
```

4. Users that are authorized to enter domoe commands will need update access to the FACILITY class resource BPX.SERVER. Add the following rule entry as necessary for the users to the BFX FACILITY rule:

```
SERVER UID(domoe_cmd_user_uid) SERVICE(READ,UPDATE) ALLOW
```

5. These same users also need access to the SURROGAT class resources for each server that a command is sent to. The default server name of DOMINO would result in a SURROGAT resource name of BPX.SRV.DOMINO. This would be accomplished using commands similar to the following:

```
SET RESOURCE(SUR)
COMPILE
$KEY(BPX) TYPE(SUR)
SRV.DOMINO UID(domino_cmd_user_uid) SERVICE(READ) ALLOW

STORE
```

   Note: If the BPX SURROGAT rule already exists at your site, simply add the appropriate rule entry to it.

6. A server that is started with the z/OS console support use the _BPX_JOBNAME environment variable to assign an unique job name for itself and each task associated with the server. To authorize this, the server logonid must be permitted to the facility class resource BPX.JOBNAME. Add the following rule entry to the BFX FACILITY resource rule for each server logonid:

```
JOBNAME UID(server_lid_uid) SERVICE(READ) ALLOW
```

7. If the SURROGAT or FAICLITY rules were made resident via the GSO INFODIR record, issue the following commands to activate the changes:

```
F ACF2,REBUILD(SUR)
F ACF2,REBUILD(FAC)
```

Defining Lotus Notes Users

Each user of the Lotus Notes Server must be defined to eTrust CA-ACF2 using an lnotes profile record. This profile record connects the user's logonid with the user definition name used within Lotus Notes.

To create lnotes profile records for Lotus Notes users, use the following commands:

```
SET PROFILE(USER) DIV(LNOTES)
INSERT logonid SNAME(lotus_notes_user_name)
```

After creating the lnotes profile record, refresh the user profile records as described in Operator Commands for z/OS Unix System Services, Rebuilding Directories, earlier in this chapter.

z/OS Component Broker Series (SOMobjects) Support

eTrust CA-ACF2 supports all functionality for the changes in SAF calls to handle z/OS Component Broker Series (SOMobjects). The changes include parameters in the RACROUTE EXTRACT calls as well as modifications to the InitACEE callable service.

There is also a security server provided that protects the resources used in SOMobjects. These resources include access to a particular SOM server, access to SOM objects, and access to SOM object methods. The security process uses three classes (CBIND, SERVER, and SOMDOBJ) to validate access to the resources used with SOMobjects. These three classes have been predefined to eTrust CA-ACF2 with an assigned resource type code of SAF.

To set up SOMobjects with eTrust CA-ACF2, perform the following steps:

1. Ensure that you are running with STC set in the GSO OPTS record.
2. Create the logonids and profiles for the started tasks needed by SOMobjects, including the SOM server, the naming and security server, any object server, and any application servers. The following example command shows how to create started task logonids:

   ```
   SET LID
   INSERT SOM NAME(SOM started task) GROUP(somgroup) STC UID(nn)
   HOME(home-path-name) OMVSPGM(/bin/sh

   INSERT server NAME(SOM server) GROUP(somgroup) STC UID(nn)
   HOME(home-path-name) OMVSPGM(/bin/sh

   INSERT appserver NAME(Application server) GROUP(somgroup) STC UID(nn)
   HOME(home-path-name) OMVSPGM(/bin/sh
   ```

   Where *server* is the name for the naming, security, or object server, and *appserver* is the name of the application server.

   Do the same for any other logonids and assign a unique UID number to each user.
3. Allow the started task logonids created in step one access to the SOMMVS data sets.

   ```
   SET RULE
   COMPILE *
   $KEY(SOMMVS)
   ```
4. Define group profiles used for the SOM subsystem, the associated object servers, and SOM object clients to z/OS Unix System Services by creating the required user and group profile records:

   ```
   SET PROFILE(GROUP) DIV(OMVS)
   INSERT somgroup GID(nn)
   ```

   Assign a unique GID number to each group.

5. When a SOMobjects server is started, SOMobjects determines whether that server is allowed to communicate with the SOM subsystem. The resource used in the validation is SOM.subsystem-name.server-alias-name under the SERVER class. Allow the SOMobjects server subsystem access with the following rule:

```
SET RESOURCE(SAF)
COMPILE *
$KEY(SOM) TYPE(SAF)
 subsytem-name.server-alias-name UID(somobject server) SERVICE(READ) ALLOW
```

6. Servers will run in a secure mode if this is activated using the REGIMPL utility. For information on the REGIMPL utility, see IBM *z/OS SOMobjects Configuration and Administration*. If a SOMobjects client attempts to access a secured server, various levels of validation are performed. The first check determines whether the client is allowed access to the server. A resource check is made against the SOM.subsytem-name.server-alias-name in the CBIND class.

   The following rule allows this access:

```
SET RESOURCE(SAF)
COMPILE *
$KEY(SOM) TYPE(SAF)
 subsytem-name.server-alias-name UID(client) SERVICE(READ) ALLOW
```

   If SAF validation is active for the APPL class, the following rule is also required:

```
SET RESOURCE(SAF)
COMPILE *
$KEY(SOM) TYPE(SAF)
 UID(client) ALLOW
```

   When a client accesses a SOMobject method, a validation is performed against the SOMDOBJ class with a resource name of classname.methodname. The following rule is required to allow access to the SOMDOBJ resource:

```
SET RESOURCE(SAF)
COMPILE *
$KEY(classname) TYPE(SAF)
 methodname UID(client) ALLOW
```

   If the classname is larger than 40 characters, write the rule as follows:

```
$KEY(classname) TYPE(SAF)
 'remainder-classname.methodname' UID(client) ALLOW
```

   Where the *classname* in the $KEY is the first 40 character of the classname. The remainder of the classname is on the rule line.

   If desired, you can change the type code assigned to the various SAF resource classes by defining a CLASMAP record for each class. For example:

```
SET CONTROL(GSO)
INSERT CLASMAP.CBIND RESOURCE(CBIND)RSRCTYPE(CBD) ENTITYLN(50)
INSERT CLASMAP.SERVER RESOURCE(SERVER) RSRCTYPE(SRV) ENTITYLN(50)
INSERT CLASMAP.SOMDOBJ RESOURCE(SOMDOBJ) RSRCTYPE(SOM)- ENTITYLN(246)
```

z/OS Security Server

As a separate offering, along with z/OS, IBM provides the Security Server. This offering is a bundling of RACF with a number of other products. All of these products perform some security function. Those that interface with RACF do so through standard security calls supported by eTrust CA-ACF2. None are truly dependent on RACF to function.

RACF

Although delivered as part of the security server, RACF must be independently activated and is not required to run the other Security Server components. RACF is IBM's SAF compliant security system. eTrust CA-ACF2 is fully SAF compliant and will process all appropriate SAF calls so the RACF component of the Security Server is not needed in an eTrust CA-ACF2 environment.

Disabling RACF

The level of security that you run with depends on your site's specific needs. In an eTrust CA-ACF2 environment, the RACF component must be disabled. To do this in RACF, update the appropriate IFAPRD*xx* member and change the STATE field as to:

```
STATE(DISABLED)
```

Then re-IPL the system to make the change take effect.

For example, if you want to disable the RACF component of the security server but continue to use the DCE component of the security server, update the IFAPRD*xx* entries to look like this:

```
PRODUCT OWNER('IBM CORP')
        NAME('z/OS')
        FEATURENAME('Security Server')
        ID(5647-A01)
        VERSION(*)
        RELEASE(*)
        MOD(*)
        STATE(ENABLED)

PRODUCT OWNER('IBM CORP')
        NAME('z/OS')
        FEATURENAME('RACF')
        ID(5647-A01)
        VERSION(*)
        RELEASE(*)
        MOD(*)
        STATE(DISABLED)
```

Or, if you want to disable the security server completely and run without security, update the IFAPRD*xx* entry to look like this:

```
PRODUCT OWNER('IBM CORP')
        NAME('z/OS')
        FEATURENAME('Security Server')
        ID(5647-A01)
        VERSION(*)
        RELEASE(*)
        MOD(*)
        STATE(DISABLED)
```

DCE Security Server

The DCE Security Server is a separate security product that provides authentication services for users and servers running DCE applications. It provides an independent or third-party platform in which to validate and authenticate security requests.

In a DCE environment, one DCE Security Server must exist. It provides a central hub for system entry validation and single-sign on authentication for all DCE platforms within a connected DCE environment. When a user passes from one DCE platform to another, the target platform passes information about the user credentials, along with other information, to the DCE Security Server for authentication and authorization. The DCE Security Server authenticates such requests by checking the supplied user credentials against those stored in the DCE Security Server's security repository or security registry.

When performing this authentication, the DCE Security Server follows an authentication algorithm, which involves not only the user credentials but also encryption keys known for each platform. The algorithm is standards-based and is platform-independent. As a result, multiple vendors and platforms offer a DCE Security Server.

The z/OS DCE Security Server performs its authentication following DCE conventions, not the conventions of RACF or any other mainframe external security manager (ESM). With z/OS DCE Security Server, the ESM is used mainly as a repository for information needed to support the DCE authentication process. In particular, the ESM is used to hold DCE-specific userid information and encryption keys. A DCE segment can be defined for any userid. The DCE userid segment allows six DCE-specific fields to be maintained for any userid. For example, one field named DCEKEY identifies each user's DCE password. During authentication, the DCE Security Server retrieves this and other information from the ESM using standard SAF interfaces. This information is then used separately to authenticate and authorize the DCE Security Server request.

eTrust CA-ACF2 Support for the DCE Security Server

eTrust CA-ACF2 defines a MVS user to z/OS Unix System Services DCE through DCE profile records.

Record ID	Fields
recid	UUID(*uuid)* DCENAME(dcename) HOMEUUID(*homeuuid*) HOMECELL(*homecell*) AUTOLOG|NOAUTOLOG

Field Descriptions

recid
MVS userid; this value cannot be masked.

UUID
Specifies the character string form of the user's UUID. This string can have a maximum length of 36 characters.

DCENAME
Specifies the character principal name of the user. This field can be a maximum of 1023 characters.

HOMEUUID
Specifies the character string form of the user's home cell UUID. This field can contain a maximum of 36 characters.

HOMECELL
Specifies the character home cell name. This field can contain a maximum of 1023 characters.

AUTOLOG
Indicates that the user should be automatically signed on to z/OS Unix System Services DCE. NOAUTOLOG is the default.

DCEKEY
A field that is not administered by the ACF command and can never be displayed. The DCEKEY is stored in encoded (masked) or encrypted form in the database depending on how it was originally designated to be stored in the KEYSMSTR profile record. See the Defining DCE under eTrust CA-ACF2 for additional information.

eTrust CA-ACF2 automatically creates a UUID/logonid cross-reference table at start-up. If any new profile records are inserted or any changes are made to the UUID or HOMEUUID fields, the cross-reference tables must be refreshed using the F ACF2,OMVS console command.

Note: Support for the DFS/SMB encrypted password is included in the DCE profile record. The smbpw command does an EXTRACT REPLACE for the DCEKEY. This value is stored in the DCEKEY field. The DCEKEY is a non-displayable portion of the DCE profile record.

Example: An MVS user is defined to z/OS Unix System Services DCE by assigning a UUID and HOMEUUID to the user. To do this, create a DCE profile record as follows:

```
SET PROFILE(USER) DIV(DCE)
INSERT JORDAN DCENAME(JORDANM) UUID(uuid) HOMEUUID(homeuuid) -
HOMECELL(homecell) NOAUTOLOG
```

Using the MVSEXPT and MVSIMPT Utilities

The IBM *z/OS Unix System Services DCE Administration Guide* contains a chapter that discusses security interoperability and the MVSEXPT and MVSIMPT utilities. To successfully run these utilities in an eTrust CA-ACF2 environment, you must perform the following actions for each utility:

MVSEXPT
The DCE MVSEXPT utility reads the DCE registry and creates a file containing MVS TSO commands that are used to create corresponding users on MVS. You can use the ACF$DCE EDIT MACRO to convert these commands to eTrust CA-ACF2 command input.

When you use the MVSEXPT utility, run stage one and then edit the output file. While in edit, enter ACF$DCE on the command line to transform the commands into eTrust CA-ACF2 command input to the ACFBATCH utility. Allocate the CLIST library containing the ACF$DCE macro as a SYSPROC library.

MVSIMPT

The DCE MVSIMPT utility takes a list of MVS users and creates corresponding entries in the DCE registry. To create the input list of MVS users, first create the eTrust CA-ACF2 user profile DCE records; then run the ACF2UNLD utility.

Define a user profile record for each MVS user imported to DCE. By default, MVSIMPT uses the MVS userid as the DCE principal name. To use a different value, specify it in the DCENAME keyword when defining that profile record. Do not specify a UUID because MVSIMPT assumes that the DCE user already exists if a UUID value is present.

The following example shows the creation of a user profile DCE record that is retrieved later by the ACF2UNLD utility:

```
SET PROFILE(USER) DIV(DCE)

INSERT JORDAN DCENAME(JORDANM)-
HOMEUUID(ABBC323C-5CE2-11CF-A61E-08005A470BA1)-
HOMECELL(/…/CIS_TEST1.CIS.DOG.COM)-
NOAUTOLOG
```

Once the profile records exist, the ACF2UNLD utility reads these records and creates an output file that is used as input to the MVSIMPT utility. The ACF2UNLD utility can only return those user profile DCE records that the user running the job has the authority to list, so ensure that the user running the utility has the ability to list all user profile DCE records. The output data is placed into the file specified in the ACF2UNLD DD statement. Do not specify any DCB information for the output file.

An example of the JCL to run this utility follows:

```
//jobname  JOB  job_information
//STEP1    EXEC PGM=ACF2UNLD
//ACF2UNLD DD  DSN=cai.unload.data,
//             SPACE=(CYL,(1,1)),
//             UNIT=unit,
//             DISP=(,CATLG)
```

The following condition codes can be returned by the utility:

00

Successful completion.

04

No user profile DCE records were returned. Ensure that the user running the utility has the proper authority.

08

An OPEN error occurred.

The output file contains sorted DCE information; no further sorting is needed. Review the output and make any appropriate changes. Copy the file to the HFS file system using the RACFUNLD file name. RACFUNLD is documented in the IBM *z/OS Unix System Services DCE Administration Guide*. Follow the instructions in that guide for the remaining operation of the MVSIMPT utility.

z/OS DCE Support

DCE provides a framework for programs and data on multiple platforms to interoperate following a set of industry-recognized standards. A set of named servers provides central services needed to support the DCE framework. For example, a DCE Time Server maintains a consistent time-of-day clock for all platforms within the DCE environment. A DCE Security Server provides central system entry validation and single-sign on authentication for all DCE users.

An IBM mainframe can fully participate in a DCE environment using the DCE support included with z/OS. This support allows DCE programs and data to reside on the mainframe under z/OS Unix System Services and allows the mainframe to interoperate with other DCE platforms. In addition to this support, a separate, optional IBM product enables the mainframe to be a DCE Security Server. While the functions and the administration of a DCE Security Server are completely separate from the mainframe security product (for example, eTrust CA-ACF2, RACF, and so forth), this optional product does allow a limited interface.

Defining DCE under eTrust CA-ACF2

Perform the following steps to define DCE under eTrust CA-ACF2:

1. z/OS Unix System Services must be up and running before DCE support can be configured and activated. The eTrust CA-ACF2 setup of basic z/OS Unix System Services is separately described but should be reviewed for completeness before continuing. See the Creating eTrust CA-ACF2 Logonid Records for z/OS Unix System Services section.
2. Define a logonid for the DCEKERN started task (daemon) that executes when DCE is active. Like any other OMVS userid, this logonid must belong to at least one OMVS group ID. The commands that follow define an OMVS group and logonid for DCEKERN using values matching IBM DCE documentation:

```
SET LID
INSERT DCEKERN NAME(DCE KERNEL) STC GROUP(DCEGROUP) UID(0)
HOME(/opt/dcelocal/home/dcekern) OMVSPGM(/bin/sh)

SET PROFILE(GROUP) DIV(OMVS)
INSERT DCEGROUP GID(nn)
```

3. The DCEKERN started task requires access to a FACILITY class resource. To accomplish this, create and store the following rule set:

```
SET RES(FAC)
Comp *
$KEY(DCEKERN) TYP(FAC)
START.REQUEST UID(DCEKERN's UID string) SERVICE(READ) ALLOW
STORE
```

4. The environment variables (ENVVAR) files for DCE must be updated to disable automatic logon when DCE is started. To make this update, specify _EUV_AUTOLOG=NO in the ENVVAR files in the following directories:

```
/opt/dcelocal/home/dcekern
/opt/dcelocal/home/cdsadv
/opt/dcelocal/home/cdsclerk
/opt/dcelocal/home/dtsd
/opt/dcelocal/home/dced
```

   Failure to make these changes will result in occurrences of the following message when the DCEKERN started task is started.

```
EUVS00750E AUTOMATIC DCE SIGNON PROCESSING FAILED
 USER NOT DEFINED TO MVS SECURITY MANAGER CORR: 0 (EUVZUSER)
```

   This message is not a fatal error; even when issued, DCEKERN will fully initialize.

5. eTrust CA-ACF2 lets a DCE segment be optionally defined for any user so that the user's DCE information is reflected if desired. Note that this segment and the information contained within this segment are not required for any user.

   Users who enter an explicit DCE_LOGIN command under OMVS when beginning their DCE sessions do not use this user information since the DCE_LOGIN command instead relies on the DCE Security Server to provide this data. A user who enters DCE using an already-established DCE session (from another platform) also does not use the DCE segment information, since this data is automatically pre-supplied as part of the user's credentials when the connection is established.

   The only time that this information is used is when the mainframe houses the DCE Security Server or when an automatic sign-on (implicit DCE_LOGIN) is desired for OMVS users. For the first of these cases, see the IBM documentation on the z/OS DCE Security Server. For the second case, the user segment information must be defined and AUTOLOGIN must be set within the user's DCE profile record.

6. Perform the following steps to define a KEYSMSTR profile record under eTrust CA-ACF2 for the purposes of designating how the DCEKEY in the DCE record segment (i.e., DFS/SMB password) should be stored, encoded (masked or encrypted):

   - Activate the KEYSMSTR class for DCE administration, by loading the DCE.PASSWORD.KEY record into storage via the addition of an entry for the PKEY infostorage directory in the GSO INFODIR record. For example:

     ```
     ACF
     set control(gso) sysid(prod)
     change infodir types(r-pkey)
     ```

   - Insert a KEYSMSTR profile segment record, specifying a 16-digit encryption key mask value as the SSKEY accompanied by **one** of the following attributes, KEYMASK or KEYCRYPT, to encode (mask) or encrypt the DFS/SMB password. For example:

     ```
     ACF
     set profile(keysmstr) div(ssignon)
     insert dce.password.key sskey(1234567812345678) keymask
     ```

   - Refresh the GSO INFODIR record and rebuild the P(KEY) directory. This must be done whenever any changes are made to the DCE.PASSWORD.KEY record. For example:

     ```
     F ACF2, REFRESH(INFODIR)
     F ACF2, REBUILD(key),CLASS(p)
     ```

   Note: If sharing databases with a r6.3 or lower, eTrust CA-ACF2 systems, you cannot define a KEYSMSTR profile record.

   See the KEYSMSTR Profile Record section of the *Administrator Guide* for more information.

Distributed File Service

This Distributed File Service (DFS) is an application that sets up for the DCE access to a set of files and directories on a global basis regardless of machine boundaries. For details and additional information, see the *z/OS Distributed File Service Administrator Guide and Reference*.

To set up this environment under eTrust CA-ACF2, perform the following:

1. Define logonids and profile records for the DFS associated started tasks:

```
SET LID
INSERT DFSKERN NAME(DFS started task) STC GROUP(DFSGRP) UID(0)
HOME(/opt/dfslocal/home/dfscntl)

INSERT DFS NAME(DFS started task) STC GROUP(DFSGRP) UID(0)
HOME(/opt/dfslocal/home/dfscntl)

INSERT DFSCM NAME(DFS started task) STC GROUP(DFSGRP) UID(0)
HOME(/opt/dfslocal/home/dfscntl)
```

2. Define the group assigned to DFS related logonids as follows:

```
SET PROFILE(GRP) DIV(OMVS)
INSERT DFSGRP GID(nn)
```

   where *nn* is an appropriate group number.

3. Allow the logonids related to DFS update access to the FACILITY resource DFSKERN.START.REQUEST. The following commands are an example of setting up the appropriate resource rule:

```
SET RESOURCE(FAC)
COMPILE
$KEY(DFSKERN)
START.REQUEST UID(dfjkern_uid) SERVICE(READ,UPDATE) ALLOW
START.REQUEST UID(dfs_uid) SERVICE(READ,UPDATE) ALLOW
START.REQUEST UID(dfscm_uid) SERVICE(READ,UPDATE) ALLOW

STORE
```

   Note: If the DFSKERN FACILITY resource rule already exists at your site, simply add the appropriate rule entries to it.

4. Issue the following commands to activate the changes specified above:

```
F ACF2,REBUILD(USR),CLASS(P)
F ACF2,REBUILD(GRP),CLASS(P)
F ACF2,OMVS
F ACF2,REBUILD(FAC)
```

   Note: The last command is only necessary if the FACILITY class resource rules were made resident via the GSO INFODIR record.

5. Individual users are defined via DCE credentials. See the eTrust CA-ACF2 Support for the DCE Security section for details.

Network File System (NFS)

z/OS Network File System (NFS) enables remote access to *z/OS* data sets and UNIX System Services HFS files and directories. NFS can protect file systems on MVS using four protection schemes. These setting are defined within the NFS "Site Attributes' Security".

The possible settings are:

NONE
Do not restrict access. NO MVS userid required.

EXPORTS
Restrict access by client IP address. NO SAF check.

SAF
Use SAF to control access to data sets or HFS data.

SAFEXP
Use SAF and EXPORTS to control access. This is the most secure setting.

Both SAF and SAFEXP require the user to use the 'mvslogin' process to validate access through a SAF call. For this reason, eTrust CA-ACF2 recommends a minimum of security (saf). Users who attempt to access HFS data must have a valid OMVS segment assigned to their MVS logonid. Access to HFS files is determined by validating the client's UID and group against the file UNIX permission bits. Under normal circumstances, access to z/OS data sets requires the z/OS NFS server and client user to pass a security check for the resource. The exception to this is when DataCaching is enabled. Data caching causes data to be stored on the z/OS NFS client system.

When data caching is enabled, the first user attempting to access a z/OS data set must pass a SAF security check. This SAF call is issued by the z/OS NFS Server. Once passed, the data set is stored in the z/OS NFS Client server. Subsequent requests will let all users access the cached data without further restrictions. Data caching by default is enabled. eTrust CA-ACF2 recommends that DataCaching be disabled. With DataCaching(N), no client data caching takes place, therefore each user must pass the z/OS NFS Security server check prior to being granted access to data. z/OS NFS Server 'Site Attribute' 'checklist' lists the files and or directories for which SAF security is bypassed, even when SAF or SAFEXP is specified. For this reason, proper care must be taken to secure this data set. The checklist data set is defined using the CHKLIST DD in the MVSNFS procedure.

eTrust CA-ACF2 Support for z/OS NFS

To define eTrust CA-ACF2 support for z/OS NFS, use the following procedure:

1. Create logonids and user profile records for the NFS Server, the NFS Client, the NFS Lock Manager (NLM), and the NFS Network Status Monitor (NSM) started tasks (these logonids require a UID(0)):.

   ```
   SET LID
   INSERT MVSNFS NAME(NFS Server) STC GROUP(OMVSGRP) NON-CNCL UID(0)
   HOME(/u/mvsnfs) OMVSPGM(/bin/sh)

   INSERT MVSNFSC NAME(NFS Client) STC GROUP(OMVSGRP) UID(0) HOME(/u/mvsnfsc)
   OMVSPGM(/bin/sh)

   INSERT MVSNLM NAME(NFS Lock Manager) STC GROUP(OMVSGRP) UID(0)
   HOME(/u/mvsnlm) OMVSPGM(/bin/sh)

   INSERT MVSNSM NAME(NFS Network Status Monitor) STC GROUP(OMVSGRP) UID(0)
   HOME(/u/mvsnsm) OMVSPGM(/bin/sh)
   ```

2. If the group assigned to these started tasks has not been previously created, create a group profile record:

   ```
   SET PROFILE(GROUP) DIV(OMVS)
   INSERT OMVSGRP GID(nn)
   ```

   Replace *nn* with the appropriate GID value.

3. The logonids associated with the started tasks require access to data sets that the remote users are accessing. Ensure that these logonids have the appropriate authority to access these data sets.

4. To activate the changes made for NFS, issue the following commands:

   ```
   F ACF2,REBUILD(USR),CLASS(P)
   F ACF2,REBUILD(GRP),CLASS(P)
   F ACF2,OMVS
   ```

Firewalls

z/OS provides the ability to run a firewall under UNIX System Services. Support is distributed in part with the communication server and in part with the security server. The z/OS firewall can reduce, but not necessarily eliminate the need for a non-z/OS platform firewall. The firewall itself is not configured using eTrust CA-ACF2. Administration is performed through configuration files.

Follow these steps to set up the z/OS firewall with eTrust CA-ACF2:

1. Define the firewall startup address space logonid and user profile record:

```
INSERT FWKERN NAME(Firewall Startup Id) STC GROUP(FWGRP) UID(0)
HOME(/usr/lpp/fw/home/fwkern) OMVSPGM(/bin/sh)
```

2. Define the additional started task logonids used by the firewall daemons and their user profile records:

```
INSERT ICAPSLOG NAME(Firewall Daemon Id) STC GROUP(FWGRP) UID(0)
HOME(/u/fwkern) OMVSPGM(/bin/sh)

INSERT ICAPSOCK NAME(Firewall Daemon Id) STC GROUP(FWGRP) UID(0)
HOME(/u/fwkern) OMVSPGM(/bin/sh)

INSERT ICAPPFTP NAME(Firewall Daemon Id) STC GROUP(FWGRP) UID(0)
HOME(/u/fwkern) OMVSPGM(/bin/sh)

INSERT ICAPCFGS NAME(Firewall Daemon Id) STC GROUP(FWGRP) UID(0)
HOME(/u/fwkern) OMVSPGM(/bin/sh)

INSERT ICAPSTAK NAME(Firewall Daemon Id) STC GROUP(FWGRP) UID(0)
HOME(/u/fwkern) OMVSPGM(/bin/sh)

INSERT ICAPIKED NAME(Firewall Daemon Id) STC GROUP(FWGRP) UID(0)
HOME(/u/fwkern) OMVSPGM(/bin/sh)
```

3. Add a profile record to define the group used by the firewall logonid using any unused GID number:

```
SET PROFILE(GROUP) DIV(OMVS)
INSERT FWGRP GID(nn)
```

4. Allow FWKERN to issue start commands:

```
SET RESOURCE(FAC)
COMPILE
$KEY(FWKERN) TYPE(FAC)
 START.REQUEST UID(fwkern) ALLOW

STORE
```

5. Allow FWKERN access to READ the TCP/IP data sets using the standard data set access rules:

```
COMPILE
$KEY(TCPIP)
 - UID(fwkern) R(A)

STORE
```

   Note: The high level qualifier of these data sets might have been renamed from "TCPIP" when installed on your system.

6. Allow the FWKERN logonid access to the SMF logging facility and the BPX.DAEMON facility. Add the following rules entries to the BPX FACILITY rule:

```
SMF UID(fwkern) SERVICE(READ) ALLOW
DAEMON UID(fwkern) SERVICE(READ) ALLOW
```

7. Any user specified on the configuration GUI must be given permission to update the FACility resource ICA.CFGSRV even if the user has superuser status. Create the following resource rule to allow the required users access to this resource:

```
$KEY(ICA) TYPE(FAC)
CFGSRV UID(userid_uid) SERVICE(READ,UPDATE) ALLOW
```

8. If FWKERN uses the ISAKMP server to allow secure communications through non-secure networks, the following needs to be done with eTrust CA-ACF2:

   a) Allow the FWKERN logonid access to the FACility resources CDS.CSSM, CDS.CSSM.CRYPTO, and CDS.CSSM.DATALIB as follows:

```
$KEY(CDS) TYPE(FAC)
CSSM         UID(fwkern) SERVICE(READ) ALLOW
CSSM.CRYPTO  UID(fwkern) SERVICE(READ) ALLOW
CSSM.DATALIB UID(fwkern) SERVICE(READ) ALLOW
```

   b) If the system operates with z/OS Unix System Services, let FWKERN access the FACility resource BPX.SERVER so that the ISAKMP server can use OSCF services as follows. Add the following rule entry to the BPX FACILITY rule:

```
SERVER UID(fwkern) SERVICE(READ) ALLOW
```

   c) So the ISAKMP server can use the OCEP services, allow FWKERN access to the FACility resources IRR.DIGTCERT.LIST and IRR.DIGTCERT.LISTRING as follows:

```
$KEY(IRR) TYPE(FAC)
DIGTCERT.LIST     UID(fwkern) SERVICE(READ) ALLOW
DIGTCERT.LISTRING UID(fwkern) SERVICE(READ) ALLOW
```

   d) The ISAKMP server supports peer authentication using RSA signature mode. RSA signatures require that digital certificates be stored in eTrust CA-ACF2 and connected onto a key ring.

   e) See the SHOW CRITMAP section. Show CRITMAP displays information contained in CRITMAP records as laid out in the internal CRITMAP table. The diplay shows the record id, SYSID, APPLID, USERID, and associated application variables.

```
sho critmap

- CRITERIA TABLE –

Record key          SYSTEMID   APPLID    USERID    APPLICATION VARS
=================== ========   ======    ======
==============================
CRITMAP.PUBLIC2     *          CICSAPPL  PUBLIC2

CRITMAP.PLATINUM    *          HRAPPL    PLATUSER  COMPANY=PLATINUM

CRITMAP.UCCEL       *          HRAPPL    UCCUSER   COMPANY=UCCEL

CRITMAP.PUBLIC1     *          WEBAPPL   PUBLIC1
```

Adding Firewall Administrators to FWGRP

Firewall administrators must be members of the group FWGRP or have superuser authority. The following commands give an administrator access to the firewall group:

```
SET RESOURCE(TGR)
COMPILE
$KEY(FWGRP) TYPE(TGR)
 UID(adminid) ALLOW

STORE
```

Firewalls can invoke z/OS Integrated Cryptographic facilities to perform internal security functions. These services are protected using the resource class CSFSERV. This is accomplished using the following commands:

```
SET RESOURCE(SAF)
COMPILE
$KEY(service-name) TYPE(SAF)
 UID(userid) SERVICE(READ) ALLOW

STORE
```

Note: The CSFSERV class defaults to SAF and should be changed to a type code unique to the CSFSERV class via an eTrust CA-ACF2 CLASMAP record. See the Integrated Cryptographic Service Facility Support section for additional information.

Users must be permitted to the necessary individual services. The individual service-names are documented in the appropriate IBM firewall manuals and the IBM *ICSF/MVS Administrators Guide*. Also see Integrated Cryptographic Service Facility Support in this chapter.

Integrated Cryptographic Service Facility

The Integrated Cryptographic Service Facility (ICSF) from IBM is a high-powered cryptographic coprocessor that allows z/OS applications to use cryptography. The z/OS security server provides APIs to invoke ICSF rather than use software algorithms to perform the same functions. ICSF also provides various functions involved with the management of keys. These services combine to provide a site with the ability to manage public keys.

The interface to ICSF and the security manager consists of two areas: the ability for an application to read a key and the APIs to manage keys. ICSF makes SAF calls to external security to allow ICSF to be secured and audited. There are two resource classes that provide this support:

- **CSFKEYS —** This class is used to secure encryption keys. The value, which is owned and permitted, is the key label. The key label is specified in the CKDS or PKDS when a key is defined.
- **CSFSERV —** This class is used to secure the various cryptographic services needed to encrypt data and manage keys. The services are listed in the IBM *z/OS Integrated Cryptographic Service Facility: Administrator's Guide*.

These two classes are predefined in the internal CLASMAP records and specify a default type code of SAF for any resource rules. For example, if a user needs to use the ANSI X9.17 key export callable service, the SAF call would validate a resource of CSFAKEX, as indicated in the ICSF documentation. To allow access to this resource quoted in the example, a resource rule, similar to the one presented below, should be coded:

```
SET RESOURCE(SAF)
COMPILE
$KEY(CSFAKEX) TYPE(SAF)
 UID(userid) ALLOW

STORE
```

Note: You can separate the ICSF resource rules into unique type codes, if desired, by defining CLASMAP records and specifying a different rsrctype as suggested below:

```
SET CONTROL(GSO)
INSERT CLASMAP.CSFKEYS RESOURCE(CSFKEYS) RSRCTYPE(CSK) ENTITYLN(73)
INSERT CLASMAP.CSFSERV RESOURCE(CSFSERV) RSRCTYPE(CSF) ENTITYLN(8)
```

It has been noted that security calls pertaining to ICSF issued in a CICS environment are validating the logonid of the CICS region not the logonid of the user making the call. IBM has noted that ICSF currently works by picking up the userid from the associated TCB, which contains the CICS region userid. ICSF does not currently have the ability to pick up the signed in userid that actually made the call in a CICS environment.

Novell Network Services

Novell Network Service allows a workstation to share files, printers, and other resources among various PCs running on DOS and Windows. Support has been added to map a Novell Directory Services (NDS) application user identity to an eTrust CA-ACF2 logonid.

Novell Network Services requires these special logonids for processing: NWROOT, NWUSER, and NOBODY.

The following example shows how to set up the Network Services logonids and their associated profile records:

```
SET LID
INSERT NWROOT GROUP(NWGROUP) RESTRICT UID(nn) HOME (/) OMVSPGM(/bin/sh)

INSERT NWUSER GROUP(NWGROUP) RESTRICT UID(nn) HOME (/) OMVSPGM(/bin/sh)

INSERT NOBODY GROUP(NOGROUP) RESTRICT UID(nn) HOME (/) OMVSPGM(/bin/sh)
```

Replace *nn* with the appropriate UID value.

```
SET PROFILE(GROUP) DIV(OMVS)
INSERT NWGROUP GID(xx)
INSERT NOGROUP GID(yy)
```

Replace *xx* and *yy* with appropriate GID values.

To support Novell Network Services, insert a NDS user profile record for each user using Novell Network Services:

```
SET PROFILE(USER) DIV(NDS)
INSERT NWROOT UNAME(nwroot)
INSERT NWUSER UNAME(nwuser)
INSERT NOBODY UNAME(nobody)
INSERT USERA UNAME(usera on Novell)
```

Anytime changes are made to USER or GROUP profile records, the following commands must be issued to activate the changes immediately:

```
F ACF2,REBUILD(USR),CLASS(P)
F ACF2,REBUILD(GRP),CLASS(P)
F ACF2,OMVS
```

Kerberos

Network Authentication and Privacy Service, known as Kerberos, uses eTrust CA-ACF2 to store and administer information about principals and realms. KERB and KERBLINK USER profile records and REALM GSO records have been incorporated into eTrust CA-ACF2 to store this information.

The KERB USER profile record is used to store information about Network Authentication and Privacy Service principals on your local system. The KERBLINK USER profile record lets you map principals to eTrust CA-ACF2 logonids on your system, and GSO REALM record defines the local Network Authentication and Privacy Service realm and its trust relationships with foreign realms.

eTrust CA-ACF2 also provides support for callable services, namely R_ticketserv and R_kerbinfo for z/OS applications servers that use Kerberos services.

Note that as Kerberos is being incorporated into eTrust CA-ACF2, some terminology and naming structures might change.

Authentication of Principals

Kerberos for z/OS verifies requests as a trusted third-party authentication service. Using conventional shared secret key cryptography, Kerberos confirms the identities of principals (users), without relying on authentication by the host operating system, without basing trust on host addresses, without necessitating physical security of all hosts on the network, and under the premise that packets traveling along the network can be read, changed, and inserted at will.

Kerberos uses electronic tickets to authenticate a user to a server. A ticket, which is good only for a single server and a single user during a certain period of time, is an encrypted message containing the name of the user and server, the user's network address, a time stamp, and a session key. Once the user gets this ticket, he can use it to access the server as many times as desired until the ticket expires. The user cannot decrypt the ticket but can only present it to the server. Nobody listening in on the network can read or modify the ticket as it passes through the network without detection or invalidation.

The Kerberos protocol involves two servers, the Kerberos Authentication Server and one or more TGSes (Ticket-Granting Servers). The steps involved in the Kerberos process are as follows.

1. To obtain a ticket to a particular target server, the user first requests from the Kerberos Authentication Server a ticket to the Kerberos TGS. This request takes the form of a message containing the user's name and the name of his TGS (there can be several).

2. The Authentication Server looks up the user in its database and then generates a session key to be used between the user and the TGS. Kerberos encrypts this session key using the user's secret key (a one-way hash of the user's password). Then it creates a TGT (ticket-granting ticket) for the user to present to the TGS and encrypts the TGT using the TGS's secret key (which is known only to the Authentication Server and the TGS). The Authentication Server sends both of these encrypted messages back to the user.
3. The user decrypts the first message and recovers the session key. Next, the user creates an authenticator consisting of his name and address and a time stamp, all encrypted with the session key just generated by the Kerberos Authentication Server. The user then sends a request to the TGS for a ticket to a particular target server. This request contains the name of the server, the TGT received from Kerberos (which is already encrypted with the TGS's secret key), and the encrypted authenticator.
4. The TGS decrypts the TGT with its secret key and then uses the session key included in the TGT to decrypt the authenticator. It compares the information in the authenticator with the information in the ticket, the user's network address with the address the request was sent from, and the time stamp with the current time. If everything matches, it allows the request to proceed.
5. The TGS creates a new session key for the user and target server and incorporates this key into a valid ticket for the user to present to the server. This ticket also contains the user's name, network address, a time stamp, and an expiration time for the ticket--all encrypted with the target server's secret key--and the name of the server. The TGS also encrypts the new user-target session key using the session key shared by the user and the TGS. It sends both messages to the user.
6. The user decrypts the message and extracts the session key for use with the target server. The user is now ready to authenticate himself to the server. He creates a new authenticator encrypted with the user-target session key that the TGS generated. To request access to the target server, the user sends along the ticket received from Kerberos (which is already encrypted with the target server's secret key) and the encrypted authenticator. Because this authenticator contains plain text encrypted with the session key, it proves that the sender knows the key. Just as important, encrypting the time of day prevents an eavesdropper who records both the ticket and the authenticator from replaying them later.
7. The target server decrypts and checks the ticket and the authenticator, also checking the user's address and the time stamp. If everything checks out, the server now knows the user is who he claims to be, and the two share an encryption key that they can use for secure communication. (Since only the user and the server share this key, they can assume that a recent message encrypted in that key originated with the other party.)

8. For those applications that require mutual authentication, the server sends the user a message consisting of the time stamp plus 1, encrypted with the session key. This serves as proof to the user that the server actually knew its secret key and was able to decrypt the ticket and the authenticator.

Realms

The Kerberos protocol functions across organizational boundaries. An organization running on a Kerberos server must establish its own *realm*. A client's name contains the name of the realm in which a client is registered, so that the application server can use the realm name to determine whether to grant a request.

With the establishment of *inter-realm* keys, the administrators of the two realms can permit a client authenticated in one realm to use its credentials in the other realm. Exchanging inter-realm keys registers the ticket-granting service of each realm as a principal in the other. A client can then procure a ticket-granting ticket for the remote realm's ticket-granting service from its local ticket-granting service. Tickets distributed to a service in the remote realm indicate that the client was authenticated from another realm.

This procedure can be used to authenticate throughout an organization across multiple realms. In order to construct an authentication path to a foreign realm, the local realm must share an inter-realm key with the target realm or with an intermediate realm that communicates with the target realm or with another intermediate realm.

Realms are often organized hierarchically. A realm shares a key with its parent and different key with each child. In a hierarchical organization, an authentication path can be easily established if two realms do not share an inter-realm key. If a hierarchical organization is not in place, referring to a database in order to build an authentication path between realms can be required.

Although realms are often hierarchical, intermediate realms can be overridden, resulting in cross-realm authentication through alternate authentication paths. The end-service must know which realms were transited when determining how much confidence to have in the authentication process. To aid this process, a field in each ticket includes the realm names that helped authenticate the client.

Before setting up the eTrust CA-ACF2 definitions to support the Network Authentication and Privacy Service Implementation, you must understand the records that must be defined and how and their relationship. REALM records define your local foreign realms. The local realm name (REALM field on the REALM GSO record) is used in the construction of the key generated for foreign realms and local principals. This is very important especially in a shared database environment. Care must be taken when changing the local realm name. If the local realm name is changed after creating the keys for your principals and realms, the keys are invalid and must be reset.

Shared Database Environment

Keys for principals are kept in both the eTrust CA-ACF2 Logonid database and in the CA-ACF2 Inforstorage database. The Logonid record retains the default DES kerberos key. The PASSWORD segment of the USER profile record is used to retain the DES3 and DESD kerberos keys when the GSO OPTS record specifies KERBLVL(1).

Defining Your Local Realm

You must define your local realm to eTrust CA-ACF2 before you define local principals. This is because the local realm is used to generate keys for local principals. You define your local realm by creating a REALM GSO record called KERBDFLT. You can specify the following information about your local realm:

Command syntax:

```
INSERT REALM.qualifier REALM(Kerberos-realm-name) MINTKTLF(Min-ticket-life)
MAXTKTLF(Max-ticket-life) DEFTKTLF(Default-ticket-life) DES DES3 DESD
KERBPSWD(Kerberos-password)
```

Qualifier
A label-name to identify the GSO realm record. You must specify the name of **KERBDFLT** for the local realm.

REALM
(Kerberos realm name) Specifies the unqualified name of the local realm for Network Authentication and Privacy Service Server. The maximum length of this field is 117 characters. The fully qualified name of the local realm, /.../Kerberos_realm_name/krbtgt/Kerberos_realm_name, must not be specified.

The name assigned to the local realm limits the length of local principal names, since fully qualified principal names/…/Kerberos_realm_name/Principal_name, cannot exceed 240 characters.

Regardless of the case in which it is entered, eTrust CA-ACF2 rolls the name of the local Network Authentication and Privacy Service realm to upper case. However, eTrust CA-ACF2 does not ensure that a valid Kerberos-realm-name has been specified.

Note: Because the relationship between the REALM value and generating Kerberos tickets for principal users is based, in part, on the local REALM value, care must be taken when choosing a REALM value. Renaming the REALM should be avoided at all costs.

MAXTKTLF
Specifies the maximum ticket life in seconds and is represented by a numeric value between 1 and 2 147 483 647. Note that 0 is not a valid value. This keyword is only applicable when defining the KERBDFLT GSO REALM record. If MAXTKTLF is specified, DEFTKTLF and MINTKTLF must also be specified.

MINTKTLF
Specifies the minimum ticket life in seconds, and is represented by a numeric value between 1 and 2 147 483 647. Note that 0 is not a valid value. This keyword is only applicable when defining the KERBDFLT GSO REALM record. If MINTKTLF is specified, DEFTKTLF and MAXTKTLF must also be specified.

DEFTKTLF
Specifies the default ticket life in seconds, and is represented by a numeric value between 1 and 2 147 483 647. Note that 0 is not a valid value. This keyword is only applicable when defining the KERBDFLT GSO REALM record. If DEFTKTLF is specified, MINTKTLF and MAXTKTLF must also be specified.

DES|NODES
Enables the DES encryption type setting to be defined for this user. This field is valid when KERBLVL(1) is active on the GSO OPTS record. The default is DES.

DES3|NODES3
Enables the DES3 encryption type setting to be defined for this user. This field is valid when KERBLVL(1) is active on the GSO OPTS record. The default is DES3.

DESD|NODESD
Enables the DESD encryption type setting to be defined for this user. This field is valid when KERBLVL(1) is active on the GSO OPTS record. The default is DESD.

KERBPSWD
Specifies a password for the default realm. The maximum length of this value is 8-characters. The PASSWORD keyword is applicable to all REALM GSO record definitions. A password is required in order for the local realm to grant ticket-granting-tickets and a password must be associated with the definition of an inter-realm trust relationship, or the definition is incomplete.

The Kerberos password that you define to eTrust CA-ACF2 might consist of any character. It is highly recommended that use of any of the EBCDIC variant characters be avoided to prevent problems with different code pages. Passwords are case-sensitive and are maintained in the case in which they are entered. **Note:** This password is not an eTrust CA-ACF2 user password and is not constrained by password rules that might be specified to control user passwords.

Command example:

```
Set control(GSO) sysid(test)
 CONTROL
Insert realm.kerbdflt realm(lisle.ca.com) min(30) max(86400) def(36000)
kerb(children)

  TEST/REALM.KERBDFLT LAST CHANGED BY ADMIN01 ON 08/28/04-12:13
       CURKEYV(1) DEFTKTLF(36000) DES DESD DES3 MAXTKTLF(86400) MINTKTLF(30)
       REALM(LISLE.CAI.COM)
 CONTROL
```

In this example, the realm name is KERBDFLT, which identifies this record as the default realm record. The Kerberos realm name is Local.ca.com, with a password of children. The default ticket life is 10 hours, a minimum ticket life of 30 seconds, and a maximum ticket life of 24 hours.

See the *Administrator Guide* for any additional information about GSO REALM records.

Defining Local Principals

eTrust CA-ACF2 logonids can become Kerberos local principals by defining a KERB segment USER profile record. The KERB segment maps a Kerberos for z/OS application user identity to an eTrust CA-ACF2 logonid. Each local pricipal must have an eTrust CA-ACF2 password so do not define a local principal with the RESTRICT attribute.

The KERB segment has the following fields:

recid
This is a one to eight character logonid that is to be associated with the local principal name. A one to eight character suffix can be appended to the userid to create a unique record key. The suffix must be separated from the userid with a period.

KERBNAME

Specifies the local Kerberos principal name. The *kerberos-principal-name* you define can consist of any character except the @ (X'7'C) character. It is highly recommended that any of the EBCDIC variant characters be avoided to prevent problems between different code pages. This field is case sensitive. eTrust CA-ACF2 will not ensure that a valid Kerberos principal name has been entered. A local Kerberos principal name must not be qualified with a realm name when specified in the KERBNAME parameter. eTrust CA-ACF2 will verify that the local principal name, when qualified with the local realm name, does not exceed 240 characters.

For example, if the local realm name is REALM1, fully qualified local principal names are prefixed with/…/RELAM1/ and are limited to 228 characters. If the local realm name is REALM1.LISLE.CA.COM, fully qualified local principal names are prefixed with /…/REALM1.LISLE.CA.COM/ and are limited to 215 characters. The length verification requires that the GSO REALM record for the local realm KERBDFLT be defined and contain the name of the local realm before inserting the KERB USER profile records. Otherwise, the local Kerberos principals might not be properly defined.

MAXTKTLF

Maximum ticket life for this field is in seconds. The range of values is from 1 to 2,147,483,647 seconds. If the MAXTKTLF parameter is defined for a principal, the system takes the most restrictive of the values defined for the principal and the value specified on the definition of the local realm (GSO REALM record with REALM(KERBDFLT). If the value in the principal definition exceeds the value in the local realm definition, the value in the local realm definition is used.

DES|NODES

Enables the DES encryption type setting to be defined for this user. This field is valid when KERBLBL(1) is active on the GSO OPTS record. The default is DES.

DES3|NODES3

Enables the DES3 encryption type setting to be defined for this user. This field is valid when KERBLVL(1) is active on the GSO OPTS record. The default is DES3.

DESD|NODESD

Enables the DESD encryption type setting to be defined for this uer. This field is valid when KERBLVL(1) is active on the GSO OPTS record. The default is DESD.

KERB-VIO

Specifies the number of Kerberos key violations. This field is similar to the PSWD-VIO count and will e used to suspend the user's LID when the count exceeds the global password violation limit.

Command example:

```
set profile(user) div(kerb)
 PROFILE
insert TLC089 kerbname(steven.ca.com) maxtktlf(86400)
 KERB / TLC089 LAST CHANGED BY ADMIN01 ON 08/26/04-12:15
               DES DESD DES3 KERB-VIO(0) KERBNAME(steven.ca.com)
               MAXTKTLF(86400)
 PROFILE
```

In this example, a KERB segment was added to user TLC089, with a Kerberos principal name of steven.tlc.com, and a maximum ticket life of 24 hours.

Generating Keys for Local Principals

Each local principal must have a key registered with the local Network Authentication and Privacy Service server in order to be recognized as a local principal. The user's definition as a local principal is not complete until the key is generated. The key is generated from the principal's eTrust CA-ACF2 user password at the time of the user's password change. If you want to change the password to generate the key, be sure to use a password change facility that will not result in an expired password that the user must change at the next logon. For example, you can use NOPSWD-EXP on the change command.

A local principal's key is revoked whenever the user's eTrust CA-ACF2 user id is suspended or the password is expired. If the user's key is revoked, the server will reject ticket requests from this user.

Security administrator method: You can change a user's password so that a key can be generated using the CHANGE command with the NOPSWD-EXP option. For example:

```
Set lid
Change tlc089 password(mypass) nopswd-exp
```

User method: Users can change their own passwords, completing their own definitions as local principals, by using any standard eTrust CA-ACF2 password-change facility, such as the following:

```
TSO ACF command (Change command)
TSO LOGON
CICS LOGON
```

You may specify the level of kerberos key encryption support by indicating the KERBLVL(0|1) in the GSO OPTS records.

'0' indicates the DES kerberos key is generated for passwords. This is the default setting. The DES kerberos key is stored on the user's Logonid record.

'1' indicates the DES, DES3 and DESD kerberos keys will be generated for passwords. This level enables the use of the DES, DES3 and DESD encryption settings on the KERB USER PROFILE record.

The PASSWORD segment of the USER profile record is used to retain the DES3 and DESD kerberos keys when the GSO OPTS record specifies KERBLVL(1). Most fields in this record are not displayed. Users can list and delete, but cannot insert or change this record. It is maintained internally by eTrust CA-ACF2.

System Considerations for Key Generation

Your installation can share the eTrust CA-ACF2 databases with systems that support KERB User profile records and those that do not. You should not allow users to change their own passwords for key generation from systems that do not support the KERB User profile records. Keys will only be generated when users change their passwords from systems that support the KERB User profile records. Password changes made from systems that do not support the KERB User profile record will not generate keys to complete the user's local principal definitions.

Customizing your Foreign Environment

Defining Foreign Realms

You define foreign realms by creating REALM GSO records. The REALM field of the REALM GSO record contains the fully qualified name of both servers in the relationship. The REALM field uses the following format:

```
/.../realm_1/KRBTGT/realm_2
/.../Chicago.ca.com/krbtgt/newyork.ca.com
```

A foreign realm definition also contains a password. The password is not the same as the eTrust CA-ACF2 user password and is not constrained by the same rules. The password must be defined to create a trust relationship between the realms. For example:

```
Set Control(GSO)
Insert realm.newyork realm(/.../Chicago.ca.com/krbtgt/newyork.ca.com)
kerbpswd(abcdefg)
```

Mapping Foreign Principal Names

You map foreign principal names to eTrust CA-ACF2 users on your local system by defining KERBLINK User profile records. You can map each principal in a foreign realm to it's own logonid on the local system or you can map all principals from a foreign realm to the same logonid on your system.

eTrust CA-ACF2 users that map to foreign principals do not need KERB User profile records. These id's are intended to be used only to provide local eTrust CA-ACF2 identities to associate with access privileges for local resources that are under control of an application server, such as DB2.

KERBLINK User Profile Record

The KERBLINK USER profile record defines the foreign principals to a local node by linking it to a defined user. The eTrust CA-ACF2 user mapped to the foreign principal is defined in the recid of the record. KERBLINK profiles can define an individual principal from a foreign realm or all principals from a particular foreign realm. The local logonid does not need to have a KERB User Profile record associated with it.

The fields in the KERBLINK record

Recid
This is a 1 to 8-character userid that is to be associated with the foreign principal

Qualifier
A 1- to 8-character qualifier or suffix can be appended to the recid to create a unique record key. The suffix must be separated from the userid with a period.

KBLKNAME
Defines the foreign principal name. The foreign principal must be fully qualified with the name of the foreign realm. eTrust CA-ACF2 will verify that the KBLKNAME field starts with /…/ to ensure a valid field. The maximum length of this field is 240-characters. If you wish to map the same eTrust CA-ACF2 LID to all foreign principals in a foreign name, only specify the foreign realm name. The KBLKNAME field is folded to uppercase upon entry.

Command examples:

```
Set Profile(user) Div(kerblink)
insert tlc696 KBLKNAME(/.../chicago.tlc.com/FamousBear)
 KERBLINK / TLC696 LAST CHANGED BY TLC250 ON 04/02/02-13:53
                      KBLKNAME(/.../CHICAGO.TLC.COM/FAMOUSBEAR)

insert tlcpub KBLKNAME(/.../chicago.ca.com)
KERBLINK / TLCPUB LAST CHANGED BY TLC250 ON 04/02/02-13:55
                      KBLKNAME(/.../CHICAGO.TLC.COM)
```

The first example maps the foreign principal FamousBear from chicago.tlc.com to the logonid TLC696. The second example maps all other principals from chicago.tlc.com to the logonid TLCPUB.

Note: All characters in the KBLKNAME field are folded to uppercase.

Controlling Applications that Invoke the R_ticketserver Callable Service

Authorized applications, such as servers, can invoke the R_ticketserv callable service to extract principal names from a GSS-API context token. This enables an application server to determine the client principal who originated an application-specific request, when the request includes a GSS-API context token and the intended receipt is the application server. For detailed information about invoking the R_ticketserv callable service, see the IBM's *z/OS SecureWay Security Server RACF Callable Services* guide.

Applications that run in system key or in supervisor state do not require eTrust CA-ACF2 authorization to use the R_ticketserv callable service. Applications that do not run in system key or in supervisor state require READ access from eTrust CA-ACF2 to the IRR.RTICKETSERV Facility class resource. For example:

```
Set Resource

Compile
 .$KEY(IRR) TYPE(FAC)
 .RTICKETSERV UID(***USER1) SERVICE(READ) ALLOW

Store
```

In order for this rule to take effect you must rebuild the Facility class rules by issuing:

```
F ACF2,REBUILD(FAC)
```

HTTP Server

The HTTP server, originally known as the Lotus Domino Go Webserver (DGW), and also known as the z/OS Internet Connection Security Server, is an IBM component that lets the MVS mainframe act as an Internet web server. This component is installed and managed as a z/OS Unix System Services application.

By default, the HTTP server application files are installed in the OMVS environment in a directory named /usr/lpp/internet.

As documented by IBM, installation of the HTTP server involves several steps for the security administrator. The following information documents the commands necessary when installing this component under eTrust CA-ACF2 .

Prerequisites

To use the HTTP server in an eTrust CA-ACF2 environment, started task (STC) validation must be active.

Installation Steps

Follow these steps to install and implement the HTTP server:

1. Logonids must exist for the OMVS, INETD, and TCPIP started tasks. In addition, these logonids must be connected to an OMVS group record. The HTTP server also requires access to a group called TTY. The following examples reflect default started task (STC) procedure names, and a typical name and GID for the OMVS group ID. Overall, these commands ensure that a valid OMVS UID and GID exist for each of the started tasks necessary to access OMVS.

   Note: If z/OS Unix System Services has been previously set up, some or all of this step may already be complete.

   - To create the necessary logonids and profile records, issue these commands:

     ```
     ACF

     SET LID
     INSERT OMVS GROUP(OMVSGRP) STC UID(0)
     INSERT INETD GROUP(OMVSGRP) STC UID(0) HOME(/) OMVSPGM(/bin/sh)
     INSERT TCPIP GROUP(OMVSGRP) STC UID(0)
     ```

   - To define the necessary eTrust CA-ACF2 group profile records, issue these commands:

     ```
     SET PROFILE(GROUP) DIV(OMVS)
     INSERT OMVSGRP GID(nn)
     INSERT TTY GID(nn)

     END
     ```

2. The HTTP server requires logonids for the Web server started task and for a Web administrator. Both of these logonids must be connected to a Web server group ID. The commands below accomplish this and contain values matching IBM examples. Note that the Web server started task is also referred to as the Web server daemon. The Web server STC procedure name is WEBSRV.

   To create the necessary logonids and profile records for the Web server started task and the Web administrator, issue these commands:

   ```
   ACF

   SET LID
   INSERT WEBSRV NAME(WEBSERVER DAEMON) GROUP(IMWEB) STC UID(0)
   HOME(/usr/lpp/internet) OMVSPGM(/bin/sh)

   INSERT WEBADM NAME(WEB ADMINISTRATOR) GROUP(IMWEB) PASSWORD(password) UID(nn)
   HOME(/usr/lpp/internet) OMVSPGM(/bin/sh)

   SET PROFILE(GROUP) DIV(OMVS)
   INSERT IMWEB GID(205)

   END
   ```

Note: Since the WEBSRV logonid is accessed via a RACROUTE EXCTRACT request, the access information in this logonid is not updated. If your site monitors the access information to determine inactive logonids, please note this fact.

Note: If the WEBSRV logonid is given NO-SMC to improve the through put in the address space, you must ensure that the WEBSRV address space is never cancelled or forced from the system.

3. The IBM documentation discusses the probable need for several surrogate userids. Three suggested logonids and their associated group definitions are mentioned in the documentation. The names of the userids, groups, and the suggested UID and GID values are shown below:

```
ACF

SET LID
INSERT PUBLIC NAME(WEB SURROGATE ID) GROUP(EXTERNAL) RESTRICT UID(998)
HOME(/) OMVSPGM(/bin/sh)
INSERT INTERNAL NAME(WEB SURROGATE ID) GROUP(EMPLOYEE) RESTRICT UID(537)
HOME(/) OMVSPGM(/bin/sh)
INSERT PRIVATE NAME(WEB SURROGATE ID) GROUP(SPECIAL) RESTRICT UID(416)
HOME(/) OMVSPGM(/bin/sh)

SET PROFILE(GROUP) DIV(OMVS)
INSERT EXTERNAL GID(999)
INSERT EMPLOYEE GID(500)
INSERT SPECIAL GID(255)

END
```

4. The logonid for the Web server started task (daemon) requires access in both the FACILITY and SURROGAT resource classes. The following rules use the default type code values specified in the CLASMAPs shipped with eTrust CA-ACF2 . Be sure to verify that your site has not added local CLASMAP records to change these type code values.

   For the FACILITY resource class, add the following rule entries to the existing BPX FACILITY resource rule:

```
DAEMON UID(websrv) SERVICE(READ) ALLOW
DAEMON UID(inetd-lid) SERVICE(READ) ALLOW
SERVER UID(websrv) SERVICE(READ,UPDATE) ALLOW
```

   For the SURROGAT resource class, create a resource rule similar to the example below, using the following commands:

```
ACF

SET RESOURCE(SUR)

COMPILE
 $KEY(BPX) TYPE(SUR)
  SRV.INTERNAL UID(websrv) SERVICE(READ) ALLOW
  SRV.PRIVATE UID(websrv) SERVICE(READ) ALLOW
  SRV.PUBLIC UID(websrv) SERVICE(READ) ALLOW
  SRV.WEBADM UID(websrv) SERVICE(READ) ALLOW

STORE

END
```

Note: If the SURROGAT (TYPE SUR) rule already exists, add the necessary rule entries to it and recompile the rule.

If either the FAC or SUR rules are made resident via the GSO INFODIR record, issue the following commands, as appropriate:

```
F ACF2,REBUILD(FAC)
F ACF2,REBUILD(SUR)
```

5. Users need access to web server related libraries. See the appropriate IBM documentation and/or site naming conventions for the names of the data sets.
6. Install steps that discuss permission bits and the "sticky bit" are related to OMVS file security itself. Follow these steps as described.
7. Your site can run web server with optional functions. The first is running the web server in scalable server mode to support WLM and the other is allowing the web server access to SMF.

   To support WLM, the web sever needs access to the FACILITY resources BPX.WLMSERVER and MVSADMIN.WLM.POLICY. Do the following in eTrust CA-ACF2 to enable this:

   Add the following rule entry to the BPX FACILITY resource rule:

   ```
   WLMSERVER UID(websrv_uid) SERVICE(READ) ALLOW
   ```

   Create the following MVSADMID FACILITY rule:

   ```
   SET RESOURCE(FAC)
   COMPILE
   $KEY(MVSADMIN) TYPE(FAC)
   WLM.POLICY UID(webserv_uid) SERVICE(READ,UPDATE) ALLOW

   STORE
   ```

   To support access to SMF, add the following rule entry to the BPX FACILITY resource rule:

   ```
   SMF UID(websrv_uid) SERVICE(READ) ALLOW
   ```

   If the FACILITY class resources were made resident via the GSO INFODIR record, enter the following command to activate any changes:

   ```
   F ACF2,REBUID(FAC)
   ```

WebSphere Application Server for z/OS

The WebSphere Application Server for z/OS is an IBM component that lets the MVS mainframe act as an Internet Web server. This component is installed and managed as a z/OS Unix System Services application. WebSphere for z/OS supports access to resources by clients and servers in a distributed network. Part of your security strategy should be to determine how to control access to these resources and prevent inadvertent or malicious destruction of the system or data.

These are the pieces in the distributed network that you must consider:

- You must authorize servers to the base operating system services in z/OS. These services include eTrust CA-ACF2 , database management, and transaction management.
- For the servers, you must distinguish between control regions and server regions. Control regions run authorized system code, so they are trusted. Server regions run application code and are given access to resources, so you should carefully consider the authorizations you give server regions.
- You must also distinguish between the level of authority given to run-time servers compared to your own application servers. For example, the System Management server needs the authority to start other servers, while your own application servers do not need this authority.
- You must authorize clients (users) to servers and objects within servers. The characteristics of each client requires special consideration:
- Is the client on the local system or is it remote? The security of the network becomes a consideration for remote clients.
- Will you allow unidentified (unauthenticated) clients to access the system? Some resources on your system might be intended for public access, while others must be protected. In order to access protected resources, clients must establish their identities and have authorization to use those resources.
- What kind of objects will the client access? Enterprise beans and CORBA objects have differing authorization mechanisms.

If you must protect resources, identifying who accesses those resources is critical. Thus, any security system requires client (user) identification, also known as authentication. In a distributed network supported by WebSphere for z/OS, clients can be accessing resources from:

1. Within the same system as a server
2. Within the same sysplex as the server
3. Remote z/OS systems
4. Heterogeneous systems, such as WebSphere on distributed platforms, CICS, or other CORBA-compliant systems.

Additionally, clients can request a service that requires a server to forward the request to another server. In such cases the system must handle delegation, the availability of the client identity for use by intermediate servers and target servers.

Finally, in a distributed network, how do you ensure that messages being passed are confidential and have not been tampered? How do you ensure that clients are who they claim to be? How do you map network identities to z/OS identities? These issues are addressed by the following support in WebSphere for z/OS:

1. The use of SSL and digital certificates
2. Kerberos
3. Distributed Computing Environment (DCE)

Network security is not required for your initial installation and customization of WebSphere for z/OS. This information is provided to introduce you to WebSphere for z/OS security and allow you to make early planning decisions about system security. The following topics describe how WebSphere for z/OS supports security. The descriptions are organized under the following subtopics:

- Authorization Checking
- User Identification, Authentication and Network Security

Authorization Checking

Each control region, server region, and client must have its own eTrust CA-ACF2 logonid (more about user identification and authentication later). When a request flows from a client to the server or from a server to a server, WebSphere for z/OS passes the user identity (client or server) with the request. Thus each request is performed on behalf of the user identity and the system checks to see if the user identity has the authority to make such a request.

The following is an outline of the set up requirements for WebSphere. See ACFCSEC member which can be found in the eTrust CA-ACF2 SAMPJCL library (the number below refers to the steps in ACFCSEC).

1. Define the basic logonids and group profiles.
2. Define the IVP1 logonids and group profiles.
3. Define the IVP2 logonids and group profiles.
4. Define server access to the FACILITY BPX resources.
5. Define the LOGSTRM class permissions for logonids writing to the error log stream.
6. Define CBIND class resources access to allow a client to bind to a control region.

7. Define access to the OPERCMDS class resources necessary to start and stop server jobs.
8. Define access to the SOMDOBJS class resources to allow access to installation-defined methods.
9. Define access to the EJBROLES class resources to allow access to installation-defined methods.
10. Define access to the FACILITY class resource IRR.RDCEUID for DCE and IRR.RUSERMAP for KERBEROS. For information regarding DCE or KERBEROS set up see the Defining DCE under eTrust CA-ACF2 and Kerberos section in this guide.
11. Define access to the FACILITY class resource IMSXCF.OTMACI for each server permitted to use the OTMA callable interface.
12. Define access to the SURROGAT class resources logonid.DFHEXCI to servers using CICS EXCI PAALRM.
13. Define server SSL controls to create the certificates, keyrings, and rules needed to support SSL connections between servers.
14. Define client SSL controls to create the IVP client certificates and keyrings needed to run the clients using an SSL connection.

Level of Trust and Authority for Regions

Region	Level of Trust and Access Authority
Control region	Contains WebSphere for z/OS system code.Trusted, deals with multiple users. Greater authorization. Runs APF-authorized.
Server region	Contains application code. Untrusted. Other than having authorization to get work and to attach to data stores, should run unauthorized.

Regarding the WebSphere for z/OS run-time servers, the rule of thumb is to give less authority to the Daemon and Naming Server, and greater authority to the System Management Server, as explained in the following table:

Run-time Server	Region	Required Authorities
Daemon Server	Control	STC, access WLM services, access to DNS, OPERCMDS access to START, STOP, CANCEL, FORCE & MODIFY other services
Naming Server	Control	STC, access to WLM services
Naming Server	Server	STC, READ auth to the SERVER class, DBADM for the LDAP Database
Sys. Mgmt. Server	Control	STC
Sys. Mgmt. Server	Server	STC, Read auth. To the SERVER class, OPERCMDS access to START, STOP, CANCEL FORCE, and MODIFY other services
Interface Repository	Control	STC
Interface Repository	Server	STC, READ Auth. To the SERVER Class, DBADM for the LDAP database

Assigning authorities to WebSphere for z/OS run-time server control and server regions

- Remember to protect the RRS log streams.
- Protect the WebSphere for z/OS environment files, especially if they contain passwords.

User Identification, Authentication and Network Security

LDAP Access Control Lists (ACLs)

LDAP uses access control lists to control client access to naming services. Usually, you set up a general ANYBODY user identity with read access to the LDAP name space, allowing any client to access naming services.

CBIND Class

You can use the CBIND class to restrict a client's ability to access servers. There are two types of resources that WebSphere for z/OS uses in the CBIND class:

- One that controls whether a local or remote client can access servers. The name of the resource has this form:

  ```
  CB.BIND.server_name
  ```

  where *server_name* is the name of the server.

- One that controls whether a client can use objects in a server. The name of the resource has this form:

  ```
  CB.server_name
  ```

  where *server_name* is the name of the server.

  Note: When you add a new server, you must authorize all systems management logonids (for example, CBADMIN) to have read access to the CB.**server_name** and

  ```
  CB.BIND.server_name resources.
  ```

  Example: To allow CBADMIN needs read authority to the CB.BBOASR1 and CB.BIND.BBOASR1 resources, add the following to the CBIND class CB resource rule:

  ```
  BBOASR1 UID(cbadmin_uid) SERVICE(READ) ALLOW
  BIND.BBOASR1 UID(cbadmin_uid) SERVICE(READ) ALLOW
  ```

  The CBIND class defaults to a generic type code of SAF. It is recommended that a GSO CLASMAP record be added to change this to a site selected resource unique to the CBIND class such as CBI. The following shows how the suggested change example would be coded:

  ```
  SET CONTROL(GSO)
  INSERT CLASMAP.cbind RESOURCE(CBIND) RSRCTYPE(cbi) ENTITYLN(41)
  ```

  To activate the change immediately, issue the following command:

  ```
  F, ACF2,REFRESH(CLASMAP)
  ```

EJBROLE and GEJBROLE Classes

Access to an enterprise bean can be controlled through security roles that are the group of permissions that a user must have to successfully use an application. Using a Java method called *isCallerInRole,* the application programmer or systems administrator specifies the security role names that are allowed to access a particular bean. Only users having access to these security role names are granted access to the bean.

Execution of *isCallerInRole* in a z/OS environment causes invokes IRRPNL00 profile name list service routine. This routine returns a list of all of the Java security roles that a user has access to.

eTrust CA-ACF2 fully supports this use of the IRRPNL00 routine through the use of EJB generalized resource rules. (By default, the internal CLASMAP record maps the EJBROLE resource class to an EJB type code.) When SAF processing passes the IRRPNL00 routine request to external security, eTrust CA-ACF2 processes each EJB resource rule to determine if the specified user has access to each defined role. After processing all of the EJB rules, eTrust CA-ACF2 passes a list of all allowed roles for use by the Java *isCallerInRole* method, which then allows or prevents access to the bean based on this list.

To set up the external security environment, do the following:

1. Identify the security roles specified by your application programmers or OEM/ISV providers and the users that should have access to each role. The role name used in the EJB resource rules is the security role specified in the jar file or for the application. It can be up to 256 characters in length and can contain mixed case characters but can not contain blanks.
2. Write and store EJB resource rules for each security role. There are some special guidelines that apply to EJBROLE resource rules as follows:
   - Since the purpose of the EJB rules is to provide a specific list of allowable roles for each user, each defined role in your environment must have a specific rule for it. This means that masking is not allowed in the resource name of an EJB rule.
   - EJBROLE names use mixed case names. The eTrust CA-ACF2 resource rules now support mixed case resource names. To identify this, the resource class is designated as mixed by using the MIXED operand in the appropriate CLASMAP record. The SHOW CLASMAP output now indicates that MIXED operand is set on by placing a YES in the column labeled MIXED. Once the MIXED keyword is set on, you can use mixed case in the $KEY, the $PREFIX, the $USERDATA, the resource name, and the NEXTKEY parameter. Ensure that the administrative platform used for processing the EJB rules supports the entry and display of mixed case.
   - EJBROLE processing can be used to retrieve data for use in an application. To access the APPLDATA, the application performs a SAF EXTRACT call. In an eTrust CA-ACF2 environment, the APPLDATA is stored in the $USERDATA control card of the EJBROLE resource rule. The value placed in the $USERDATA is returned as APPLDATA in response to the EXTRACT call. The value in the $USERDATA can be mixed case.
3. EJB resource rules must be globally resident by means of a resident directory so that eTrust CA-ACF2 can process them. To accomplish this, enter the following commands:

```
SET CONTROL(GSO)
CHANGE INFODIR TYPES(R-REJB)
```

4. When a resident resource rule is created or changed, the directory for it must be rebuilt through the following command:

```
F ACF2,REBUILD(EJB)
```

5. Example: The ACCOUNT bean is an *isCallerInRole* Java method coded to allow access only by those users having access to the *accounting.clerk* and the *accounting.manager* security roles. The following resource rule might be written to accomplish this:

```
COMPILE
$KEY(ACCOUNTING) TYPE(EJB)
CLERK UID(uid_string_clerk) ALLOW
MANAGER UID(uid_string_manager) ALLOW

END
STORE
```

For additional information on the use of EJBROLE security in WebSphere Application Server environment, consult:

WebSphere Application Server for z/OS: Installation and Customization

WebSphere Application Server for z/OS: Writing Enterprise Beans in WebSphere.

SOMDOBJS Class

The application assembler must assign method permissions to the bean or method using the Application Assembly Tool.

- Define the roles relevant to the application.
- Once defined, the role can be assigned to access an application (as a method permission).
- After the application assembly is complete, the application must be reinstalled using the Administration application.

Use the SOMDOBJS class in eTrust CA-ACF2 to control a client's access to CORBA objects. Resource names in SOMDOBJS have the form:

```
server_name.home.method
```

Where **server_name** is the server name. It must be 8 characters or less. **home** is the home name. It must be 192 characters or less. **method** is the method name. It can be up to the length of the remainder of 244 minus the sum of the server and home name lengths.

Example: If the server name is 8 characters, and the home name is128 characters, the method name can be 108 (244 - (8 + 128)). If a method is protected by SOMDOBJS, a client must have READ or UPDATE authority depending on the type of access being attempted. If a client program is using the method to update an attribute of an object, give the client UPDATE authorization for the method; if a client program is using the method to read an attribute of an object, give the client READ authorization for the method. All names are folded into uppercase characters, regardless of how you enter them. Thus, there is no difference between MY_server.MY_home.MY_method and MY_SERVER.MY_HOME.MY_METHOD.

In addition to the SOMDOBJS definitions, you must specify method-level access checking through the WebSphere for z/OS Administration application. Check the box for method-level access checking when you define your application.s container.

Resource Managers

Resource managers such as DB2, IMS, and CICS have implemented their own resource controls, which control the ability of clients to access resources.

When resource controls are used by DB2, use eTrust CA-ACF2 for DB2 or issue the relevant DB2 GRANT statements.

Access to OTMA for IMS access is through the FACility class resource IMSXCF.OTMACI.

Access to EXCI for CICS is through the SURROGAT class resource logonid.DFHEXCI.

You can control access to data sets through the eTrust CA-ACF2 access rules class and HFS files through file permissions or the HFS resource rules.

Protection and Protect Directives

The sample config file provided for WebSphere for z/OS includes two elements that have similar names that relate to how security is implemented for files. One is called Protection directives and the other is called Protect directives. Protection directives tell the HTTP Server how to secure a particular file. The following example is for a protection directive called PROTECTED_INFO and indicates, among other things, that a certificate is required when accessing files protected with this directive.

```
Protection PROTECTED_INFO {
ServerId Security_Administration
AuthType Basic
PasswdFile %%SAF%%
Userid %%CERTIF%%
Mask anybody
}
```

Protect directives associate a file or group of files with a specific Protection directive. The following example indicates anyone accessing files in the /secret directory through the Http Server will fall under the PROTECTED_INFO Protection directive and must supply a certificate that is trusted by ACF2.

```
Protect /secret/* PROTECTED_INFO
```

Protect directives identify specific directories or files that are accessed by different users than the default. The default Protection directive is PUBLIC. For more information regarding this server configuration see the *HTTP Server Planning, Installing and Using Guide*.

Prerequisites

To use the WebSphere application server in an eTrust CA-ACF2 environment, started task (STC) validation must be active.

Installation Steps

WebSphere for z/OS documentation points sites to a utility called BBOCBRAC that can be used to generate the set up required for external security but only assumes a RACF environment. eTrust CA-ACF2 support has put together an eTrust CA-ACF2 equivalent batch job. Careful review of the job is necessary because the local site must supply variables unique to their environment as well as the fact that some of the required updates may already be in place by your site. See the ACFCSEC member in the eTrust CA-ACF2 SAMPJCL data set for the actual JCL.

Chapter 22

Controlling Access to the Hierarchical File System

With eTrust CA-ACF2 Security, there are two processes that a site can use to secure the Hierarchical File System (HFS). The first process is internal to z/OS Unix System Services and is based on a UNIX model of security. The second process is external security and uses standard eTrust CA-ACF2 security rules to secure the HFS. These processes are mutually exclusive, so your site must select which one to use.

Accessing an HFS Data Set from MVS

If you attempt to access a MVS data set that represents a hierarchical file system (HFS), through ISPF 3.2 or 3.4, it is possible that you will get an "OBTAIN failed" message. The extended message reads:

"datasetname has unknown attributes, OBTAIN RC = 12 hex".

This occurs if the HFS data set is not mounted to OMVS.

When data set information is requested for an unmounted HFS data set, z/OS UNIX System Services will write information to the /tmp directory. If the user making the request does not have write access, the error message is displayed.

To avoid this error, you must ensure that the public access permission for the /tmp directory is set to allow all access. It is suggested that the permission bits for the /tmp directory be set to 777 to allow all access.

Note: Using UNIX System Services security with eTrust CA-ACF2 still requires that access be allowed using resource rules to the /tmp directory.

Controlling HFS Using the UNIX Security Model

z/OS Unix System Services files are organized in a hierarchy as in a UNIX system. All files are members of the directory. Each directory is a member of another directory at a higher level of the hierarchy. The highest level of the hierarchy is the root directory.

Security for the file system directories and files is based on a UNIX model of security. Each file and directory is assigned an owning UID and an owning GID. This assignment is defined and saved in the file system, not in the external security product.

Normally each file or directory saves the access permissions in the form of four octal numbers nnnn. The first position represents special access flags while the remaining three are the permission categories. The access flags include the sticky bit, the setuid on execution, and the setgid on execution.

The other three categories of users can access each directory and file in the HFS. They are:

- The user that owns the file
- The group that owns the file
- All other users defined to z/OS Unix System Services

Three different access levels (READ, WRITE, and EXECUTE) can be set for any of these three categories. For example, permissions can be defined so that the file owner gets READ and WRITE access, a member of the file's group gets only READ access, and all other users get neither READ nor WRITE access.

Under eTrust CA-ACF2, you must define a UID for each z/OS Unix System Services user and a GID for each group that accesses z/OS Unix System Services. You must also assign a default group in all z/OS Unix System Services userids and give the users access to any supplemental groups needed.

For more information about the Hierarchical File System and setting file permissions, see the following IBM guides:

- *z/OS V1R2.0 UNIX System Services User's Guide*
- *z/OS V1R2.0 UNIX System Services Planning*

Processes that Affect HFS Security

When using the UNIX security model, various options can affect the file validation process. The processes and their effect on file security or validation are described in this section.

Access Control Lists

Access Control Lists (ACLs) provide more granular control over the HFS file system than native HFS security. To activate the use of ACLs in the valiation process, the only requirement is to specify HFSACL in the GSO UNIXOPTS record. By default, ACLs are not active (NOHFSACL)

To activate ACLs, use the following ACF command:

```
SET CONTROL(GSO)
Change unixopts hfsacl
```

To activate the change, use the following Operator command:

```
F ACF2,REFRESH(UNIXOPTS)
```

ACLs are created, deleted, and maintained by using the z/OS Unix System Services command setfacl. Existing ACLs can be viewed by using the getfacl z/OS Unix System Services command. See the IBM z/OS 1.3 *USS Command Reference* manual for more information about these commands. ACLs can be created and maintained even if they are not active (UNIXOPTS NOHFSACL). In order for a user to issue the setfacl command they must be one of the following:

1. Owner of the file or directory
2. Superuser
3. Read access to UNIXPRIV resource SUPERUSER.FILESYS.CHANGEPERMS.

HFS FASTPATH Checking

OMVS issues a SAF call at initialization. The SAF call checks to see if the BPX.SAFFASTPATH FACILITY class profile is defined. If the profile is defined, OMVS performs permission bit checking internally instead of calling the external security manager, bypassing any audit trail of violations. This is referred to as FASTPATH processing.

To disable FASTPATH processing you must insert the following SAFDEF record:

```
INSERT SAFDEF.OEFSTART FUNCRET(4) ID(OEFSTAUT) JOBNAME(OMVS) MODE(IGNORE) -
    RB(BPX-) RACROUTE(REQUEST=AUTH CLASS=FACILITY ENTITY=BPX.SAFFASTPATH) REP
```

MOUNT NOSECURITY

You now have the option to MOUNT a file system or part of a file system with or without SECURITY. The use of the MOUNT command requires superuser authority. If the file system is mounted with the NOSECURITY option, access checks are made by OMVS using system credentials instead of passing user credentials. OMVS treats system credentials like superusers so access will always be allowed.

There are no indications in the mount messages that a file was mounted with the NOSECURITY option. Always review the BPXPRMxx member of parmlib to ensure that none of the file systems are being mounted NOSECURITY. Also, restrict access to the MOUNT command itself to minimize the potential for this being done.

Change Owner Command (CHOWN)

The CHOWN command lets the owner or a superuser change a file's owner. In eTrust CA-ACF2, the GSO UNIXOPTS record CHOWNRES parameter controls who can issue this command. CHOWNRES implements POSIX CHOWN RESTRICTED, which states that only a superuser can modify the owner UID of a file. If the option is turned off (NOCHOWNRES), it implements POSIX CHOWN UNRESTRICTED, which lets the current owner modify the owning UID of a file.

Audit

A logonid record with the AUDIT privilege is allowed search access and read access to directories in HFS. This allows audit option bits to be set on any file in HFS without requiring that all directories have search permissions allowing the auditor access.

Program Control in the UNIX Environment

When the BPX.DAEMON and BPX.SERVER facilities are active, processing authorized functions, such as SETUID, requires that programs or executables be loaded from an authorized library. In an eTrust CA-ACF2 environment, these authorized data sets are any library in the LPA list, the APF list, the LINKLIST if the LINKLIST has been designated as APF authorized, or the GSO LINKLIST. If a program is loaded from the HFS or an MVS data set not on the approved lists, the TCBNCTL flag, referred to as the "dirty bit," is set. This results in authorized functions failing if attempted in the "dirty" environment.

The TCBNCTL flag is only set in the eTrust CA-ACF2 environment if program control is activated. By default, this process is not active. To activate it, you must override the PROGMCHK SAFDEF, which is set to ignore. To override this SAFDEF, create the following SAFDEF:

```
SET CONTROL(GSO)
INSERT SAFDEF.PROGMCHK ID(PROGMCHK) MODE(GLOBAL) REP -
       RACROUTE(REQUEST=FASTAUTH REQSTOR=PROGMCHK SUBSYS=CONTENTS)
```

This SAFDEF activates validation for each program fetched. Before activating this SAFDEF, you must ensure that appropriate program rules are in place and that these rules are made resident through the INFODIR record. You can accomplish this by doing the following:

1. By default, the PROGRAM class resources are validated under a type code of PGM. If you wish to use a different type code, enter the following CLASMAP record:

   ```
   SET CONTROL(GSO)
   INSERT CLASMAP.PROGRAM RESOURCE(PROGRAM) ENTITYLN(8) TYPE(typecode)
   ```

2. The validation check for the PROGRAM class is done using a FASTAUTH call. This type of call requires that the resource rules be resident. If not already included in the INFODIR record, add it to this record by entering the following:

   ```
   SET CONTROL(GSO)
   CHANGE INFODIR TYPES(R-RPGM)
   ```

   Note: If you changed the default type code of PGM to another type code, replace the PGM in the above example with the type code that you assigned to the PROGRAM resource class.

3. Create the PGM type resource rules required to let users access the programs they are using. If you want to allow access to any program and set the environment similar to the one before program checking is activated, you can enter the following resource rule:

   ```
   SET RESOURCE(PGM)
   COMPILE *
   $KEY(********) TYPE(PGM)
   UID(*) ALLOW
   ```

   If you wish more specific controls, then write a rule for each program in your environment.

4. To activate all of the previously described steps, enter the following operator commands:

   ```
   F ACF2,REFRESH(ALL)
   F ACF2,REBUILD(PGM)
   ```

When an executable or program is requested in an OMVS environment, OMVS finds the executable in the HFS and loads it from there unless the "sticky bit" is turned on. If the sticky bit is set on for the executable file, then OMVS uses normal MVS load processing. To turn the sticky bit on using the OMVS chmod command, a user must own the file or be a superuser.

If an executable or a program is to be loaded directly from HFS then the "Program" extended attribute has to be set for the file in order for it to be considered a controlled program. This can be accomplished by using the OMVS extattr command, however, use of this command does require access to the BPX.FILEATTR.PROGCTL resource. To set up the rule to allows this, add the following ruleline to the BPX FACility resource rule:

```
FILEATTR.PROGCTL UID(chmod_user) ALLOW
```

Controlling HFS using CA SAF HFS Security

z/OS brings the MVS and UNIX operating systems together onto one hardware platform. Although some interoperability between MVS and UNIX exist, each environment retains its own distinct data structures and methods of access control.

UNIX data is kept in a Hierarchical File System (HFS). From the UNIX perspective, the HFS contains many discrete data files. From the MVS perspective, the HFS is one data set and can only be controlled as one data set. In other words, MVS can control access to the entire file system, but not to the individual files within the HFS.

HFS files are protected by file permission bit settings. These are set when the file owner creates the file. Centralized administration can only be performed by a superuser, a user privilege that grants much more authority than just security administration. z/OS resources are protected by access and resource rules, which are usually set up in advance by security administrators. Security administrators can be scoped in a decentralized environment.

CA SAF HFS security overcomes the shortcomings of native UNIX security by providing single-point security access control, administration, and reporting for both MVS and UNIX resources. CA ENF services present access events to eTrust CA-ACF2 for validation. Administrators use familiar commands and rules to protect UNIX files and functions, restricting access based upon the eTrust CA-ACF2 UID-string instead of the UNIX UID or GID numbers. HFS access loggings and violations are reported in the standard eTrust CA-ACF2 reports.

The following sections explain:

- HFS File Access Security using resource rules
- Securing HFS (system and file)Implementation
- The CA SAF HFS Rule Generation Utility
- The CA SAF HFS Security Modification Utility

File Access Security

When using CA SAF HFS security, native file permission bit security is bypassed, as well as the superuser authority to access any file. File access is validated by eTrust CA-ACF2 security using resource rules. All the benefits of resource rules can be utilized, including masking, NEXTKEY, scoping, %CHANGE, and reporting. Certain extensions are available that allow user directories to be defined and to allow users to maintain rules for their own files.

Path Name Translation

The structure of HFS path names presents a challenge to external security products. A path name can be up to 1023 characters in length, except when used in the JCL PATH= keyword where the limit is 255 characters. The path name is also case-sensitive and can contain special characters. To allow external security to validate HFS files, certain manipulation of path names is required. The benefits of having enhanced security and single-point administration certainly make file name translation acceptable.

Before validation, all path names are truncated, if necessary, to 255 characters. An exit point (HFSEXIT) is provided for use when file names reside in paths that are greater than 255 characters. Your site can use the exit to provide a meaningful name. See Exit Processing for more information.

Path names are converted to upper case unless your site inserts a GSO CLASMAP record for the HFSSEC class and specifies the MIXED keyword to indicate that mixed case resource names are to be used. Use the SHOW CLASMAP subcommand of the ACF command to determine if the HFSSEC class specifies MIXED. See the "Maintaining Global System Options Records" chapter for more information on the CLASMAP GSO record and MIXED keyword.

eTrust CA-ACF2 resource rule processing considers the period character as a delimiter. This delimiter is used when writing extended resource rules, that is, to provide security for resource names of greater than forty characters. Path names, however, use the slash character as a delimiter. Before a file is validated, the path name will have all slash characters, with the exception of the first, translated into a period delimiter. Other special characters will be translated into the dollar sign ($). These include characters that are used as masking characters in resource rules. If not translated, these characters could create undesired results. The special characters include the period, asterisk, dash, plus, blank, and quote. An exit point is provided that can further modify any character to meet special needs, with the exception of the slash character, which will always be translated to a period delimiter.

eTrust CA-ACF2 represents HFS path names as qualified resource names. One of the requirements of qualified resource names is that the first qualifier must be 1-40 bytes in length. If, after translation, the HFS path name does not contain a period within the first 41 bytes, the path name translation capability of the exit can be used to provide a meaningful first level qualifier.

Some examples of path name translation follow:

Original Path Name	Security Action
/bin/su	Control access to switch user command.
/u/user01/proj1/file1.txt	Define rule set for user01.
/usr/sbin/mknod	Allow system programmers to create character special files.

Translated Path Name	Sample resource rules
/BIN.SU	$KEY(/BIN) TYPE(HFS) SU UID(sysprog) ALLOW
/U.USER01.PROJ1.FILE1$TXT	$KEY(/U) TYPE(HFS) USER01.- UID(usera) ALLOW
/USR.SBIN.MKNOD	$KEY(/USR) TYPE(HFS) SBIN.MKNOD UID(sysprog) ALLOW

Seteuid Permission Bit Programs

In normal HFS, when a program or executable file with the seteuid permission bit on is executed, the seteuid process changes the UID from the actual user to the UID of the owner of the file. This allows users to access files under different permissions from their own.

With eTrust CA-ACF2 HFS processing, access to a file is done based on the UID of the person accessing the file. However, due to the seteuid processing described above, the eTrust CA-ACF2 support still allows for this. When eTrust CA-ACF2 HFS support recognizes that the seteuid bit is on, it checks to see if the effective UID is equal to the UID of the file owner. If they match, access is allowed. If they do not match, the normal resource validation is done.

Symbolic Links

When an actual file is accessed via a symbolic link, the file name passed to eTrust CA-ACF2 has been resolved to the actual file name of the file and validation is done based on that actual file name. For most validation situations this is sufficient. However, in the case of the following, eTrust CA-ACF2 validates the link itself rather than the file name:

```
unlink (delete a symbolic link)
rename (change one symbolic link name to another)
readlink (read a symbolic link to determine what it points to)
lchown (change the owner of a link)
```

Shared HFS

HFS files on one system can be accessed by other systems within the sysplex. For more information and details about a shared HFS, see the z/OS UNIX System Services Planning Guide.

There are implications to eTrust CA-ACF2 HFS rule writing inherent with the way z/OS Unix System Services specifies the path name in a shared HFS sysplex. The path name now contains the system name of the system that owns the file. For example, referring the examples of path name translation above, access to the su command on the system XE77 would have an original path name of /XE77/bin/su. This would be translated to a path name for eTrust CA-ACF2 purposes of /XE77.BIN.SU. This gives us a resource rule similar to the following:

```
$KEY(/XE77) TYPE(HFS)
BIN.SU UID(sysprog) ALLOW
```

User File Ownership

Another consideration of HFS file validation is how user files are validated. User files are those files that are below a directory entry representing a specific user. CA SAF HFS security provides the ability for users to maintain their own resource rules, can generate resource names that can be identified as existing in a user directory, and can bypass validation for user access to files within the user's own directory.

Ownership of MVS data sets is identified by use of the data set high level qualifier. In a decentralized security environment, users can create rules to protect their own data sets. This concept has been carried over into HFS file security. When GSO RULEOPTS specifies the NOCENTRAL option, users are able to maintain and store their own HFS resource rules. Ownership is identified in the $KEY by the presence of the userid prefixed by '$$'. This prefixed userid must be the only value within the $KEY. For example, USER01 is able to maintain the HFS resource rule with $KEY($$USER01). The userid must be defined as an OMVS user.

Although validation can be directed to a $$userid rule using NEXTKEY, as shown in the example in the previous section, Path Name Translation, another facility is available that automatically translates the rule to the $$userid format at validation time. This facility can be used if all user directories are anchored at the same location in the file system. An installation exit defines this location to CA SAF HFS security as the user directory mount point. A common location for user directories to be anchored is at the /u/ mount point. If this is the case, expanding upon the previous example, path name /u/user01/proj1/file1.txt is translated to $$USER01.PROJ1.FILE1$TXT. To implement this facility, use an exit as described later in this chapter in the Implementation section. Even if user directories are not anchored in one central location, the exit can be used to create the $$userid format of the resource at validation time. By default, no user directory path is recognized and the resource is not translated into the $$userid format.

Another available option is the ability for users to access files that reside in their own user directory without incurring a validation for that file. In other words, users are always able to access their own files. This option requires that a user directory anchor point be defined through use of the installation exit. The exit returns an indicator stating that the file ownership option is active. Expanding further upon the example, if this option were active, validation is bypassed when USER01 accesses /u/user01/proj1/file1.txt. By default, users do not automatically have access to their own files.

Rule Considerations

This section describes special considerations to be taken into account when writing rules for HFS resources.

In addition to access to HFS files, users might need access to directories. A user requires READ access to a directory to list the contents of that directory. When writing a rule set, you distinguish a rule line protecting the directory from rule lines protecting the files within the directory by not using an extended key in the rule line. For example, the rule used to allow users to read the /BIN directory, but only allow EXECUTE access to the files contained within the directory is:

```
$KEY(/) T(HFS)
UID(-) SERVICE(READ,EXECUTE) ALLOW
– UID(-) SERVICE(EXECUTE) ALLOW
```

Files contained in the root directory must be specified as the $KEY value in a separate rule set, that is, they cannot be specified as an extended rule line within the $KEY(/) root rule set. Therefore, the only valid rule line for the $KEY(/) root rule set is that which allows read access to the directory itself. The following shows the rule set required for the root directory and a sample rule set allowing read and write access to file /rootfile:

```
$KEY(/) T(HFS)
UID(-) SERVICE(READ) ALLOW

$KEY(/ROOTFILE) T(HFS)
UID(-) SERVICE(READ,UPDATE) ALLOW
```

Rules written to secure HFS file resources should specify the SERVICE keyword to identify the type of access to the file. If the SERVICE keyword is not used, **all** access is implied. The SERVICE keywords are:

SERVICE Keyword	Description
EXECUTE	Allows execute access to a file, usually a program file.
READ	Allows read access to a file.
UPDATE	Allows write access to a file.
ADD	Allows the ability to create and delete a file.
DELETE	A special access not used for normal file access validation. This is used with HFS function security to allow a user to change file attributes. See File Functions later for more information.

CA SAF HFS security uses fast-path resource validation. Because of this, the HFS resource rules must be defined as resident through use of the GSO INFODIR record. Also, the eTrust CA-ACF2 Resource Prevalidation(RSCXIT1) and Resource Postvalidation (RSCX172) exits are not processed for HFS file validation.

Reporting

Auditing records created by HFS file access, that is, violation, trace and logging records, are accessed through the same facilities as all other resource records, namely ACFRPTRV. In addition to all the standard items reported, the original, unmodified path name, up to 256 characters, is reported. If using your own reporting, the original path name can be found in SMF record field ACVMFXKY.

The access service can be used by z/OS Unix System Services to pre-determine whether a process has access to a particular file or directory. The process in question is not attempting to access the file or directory; rather, the z/OS Unix System Services system is simply checking to see if the process is capable of accessing the file or directory. When the access service is used in this manner, eTrust CA-ACF2 processing does not generate SMF records for ACFRPTRV.

Securing HFS Functions

In addition to file access security, HFS functions can also be secured. These functions can be a system action, such as setting a ptrace or a job's priority, or they can be file-related, such as changing the file mode or audit settings.

A system function is secured by a rule in the FACILITY class, while a file-related function is secured by a combination of a FACILITY class rule and a HFS file resource rule. By following this approach, changes to file attributes can be permitted at a global basis, or restricted to a particular file.

The resource name format for HFS FACILITY rules is:

```
BPX.CAHFS.function
```

An example of a rule would be:

```
$KEY(BPX) TYPE(FAC)
CAHFS.function UID(user) ALLOW
```

Values for *function* are listed in the next section, System Functions.

System Functions

To perform a system function, the user requires READ access to the corresponding FACILITY rule:

BPX.CAHFS.CHANGE.FILE.MODE
Allows a user to change any file mode information. This includes changes to file permission settings, setting the execution UID or GID indicators, setting the "sticky" bit, and maintaining Access Control Lists. Native z/OS UNIX permission settings are used for validation purposes only when CA SAF HFS is inactive.

BPX.CAHFS.CHANGE.PRIORITY
Lets a user change the scheduling priority of a process, process group, or user. z/OS UNIX System Services requires that the user be a superuser to use this function.

BPX.CAHFS.SET.PRIORITY
Lets a user set the scheduling priority of a process, process group, or user. z/OS UNIX System Services requires that the user be a superuser to use this function.

BPX.CAHFS.SET.RLIMIT
Lets a user set the resource limit for the calling process.

BPX.CAHFS.MOUNT
Lets a user mount file systems. z/OS UNIX System Services requires that the user be a superuser to use this function.

BPX.CAHFS.UNMOUNT
Lets a user remove a virtual file system. z/OS UNIX System Services requires that the user be a superuser to use this function.

BPX.CAHFS.PTRACE
Lets a user control and debug another process. Although the user need not be defined as a superuser to use this function, access to this resource does not give the user any more authority than a superuser would have. Access to the function will be denied if the user attempts to debug a program running with SETUID or SETGID, that is, a program that switches user identification.

BPX.CAHFS.CREATE.LINK
Lets a user create a hard link to an existing file. A hard link is essentially another name for the same file data. If the original file is removed, the hard link still points to the file data. The data is not deleted until the last link is removed. In addition to this resource, the user also requires SERVICE(ADD) access to the HFS file resource rule for both the original file and the link file.

Important! *When data associated with a hard link is accessed, the CA ENF/USS service requests the file name from z/OS UNIX Services. The file name returned might be the hard link name or the original file name regardless of the actual path accessed. It is unpredictable which name will be returned. Therefore, when a hard link exists, you might need to maintain rules for both the link name and the original name.*

BPX.CAHFS.CREATE.EXTERNAL.LINK
Lets a user create an external link to an object outside of the file system, such as a z/OS MVS data set. An external link is a file that contains the name of an external object. If the external object is removed, the external link still contains the name of the non-existent object.

BPX.CAHFS.CREATE.SYMBOLIC.LINK
Lets a user create a symbolic link to an existing file. A symbolic link is a file that contains the name of another file. If the original file is removed, the file data is deleted but the symbolic link still contains a pointer to the non-existent file. Symbolic link names are validated when the link is created and deleted. All other accesses are validated with the original file name. In addition to this resource, the user also requires SERVICE(ADD) access to the HFS file resource rule for both the original file and the link file.

File Functions

File-related functions can be secured to various levels of granularity. This is accomplished by determining a user's highest level of access to a FACILITY resource. The SERVICE keyword of the FACILITY resource rule is used for this purpose. Depending on the SERVICE level defined, a user might be allowed to perform the function, can be denied, or the user might need access to the HFS file resource rule for the function to be permitted. The following actions are taken based upon the SERVICE value:

SERVICE Value	Action Taken
ADD	The user is allowed to perform the function against all files.
DELETE	The user is allowed to perform the function if the user also has SERVICE(DELETE) access to the HFS file resource rule. The service level of DELETE is not used in normal file access. It is utilized here to provide additional controls for file functions.
UPDATE	Processing is the same as for DELETE.
READ	The user is allowed to perform the function if the user also has SERVICE(DELETE) access to the HFS file resource rule, or if the user is considered the owner of the file. This is ownership as defined by CA SAF HFS security, not UNIX file UID.
None	If the user has no access to the FACILITY resource rule, the function is denied.

Since the absence of the SERVICE keyword in a rule implies **all** services, be sure to specify SERVICE in all of the file function FACILITY rules so that you do not inadvertently allow greater access to functions than you intended.

HFS file permission settings and UID/GID ownership are not used for validation purposes when CA SAF HFS security is active. However, the following resources restrict changes to these settings for those cases in which they must be maintained.

FACILITY Resources for File Functions

The following are the file function FACILITY resources:

BPX.CAHFS.CHANGE.FILE.ATTRIBUTES
Lets a user change extended file attributes, such as APF authorization and program control. Native z/OS UNIX Services will issue a FACILITY resource call to determine authorization to set the specific attribute, but not to specific files. Use of this file function resource provides additional control down to the file level. The FACILITY resource names used by native z/OS UNIX Services are: BPX.FILEATTR.APF and BPX.FILEATTR.PROGCTL.

BPX.CAHFS.CHANGE.FILE.AUDIT.FLAGS
HFS files contain two sets of audit flags: one that can be set by a normal user and the other that can only be set by an auditor. This resource lets a user change user-audit flags in a file. Auditor-audit flags can only be set by a user with the logonid AUDIT or unscoped SECURITY privilege. There is no validation when an auditor is changing a file's auditor-audit flags.

BPX.CAHFS.CHANGE.FILE.FORMAT
Lets a user change the format of a file. Changes include defining text data delimiters or binary file format.

BPX.CAHFS.CHANGE.FILE.MODE
Lets a user change any file mode information. This includes changes to file permission settings, setting the execution UID or GID indicators, setting the "sticky" bit, and maintaining Access Control Lists. Native z/OS UNIX permission settings are used for validation purposes only when CA SAF HFS security is inactive.

BPX.CAHFS.CHANGE.FILE.MODE.STICKY
Lets a user set the "sticky" bit in the file mode information. The "sticky" bit causes a program to be loaded from MVS libraries instead of the HFS. When setting this bit, the user also requires access to the resource BPX.CAHFS.CHANGE.FILE.MODE.

BPX.CAHFS.CHANGE.FILE.MODE.EUID
Lets a user set the execution-UID indicator in the file mode information. When this indicator is set, the program runs under the UNIX UID of the file owner instead of the UID of the user running the program. When setting this indicator, the user also requires access to the resource BPX.CAHFS.CHANGE.FILE.MODE.

BPX.CAHFS.CHANGE.FILE.MODE.EGID
Lets a user set the execution-GID indicator in the file mode information. When this indicator is set, the program runs under the UNIX GID of the file owner instead of the GID of the user running the program. When setting this indicator, the user also requires access to the resource BPX.CAHFS.CHANGE.FILE.MODE.

BPX.CAHFS.CHANGE.FILE.OWNER
Lets a user change file owner UID setting. Native z/OS UNIX ownership settings are used for validation purposes only when CA SAF HFS security is inactive.

BPX.CAHFS.CHANGE.FILE.GROUP
Lets a user change file owner GID setting. Native z/OS UNIX ownership settings are used for validation purposes only when CA SAF HFS security is inactive.

BPX.CAHFS.CHANGE.FILE.TIME
Lets a user change the last access or modification time to the current time or a user-specified time. If the current time is to be set and the user has write access to the file, the function is allowed. If the user does not have write access or a user-specified time is to be set, access must be allowed to this FACILITY resource.

Sample Rules

The following example shows rules that allow Thelma to change the file mode and owner for all files. Louise is allowed to change the file mode for only those files that reside in a certain directory, but is not allowed to change the file owner in any file:

```
$KEY(BPX) TYPE(FAC)
CAHFS.CHANGE.FILE.MODE UID(thelma) SERVICE(ADD) ALLOW
CAHFS.CHANGE.FILE.MODE UID(louise) SERVICE(DELETE) ALLOW

$KEY(BPX) TYPE(FAC)
CAHFS.CHANGE.FILE.OWNER UID(thelma) SERVICE(ADD) ALLOW

$KEY(/certain) TYPE(HFS)
directory.- UID(louise) SERVICE(DELETE) ALLOW
```

Implementing CA SAF HFS Security

CA SAF HFS security is an application of CA ENF/USS (UNIX System Services). This security application is activated when both of these conditions are met:

1. The appropriate DCM modules are linked into the ENF database.
2. CA SAF HFS security is enabled.

The implementation steps are as follows:

1. Determine if exit processing is required for path name translation, user path definition or to enable file ownership. See the following for specifics regarding exit processing. If using the exit, assemble and link the exit code using the sample SMP/E usermod found in ACFPTFS member UM80001.
2. Define HFS file and function resource rules. It is recommended that all the function resource rules described above be defined. A utility is provided to assist in creating these resource rules. See the below for details.
3. If you utilize the user file ownership feature of CA SAF HFS security, also define rules for users, or in a decentralized security environment, notify users that they should write rules for themselves.
4. Define the HFS file rules as resident using the GSO INFORDIR record:

   ```
   SET C(GSO)
   CH INFODIR TYPES(R-RHFS)
   ```

5. Verify that the proper level of CA ENF is available to support ENF/USS. This is provided by CA Common Services.

   Note: You must have CA Common Services installed at 9901 genlevel or higher.

6. The ENF started task must be a valid OMVS user. Message CARR014E is issued if this is not done. Ensure the ENF logonid specifies a group and that the group is defined in an OMVS GROUP profile record. Define an OMVS USER profile record with UID(0) for the ENF logonid:

   ```
   SET PROFILE(USER) DIV(OMVS)
   INSERT enf UND(0)
   ```

7. Install the following DCM Modules into the ENF database using the ENFDB utility program: CARRDCM0 and J165DCM0.

   The DCM modules come from the following CAILIBS:

   - J165DCM0 is in the ACF2.CAILIB
   - CARRDCM0 is in the CA common Services USS CAILIB

   The sample JCL to move the ACF2 J165DCM0 can be found in the ACF2.SAMPJCL library under member ENFUSS. It can also be modified for CA-Common Services CARRDCM0. See CA-Common Services documentation for details on the CAS9DB program.

   Note: If you previously installed the J165DCM0 DCM (Data Control Module) into the ENF database, you do not have to rerun this step.

8. Defining a VLF class for use as a cache can enhance performance of ENF/USS. The cache size is determined by the MAXVIRT specification. You can approximate the number of cache entries by dividing the defined amount of VLF storage by the average size of your path names. Add the following to your current COFVLFxx member in SYS1.PARMLIB:

   ```
   CLASS NAME(CAENFU) /* ENF/USS pathname cache */
   EMAJ(PATHCACHE) /* Major name */
   MAXVIRT(256) /* 1 megabyte */
   ```

   See the CA Common Services documentation for additional information on defining a VLF class.

9. Adding the NON-CNCL attribute to the BPXOINIT logonid during initial testing will allow OMVS to successfully initialize without violations. Once appropriate rules are in place, the NON-CNCL attribute should be removed.

10. The following message is issued by CA ENF/USS at ENF startup when CA SAF HFS security hooks are in place:

    ```
    CARR036I - SAFHFINT / J165 Now Initialized
    ```

11. CA SAF HFS security must be enabled. If the GSO UNIXOPTS record specifies HFSSEC, then CA SAF HFS security will be enabled once CA ENF/USS is started. The UNIXOPTS GSO record defaults to NOHFSSEC.

    The F ACF2,HFS(STATUS|ENABLE|DISABLE) command may be used to enable, disable and check the status of CA SAF HFS Security. Further information on this command can be found in the *Systems Programmer's Guide*.

12. Run ACFRPTRV during the implementation phase. Review violations and loggings for HFS and FAC resource types and create appropriate rules. If this was done in a prior eTrust CA-ACF2 release, this step may be skipped.

Exit Processing

An exit point is provided for installation-specific processing. This exit is called for both an initialization function, where options involving pathname translation and user path processing can be selected, and a pathname translation function, where final modification to the pathname can be made before validation is performed.

The exit must be reentrant and capable of running AMODE(31) and RMODE(ANY). The exit can be defined in the GSO EXITS record or can be linked together with load module SAFHFSEC as a CSECT called SAFHFUSR. If the SAFHFUSR CSECT is linked into SAFHFSEC, it will be called and the GSO EXITS record will be ignored. Note that all exits specified in the GSO EXITS record must be placed in LPA. A sample SMP/E usermod can be found in ACFPTFS, member UM80001. This usermod contains a sample exit along with assembly and link edit job steps.

Upon entry to the exit, register R1 points to a list of addresses. The end of the list is indicated by the high order bit in the last fullword.

For an initialization function, the exit is passed the following parameter addresses:

- +0—The address of a single byte containing the character 'I' indicating that this is an initialization function.
- +4—The address of a 512-byte work area for the use of the exit program.
- +8—The address of a 255-byte field in which the user can return the path location where user directories are located. Upon input, this field contains hex zeros.
- +12—The address of a single byte which, when set to 'Y' by the exit, indicates that user ownership of files is in effect.
- +16—The address of a 256-byte translation table, which is used to translate certain special characters in a path name.

When the exit returns a user directory path location, CA SAF HFS processing uses that path name to determine if the path name to be validated should be translated to a form such that the user ID of the owner of the path becomes the high-level qualifier of the path name. This will allow HFS file rules to be written at the user level and, if running in a decentralized environment, allow users to maintain their own HFS file rules. The default is that no translation takes place for user directories.

For example, if the exit returns the value /u/ as the user directory path name location, and the file accessed is /u/user01/xfile, then the resource name validated is $$USER01.XFILE. A rule to allow access to this file could be:

```
$KEY($$USER01) TYPE(HFS)
XFILE UID(*) ALLOW
```

When the exit returns the character 'Y' indicating that user ownership of files within one's own directory is in effect, no validation is performed when the current user's logonid matches that in the user directory. In the previous example, validation is bypassed when USER01 accesses file /u/user01/xfile. This option is meaningless if a user directory path location is not also returned.

The supplied translate table is in a format acceptable as input to the assembler TR instruction. The default translate table translates all slash characters in a path name, with the exception of the leading slash, to a period character. Other special characters will be translated into the dollar sign ($). These include characters that are used as masking characters in resource rules. If not translated, these characters could create undesired results. The special characters include the period, asterisk, dash, plus, blank, and quote. The exit can further modify any character in the table to meet special needs, with the exception of the slash character, which will always be translated to a period delimiter.

For a path name translation function, the exit is passed the following parameter addresses:

- +0—The address of a single byte containing the character 'P' indicating that this is a path name translation function.
- +4—The address of a 512-byte work area for the use of the exit program.
- +8—The address of a 255-byte field containing the resource name as modified by CA SAF HFS processing. This is the name that will be used for validation. The exit can return a modified path name in this same field.
- +12—The address of a 1023-byte field containing the original, unmodified path name.

The exit can use this exit function to make any specific modifications to the path name beyond that already performed by CA SAF HFS security processing.

Troubleshooting

Reporting and logging is done through the existing eTrust CA-ACF2 resource report, ACFRPTRV. When the TRACE attribute is on in a user's logonid, trace records will also appear on the report. The report will show the translated name of the HFS file used for validation along with the first 256 characters of the original HFS path name. ACFRPTRV should be reviewed when researching validation problems.

The CA SAF HFS security interface can be traced by using the SECTRACE command. A trace of internal functions of CA SAF HFS security is enabled through use of the SECTRACE TYPE=HFS keyword. The trace output might be requested by Technical Support. The syntax is:

```
SecTrace SET,TYPE=HFS
```

The following keywords are meaningful when TYPE=HFS is specified:

```
ID=
JOBname=
USERid=
ENable|DISable
ACTION=
MATCHLIM=
DEST=CONSOLE|JOBLOG|SYSLOG
CONSid=
MSGid|NOMSGid
```

Other keywords are ignored. If DEST is not specified, the default is DEST=SYSLOG.

The SAF validation calls invoked by CA SAF HFS security can also be traced. These SAF calls are the file and function validations that are passed to eTrust CA-ACF2. Enable this tracing by first issuing the SECTRACE SET command detailed in the following, followed by a reply to the prompt:

```
ST SET,ID=id,TYPE=SAFP,DEST=dest,END
R nn,REQSTOR=SAFHFSEC,END
```

CA SAF HFS Rule Generation Utility

Several utility programs are provided to generate eTrust CA-ACF2 resource rules to be used as a starter set of rules for new implementations. The HFS resource rules that are created by the SAFHFACF utility give access based upon the file permission bits defined for 'other' users. In other words, the rules will give users the same default access to files as they have when not running CA SAF HFS security. The HFS resource rules that are created by the SAFHFACL utility give access based upon the file permission bits defined for users by Access Control Lists (ACLs). The generated rules must be reviewed and modified to allow appropriate users access to files based on owner and/or group permissions.

The input file to the SAFHFACF utility is created by the OMVS ls command. This file contains a listing of all files and directories in the HFS. The input file for the SAFHFACL utility is created by processing the output from the OMVS ls command. The output files from both utilities contain commands to compile eTrust CA-ACF2 rules sets and to add rules using ACFNRULE. These files are meant to be reviewed, adjusted, and executed using batch TMP.

The ACFHFSRP job in the SAMPJCL library consists of several steps that create the required input files and execute the utilities. You must supply your standard jobcard and change symbolics to the correct values for your site prior to executing the ACFHFSRP job.

ACFHFSRP job steps:

1. Execute the ACFHFSPP REXX script to create the HFS files to be used as input for both utilities. This REXX exec is copied from the ACF2 CAIEXEC library to the HFS file system. It will generate the files that are required as input to SAFHFACF and ACFHFACL. If these files cannot be created, this step returns a condition code of 12.

2. Execute batch TMP to copy the HFS files to MVS datasets. The HFACFIN dataset should look similar to this:

```
/:
total 232
drwx------ 3 USER OPENMVS 0 Jun 3 2001 JavaS390
drwxr-xr-x 4 USER 0 May 7 2001 bin
drwx--x--x 2 USER OPENMVS 0 Oct 1 2000 dev
drwxr-xr-x 8 USER OPENMVS 0 Nov 4 17:05 etc
drwxr-xr-x 2 USER 0 Jan 20 1998 lib
drwxrwxrwx 2 USER 0 Jan 19 11:51 tmp
drwxr-xr-x 8 USER OPENMVS 0 Jan 15 15:47 u
drwxr-xr-x 11 USER 0 Jan 20 2001 usr
/JavaS390:
total 16
drwxrwxrwx 7 USER ZEROGRP 0 Sep 25 2002 J1.1.1
```

   If the data is not in this format, SAFHFACF will terminate. An error message in the SYSPRINT file will display the line that was not recognized.

   The MVSACLIN file should look similar to this:

```
drwxr-xr-x+ /u/user1/dir1
user:211 rwx
-rwx------+ /u/user1/file1
user:212 r-x
user:213 rw-
-rwxrwx---+ /u/user1/file2
user:214 -wx
-rw-------+ /u/user1/file4
user:215 rw-
-rw-r--r--+ /u/user1/file5
user:210 r-x
user:211 rwx
user:212 r-x
user:213 r--
```

   If no user ACL's exist on this file system, the MVSACLIN file will contain the message ' No ACL's Found'.

3. Execute the SAFHFACF utility to generate ACF commands to create resource rules based on 'other' permission bits. If mixed case resource names are being used for HFS, code the EXEC card as follows:

```
//RUNHFACF EXEC PGRM=SAFHFACF, PARM=MIXED
```

   PARM=MIXED will cause the output rules to be generated in mixed case format. After the job is executed, review the data in the RULESETS and RULELINE datasets.

   The RULESETS dataset contains commands to compile all rules sets created by the utility. It also contains commands to backup existing HFS or FAC rules by decompiling them into a file. Some of the rule sets contain rules to allow but log access for 14 days. These rules are meant to assist in the implementation process. After reviewing the loggings reported by ACFRPTRV and writing appropriate rules, these rule lines should be removed.

The RULELINE dataset contains the ACFNRULE commands to add specific rule lines to the rules sets. This data should be edited to remove redundant rule lines. The format of the output should make it fairly obvious which lines could be removed. For example, some directory entries contain example or demo files, which could be consolidated into one rule. Consider the following utility output:

```
ACFNRULE KEY(/USR/LPP/TCPIP) TYPE(HFS) NOLIST -
ADD(- SERVICE(READ) UID(*) ALLOW) -
ADD(X11R6.- SERVICE(READ) UID(*) ALLOW) -
ADD(X11R6.XAMPLES.- SERVICE(READ) UID(*) ALLOW) -
ADD(X11R6.XAMPLES.CLIENTS.- SERVICE(READ) UID(*) ALLOW) -
ADD(X11R6.XAMPLES.CLIENTS.BITMAP.- SERVICE(READ) UID(*) ALLOW) -
ADD(X11R6.XAMPLES.CLIENTS.XAUTH.- SERVICE(READ) UID(*) ALLOW) -
ADD(X11R6.XAMPLES.CLIENTS.XCLOCK.- SERVICE(READ) UID(*) ALLOW) -
ADD(X11R6.XAMPLES.CLIENTS.XLOGO.- SERVICE(READ) UID(*) ALLOW) -
ADD(X11R6.XAMPLES.CLIENTS.XPROP.- SERVICE(READ) UID(*) ALLOW) -
ADD(X11R6.XAMPLES.DEMOS.- SERVICE(READ) UID(*) ALLOW) -
ADD(X11R6.XAMPLES.DEMOS.XSAMP1.- SERVICE(READ) UID(*) ALLOW) -
ADD(X11R6.XAMPLES.DEMOS.XSAMP2.- SERVICE(READ) UID(*) ALLOW) -
ADD(X11R6.XAMPLES.MAN.- SERVICE(READ) UID(*) ALLOW) -
ADD(X11R6.XAMPLES.MAN.CAT1.- SERVICE(READ) UID(*) ALLOW)
```

Because all of the directories and files are defined with READ access, the rule set can be safely reduced to the following by removing the rules for directories under the XAMPLES directory:

```
ACFNRULE KEY(/USR/LPP/TCPIP) TYPE(HFS) NOLIST -
ADD(- SERVICE(READ) UID(*) ALLOW) -
ADD(X11R6.- SERVICE(READ) UID(*) ALLOW) -
ADD(X11R6.XAMPLES.- SERVICE(READ) UID(*) ALLOW)
```

SAFHFACF may issue the message:

```
Warning - $PREFIX maximum length exceeded.
```

This indicates that the $PREFIX value generated for a rule set is greater than the 40 character maximum. This problem can be addressed by using masking at a lower level to eliminate the need for the nextkey rule set, or, by shortening the $PREFIX value by removing file name levels from the end of the $PREFIX value and adding them to the rule lines themselves.

4. Execute the ACFHFACL REXX script to generate ACF commands to create resource rules based on Access Control Lists. ACFHFACL requires 1 argument: the UID string length. The UID string is site defined, up to a maximum of 24 characters. Specify the correct UID string length for your site. This step can be commented out if user ACL's do not exist on this HFS file system.

After modifications are made, run the commands contained in the HFSACF.RULELINE, HFSACF.RULESETS, and HFSACL.RULELINE datasets as input to batch TMP. The ACFHFIKJ job in the SAMPJCL library can be used to execute the ACF2 commands contained in these datasets. Examine the output for successful completion. The ACF50035 message will indicate any errors during ACFNRULE processing. After resolving any errors, you can run the modified RULESETS and RULELINE commands through the batch TMP once again. Any rules previously created will be backed up by the commands in the RULESETS file.

As mentioned earlier, the HFS rules that are created give access based upon the file permission bits defined for 'other' users. As a final step, the rules must be modified to allow appropriate users greater access to the file resources than that granted to the general user community. The rule set keys should closely represent the various applications using HFS resources. Ownership and administrative responsibilities for the data should be determined so that the rules can be maintained on an ongoing basis.

CA SAF HFS Security Modification Command

F ACF2,HFS(STATUS|ENABLE|DISABLE)
This console command can be used to control or check the status of CA SAF HFS security.

F ACF2,HFS(STATUS)
Issues messages indicating the current status of CA SAF HFS security.

F ACF2,HFS(ENABLE)
Enables CA SAF HFS security. When enabled, normal z/OS UNIX security access validation is bypassed. This includes checking of file permission bits, superuser status and normal z/OS UNIX Security Services.

F ACF2,HFS(DISABLE)
Disables CA SAF HFS security. When disabled, normal z/OS UNIX security access validation is enabled. This includes checking of file permission bits, superuser status and normal z/OS UNIX Security Services.

If the requested function is STATUS and the user issuing the command is defined to security as an auditor, no further authorization is required. Otherwise, the user issuing the command must be allowed access to a resource in the SAF FACILITY class. The resource name is:

```
BPX.CAHFS.SECURITY.function
```

Where *function* can be STATUS, ENABLE, or DISABLE.

CA SAF HFS security can also be enabled or disabled using the GSO UNIXOPTS record.

If HFSSEC is specified then CA SAF HFS security will be enabled once CA ENF/USS is started. The UNIXOPTS GSO record defaults to NOHFSSEC.

CA SAF HFS Security Modification Utility

A utility program is provided to allow authorized users to display the status of and to enable or disable CA SAF HFS security. The utility is run in batch with the requested function passed in the PARM field of the JCL EXEC statement. Sample JCL follows:

```
//jobname  JOB CLASS=a
//stepname EXEC PGM=SAFHFMOD,PARM=function
```

Where *function* can be one of the following:

STATUS
Issues a message indicating the current status of CA SAF HFS security.

ENABLE
Enables CA SAF HFS security. When enabled, normal z/OS UNIX security access validation is bypassed. This includes checking of file permission bits, superuser status and normal z/OS UNIX Security Services.

DISABLE
Disables CA SAF HFS security. When disabled, normal z/OS UNIX security access validation is enabled. This includes checking of file permission bits, superuser status and normal z/OS UNIX Security Services.

The program must reside in an APF-authorized library. If the requested function is STATUS and the user running the job is defined to security as an auditor, no further authorization is required. Otherwise, the user running the job must be allowed to a resource in the SAF FACILITY class. The resource name is:

```
BPX.CAHFS.SECURITY.function
```

Where *function* can be STATUS, ENABLE, or DISABLE.

Chapter 23

Using the SYSPLEX Coupling Facility

The Coupling Facility is a feature of the operating system that lets systems in a SYSPLEX environment communicate and share common data with each other.

- The communication feature (XCF) provides a way for each system in the SYSPLEX to send messages or signals to all other systems.
- The data sharing feature (XES) provides the ability for systems in the SYSPLEX to share common data that would normally be obtained from a database. This feature can save input/output (I/O) requests to the database.

eTrust CA-ACF2 provides the ability to use these features for all eTrust CA-ACF2 systems running in a SYSPLEX environment. The XES data sharing feature provides the ability to share eTrust CA-ACF2 database records throughout the SYSPLEX for all eTrust CA-ACF2 systems and the XCF message router allows the communication of eTrust CA-ACF2 operator commands between systems in the SYSPLEX environment.

This chapter discusses the following topics:

- How eTrust CA-ACF2 uses the XCF and XES services
- Setting up the SYSPLEX environment for eTrust CA-ACF2
- Starting/stopping the SYSPLEX under eTrust CA-ACF2
- The SHOW SYSPLEX subcommand
- Implementing SYSPLEX
- SYSPLEX return and reason codes

SYSPLEX XES Service

The XES data sharing feature lets the customer define structures in the SYSPLEX that multiple systems can use. These structures contain data to be shared between these systems. Once the data has been placed in the structure it is valid to be read, updated, or deleted by any system connected to that structure.

It is the responsibility of the systems programmer to define the structure name to the Coupling Facility. This structure name is then defined to eTrust CA-ACF2. Alternatively, a second structure can be defined for eTrust CA-ACF2 recovery purposes. This alternate structure is used in case the primary structure has failed for any reason.

There are multiple types of structures that can be defined to the Coupling Facility, including the List, Cache, and Lock structures.

eTrust CA-ACF2 uses the List structure to contain data from its databases. With a List structure, numerous functions can be performed to the data after a connection to the structure. For example, reads, writes, deletes, or multiple deletes can be performed.

A connection to the structure is a way to initially establish communication to a structure. It is performed prior to any other function and indicates to the Coupling Facility which structure is to contain the data for future processing. When the application is done communicating with the structure in the Coupling Facility, a disconnect must be performed.

A List structure can be redefined while systems are connected to it. A redefining of the structure can take place if, for example, the structure becomes full and it is necessary to increase the size of it. This option is available based on how the structure is defined. If the current structure size is less than the maximum size of the structure as defined when the structure was created, the structure size can be increased.

A List structure is made up of headers and data elements. Each header can be used to break up different types of information in the structure. There is a maximum of sixteen headers that any one structure can have. With each header, data elements contain the data that is stored in the structure.

The number of headers and the size of each data element is defined to the Coupling Facility when the application "connects" to the structure. How the data is maintained and on which header is up to the application connected to the structure. With eTrust CA-ACF2, each header within the structure contains a different type of data. For example, header one can contain RULES data and header two can contain USER data. Within each header there are multiple entries. Each entry in turn is comprised of multiple elements that hold the actual data.

How much data can a structure hold? This is dependent on a number of factors. First, the total size of the structure is dependent on how the structure is defined to the Coupling Facility. The size definition is done at the same time the systems programmer defines the name of the structure.

For Structure Sizing, allocate the sysplex to accommodate the maximum number of records. For example, if there is a need for 2500 different logonid records in a 4-plex environment, then you need to ensure that there is enough storage to handle this storage requirement. In this case it is: 2500*1024 = 2.5 meg. In a pre-release 8.0 eTrust CA-ACF2 SYSPLEX environment, there was a minimum element size of 4K and a minimum entry size of 4K. The maximum entry size is 32K. The exact size of the entry chosen is based on whether or not rulelong is active. In r8.0, the eTrust CA-ACF2 SYSPLEX environment has been optimized for LOGONID records. The element size is defined as 1K, the same as that of a LOGONID record. Each entry can still be as great as 32K in order to accommodate rules up to 32K.

Next, the amount of data the structure can hold is dependent on how the structure is defined at connection time. The first system to connect to a structure supplies information on how the structure is set up. For example, the number of headers and the maximum data element size are two of many factors determining the amount of data that can be placed in a structure. These factors are supplied at connection time and are used in a calculation that is done during the connection. It is important to remember that no matter what structure size it defined, a certain portion of the structure will be reserved by IBM for structure overhead. For information on how to calculate what that portion is, see the IBM guide *Setting Up A Sysplex*.

Once a List structure is full, no more data can be added to it. Deleting some or all of the data is necessary to be able to continue to place data in the structure. In a pre-release 8.0 eTrust CA-ACF2 system, eTrust CA-ACF2 automatically handles the clearing of the elements from the structure when the structure reached 90% full to allow the addition of more current elements.

In a eTrust CA-ACF2 r8.0 system, you can specify in the GSO SYSPLEX record a value for FULLTHSH(). When the structure reaches this value, the action indicated in the FULLACTN() field will be executed. If FULLACTN(CLEAR) is specified, a warning message will be issued that the structure is full and the structure will be cleared. If FULLACTN(WARNMSG) is specified, ONLY a warning message will be given. The last option you can specify is FULLACTN(NONE). In this case, no action will be taken, no warning message will be given, and the structure will **NOT** be cleared.

IBM also allows you to specify at structure creation whether or not the structure is eligible for system-managed rebuild. If this option is selected, when the structure reaches the full threshold as defined in the CFRM policy used to create the structure, IBM will automatically try to increase the size of the active structure. This will succeed if the current size of the structure is less than the maximum size of the structure. This will enable more records to be written to the structure.

Pay close attention to the fact that there are two places to specify a full threshold value. The first is when the structure is created via the CFRM policy. The second is the FULLTHSH() field in the GSO SYSPLEX record. If the full threshold value in the CFRM policy is set lower than the value in the GSO SYSPLEX record, the system will always try to rebuild the structure first. If the value in the FULLTHSH() field is set lower than the full threshold specified in the CFRM policy, the action specified in the FULLACTN() field of the GSO SYSPLEX record will be executed.

The name of the structure is up to the systems programmer. Any naming convention can be used as long as it is valid in the Coupling Facility.

The Coupling Facility design handles any failure in it or its connections by disconnection and reverting to normal I/O processing. After the problem is resolved, the user must reactivate the Coupling Facility by reactivating the SYSPLEX connection. In eTrust CA-ACF2 you would need to stop the SYSPLEX processing and restart it.

SYSPLEX XCF Service

The XCF communication function within a SYSPLEX environment allows communication to all systems in the SYSPLEX through message routing. Any system within the SYSPLEX can "join" a group that is able to send and receive messages from each of the other joined members within the same group. For example, if system A wants to send a message to all other systems in the group, it can do so by issuing a send message to the XCF service. XCF is then responsible for notifying all other "joined" members within the group that a message is waiting for them. Each notified member is then responsible for getting the message that was sent. What it does after receiving the message is up to the application.

This message routing feature is a way for all systems within the SYSPLEX to communicate with each other, especially in situations that each system needs to know about.

Note: For further information on SYSPLEX XES and XCF services, see any of the IBM SYSPLEX Services manuals.

How eTrust CA-ACF2 Uses the XES and XCF Services

eTrust CA-ACF2 supports both the SYSPLEX XES and XCF services. The XES function is used to manipulate any or all of the eTrust CA-ACF2 databases. The XCF communication function is used to provide a way for the customer to route eTrust CA-ACF2 operator commands to all other eTrust CA-ACF2 systems in the SYSPLEX. Both of these features are available as options on your system.

eTrust CA-ACF2 and the SYSPLEX XES Service

The XES data sharing feature lets the customer define a primary and an optional alternate List structure in the Coupling Facility for use with all three eTrust CA-ACF2 databases. This allows the use of the Coupling Facility as a source of database records. If this feature is enabled, eTrust CA-ACF2 attempts to obtain the requested data from the Coupling Facility before any physical I/O is performed. If the requested data is obtained from the Coupling Facility, no I/O is performed, eliminating I/O overhead.

If you are using the eTrust CA-ACF2 CACHE facility, the cache call is performed prior to any call to the Coupling Facility. If the data is found in the CACHE, no calls to the Coupling Facility are performed.

You can control which eTrust CA-ACF2 databases use the Coupling Facility using the GSO SYSPLEX record. You might want only the LID database to be shared by all eTrust CA-ACF2 systems in the SYSPLEX, or you might want all the eTrust CA-ACF2 databases to use the Coupling Facility. Since this option is enabled through a GSO record, you can reset the current settings at any time and refresh the SYSPLEX record to reflect the new settings. For example, you might decide to start with only the LID database to use the Coupling Facility. At a later time you might want to add the RULES database. Simply update the GSO SYSPLEX record and issue the ACF2 REFRESH operator command to reflect the change.

Note: Because this option is refreshable, all the systems using XES sharing on one database must have the same database names for the other two, even if you are not planning on sharing these other two databases. For example, if sharing the logonid database, the rules and infostorage database names must also match, even if they are not being shared.

Whenever direct physical I/O is requested to the eTrust CA-ACF2 database, a number of decisions must be accounted for. It is important that any update to a record is reflected on the database prior to placing it in the Coupling Facility. Most important is the need to obtain the data from the Coupling Facility when a direct READ of a record is requested. If any record is being updated, a lock to the Coupling Facility is issued to prevent the same record from being updated in the Coupling Facility from another eTrust CA-ACF2 system. If all three databases are in the CF, the lock will only occur for the database being updated, not all three databases. If a request to the Coupling Facility is denied for any reason, the Coupling Facility is bypassed and normal I/O processing is done.

A List structure lets eTrust CA-ACF2 optimally control what is placed in the Coupling Facility and it maintains the data once it is there. The primary List structure, defined by the systems programmer, is used to hold the data for any one or all of the eTrust CA-ACF2 databases. The optional alternate List structure is used if something happens to the primary List structure. If no alternate is defined, and the primary is unavailable, eTrust CA-ACF2 defaults to its regular I/O processing.

How does this all work? After the structures have been defined to the Coupling Facility, there are two important modifications to eTrust CA-ACF2 GSO records that must be done. First, eTrust CA-ACF2 needs to be aware that the Coupling Facility is to be used. To do this, turn on the SYSPLEX option in the GSO OPTS records. Secondly, a GSO SYSPLEX record needs to be defined. This record reflects the structure names and the databases to be used with the Coupling Facility. Once these updates have been made, simply issue the ACF2 REFRESH operator command to reflect these changes.

At eTrust CA-ACF2 start-up, or whenever an F ACF2,SYSPLEX(START) command is issued, eTrust CA-ACF2 attempts to connect to the defined structures in the Coupling Facility. If successful, eTrust CA-ACF2 attempts to use the Coupling Facility for all direct I/O requests being made for the databases defined.

eTrust CA-ACF2 and the SYSPLEX XCF Service

The XCF message routing lets all eTrust CA-ACF2 operator commands be routed to all other eTrust CA-ACF2 systems in the SYSPLEX that are using the same defined GROUP name. This lets one system update critical eTrust CA-ACF2 information and automatically route the ACF operator command to the other eTrust CA-ACF2 systems in the SYSPLEX. This provides critical data synchronization on all systems without operator intervention.

The GROUP name, defined in the GSO SYSPLEX record, defines the group that the eTrust CA-ACF2 systems "join" when the SYSPLEX is started. If there are any other eTrust CA-ACF2 systems that have joined the same GROUP and the "send" parameter has been specified on the ACF operator command, the command is sent to those eTrust CA-ACF2 systems. The ACF operator command is then processed on those eTrust CA-ACF2 systems as if the command was entered on that system.

The "send" parameter can be specified in two ways:

- The following command is sent by XCF Services to all eTrust CA-ACF2 systems that have joined the same group as the sending system. This is only done if the command has been processed on the local system:

  ```
  F ACF2,REFRESH(OPTS),XCF(*)
  ```

- The following command is sent to the systems with member names *target1* and *target2* only:

  ```
  F ACF2,REFRESH(OPTS),XCF(target1,target2)
  ```

  The target list is used to limit the scope of systems to which the command is routed.

The member name of a system is the eTrust CA-ACF2 SYSID that is active when eTrust CA-ACF2 is started or the following command is issued:

```
F ACF2,SYSPLEX(START)
```

The member name must be unique within an XCFGROUP.

If the command is successfully sent to any other eTrust CA-ACF2 system in the SYSPLEX, the following message is displayed:

```
ACF79945 XCF SEND COMMAND COMPLETED
```

If any of the systems specified in a target list are not connected to the XCF group, the following message is displayed:

```
ACF79949 XCF SEND COMMAND INCOMPLETE - MEMBER(S) NOT CONNECTED
```

If no other eTrust CA-ACF2 systems have joined the same GROUP name in the SYSPLEX, the following message is displayed:

```
ACF79947 XCF SEND COMMAND FAILED - NO ACTIVE CONNECTIONS
```

If the attempt to send a command to other eTrust CA-ACF2 systems in the SYSPLEX failed, the following message is displayed:

```
ACF79946 XCF SEND COMMAND FAILED
```

If an attempt is made to use the "XCF(*)" parameter on the ACF operator command and XCF is not active, the following message is displayed:

```
ACF79939 SYSPLEX-XCF not active - XCF(*) invalid
```

Duplexing a Coupling Facility Structure

eTrust CA-ACF2 supports system-managed duplexing of the Coupling Facility structure. You can use the SETXCF z/OS operator command to initiate the duplexing process of the Coupling Facility structure, with no disruption to the connected systems. Duplexing may also be initiated automatically when the first system connects to the structure.

Structure duplexing is a process by which a duplexed copy of a structure is created and maintained, so that in the event of a failure, a viable structure will remain available to the application. While the structure is duplexed, operations to the structure are maintained in a synchronized manner — with system-managed duplexing through protocols established by z/OS.

The following requirements must be considered before system-managed duplexing can be established in a Parallel Sysplex environment:

- At least two coupling facilities at CFLEVEL=12 or higher are required. It is recommended to have a minimum of three coupling facilities at CFLEVEL=12 so that if one coupling facility is removed for maintenance, there will still exist two others to contain the duplexed pair of structures. The coupling facilities must all be defined in the preference list for the structure to be duplexed.
- The structure has at least two entries in the CFRM PREFLIST.
- CF-to-CF links should be available between a pair of coupling facilities in which a duplexed pair of structure instances are to be allocated. The link connectivity between the coupling facilities must be bi-directional.
- Processor models that support system-managed duplexing are IBM Eserver zSeries 900 Driver level 3C and S/390 Parallel Enterprise Server - G5 and G6.
- System-managed duplexing requires z/OS V1R2 or higher with APAR OW41617 on all systems connecting to the duplexed structure.
- The CFRM couple data set must be formatted with the ITEM NAME(SMDUPLEX) NUMBER(1) statement.
- A structure should be made eligible for duplexing by specifying DUPLEX(ENABLED) or DUPLEX(ALLOWED) in the active CFRM policy.

When the structure is defined with DUPLEX(ENABLED) in the CFRM policy, the system will automatically attempt to initiate duplexing when the first system connects to the structure as part of the initial structure allocation process.

When the structure is defined with DUPLEX(ALLOWED) in the CFRM policy, duplexing is not initiated when the structure is allocated. In this case, duplexing may later be manually initiated by an operator command.

Use the following command to initiate the duplexing process for the allocated structure, while connected systems remain connected to the structure.

```
SETXCF START,REBUILD,DUPLEX,STRNAME=strname
```

When duplexing is active for the structure, the system maintains two identical copies of structure data. The initial copy is referred to as the OLD structure, and the duplexed copy is the NEW structure.

The following command is used to instruct the system to stop duplexing for the structure, and continue to maintain either the OLD copy or the NEW copy as the active structure.

```
SETXCF STOP,REBUILD,DUPLEX,STRNAME=strname,KEEP=NEW/OLD
```

Setting Up the SYSPLEX Environment for eTrust CA-ACF2

Defining the SYSPLEX environment to eTrust CA-ACF2 is easy to do. To use the XCF or XES feature, a GSO SYSPLEX record must be created. This record contains the information necessary for eTrust CA-ACF2 to communicate with the SYSPLEX using the proper structure and member names. The SYSPLEX field in the GSO OPTS record must also be turned on. For more details about SYSPLEX see the "Maintaining Global System Options Records" chapter.

Starting and Stopping the SYSPLEX under eTrust CA-ACF2

Whether using the XES data sharing feature, the XCF command routing feature, or both, the same commands are used to start and stop these SYSPLEX features with eTrust CA-ACF2.

To start the SYSPLEX functions, enter:

```
F ACF2,SYSPLEX(START)
```

To stop the SYSPLEX functions, enter:

```
F ACF2,SYSPLEX(STOP)
```

When starting or stopping the SYSPLEX for eTrust CA-ACF2, one of the following error conditions might be encountered:

- **ACF79935 INVALID SYSPLEX STATE – REQUESTED – IGNORED**

  "START" or "STOP" was not entered.

- **ACF79936 SYSPLEX cannot be started due to GSO OPTS setting**

  The SYSPLEX option was not set on the GSO OPTS record or the GSO OPTS record was not refreshed.

- **ACF79937 SYSPLEX cannot be started - already active**

  The SYSPLEX is already active.

- **ACF79938 SYSPLEX not active**

  The SYSPLEX cannot be stopped because it is currently not active.

When starting the SYSPLEX function, an attempt is made to connect to the Coupling Facility. Whether it is successful or not, one of the following messages is issued:

- **ACF79470 C.F. CONNECT (COMPLETE/FAILED) XCF GROUP: x STRUCTURE: x**

  Whether the connection to the Coupling Facility was successful or not, this message displays. It displays the XCF group name and the XES structure name defined by the GSO SYSPLEX record. If the XCF group or the XES structure were omitted, "N/A" appears.

- **ACF79472 CONNECT TO COUPLING FACILITY NOT PROCESSED - NO XCF OR XES PARMS**

  An attempt to start the SYSPLEX facility was made, but the XCF Group or the XES structure was not defined in the GSO SYSPLEX record. If it is defined, a refresh of this record was not performed.

- **ACF79473 CONNECT TO C.F. FAILED - DATA SET NAME CONFLICT**

  An attempt to start the SYSPLEX facility was made, but there is a conflict in the eTrust CA-ACF2 databases or the catalog the database resides in. Another eTrust CA-ACF2 system is already connected to the same structure in the Coupling Facility using a different set of eTrust CA-ACF2 databases. Use a different structure name or determine why another eTrust CA-ACF2 system is using a different set of eTrust CA-ACF2 databases.

- **ACF79475 ACF2 INACTIVE - COUPLING FACILITY UNAVAILABLE**

  eTrust CA-ACF2 is currently unavailable. Try again later.

- **ACF79476 CONNECT TO COUPLING FACILITY NOT PROCESSED - NOT AT PROPER RELEASE OF MVS**

  An attempt to start the SYSPLEX facility was made, but the release of MVS/ESA does not support the function requested. To use the XCF communication function you must be at least r4.3 of MVS/ESA and to use the XES function you must be at r5.1.

When stopping the SYSPLEX function through the modify ACF2 command the following message is displayed:

```
ACF79471 C.F. DISCONNECT (COMPLETE/FAILED) XCF GROUP: x STRUCTURE: x
```

Whether the disconnect to the Coupling Facility was successful or not, this message is displayed. It displays the XCFGROUP name and the XES structure name defined by the GSO SYSPLEX record. If the XCF group or the XES structure were omitted, "N/A" appears.

If at any time during the operation of the SYSPLEX XES function a fatal error condition exists on the defined Coupling Facility structure, a disconnection to the structure is performed internally by eTrust CA-ACF2 and the following message is displayed:

```
ACF79474 C.F. XES DISCONNECT REQUESTED - SHUT DOWN IN PROGRESS.
```

Clearing the SYSPLEX Structure

When using a list structure within the Coupling Facility, data is stored until manually removed. Once the structure is full, no more data can be added without deleting data that is in the structure (unless system managed rebuild is available).

With eTrust CA-ACF2, if the structure becomes full, eTrust CA-ACF2 clears the contents of the structure. With a pre-release 8.0 system, this occurs when the structure is 90% full. With a r8.0 structure, you can specify in the GSO SYSPLEX record, the threshold value. You can also specify what eTrust CA-ACF2 should do once the structure full threshold is reached.

You can also manually clear the structure at any time by issuing the following command:

```
F ACF2,SYSPLEX(CLEAR)
```

One of the following messages will be displayed:

ACF79478 CLEAR OF STRUCTURE (COMPLETED/FAILED)

If the attempt to clear the structure failed, it might be because of an outstanding lock to the structure. Retry clearing the structure at a later time.

Switching the SYSPLEX Structure

If you notice that the structure needs to be cleared frequently, it may be necessary to define and activate a larger structure. If you have defined a primary and an alternate structure, eTrust CA-ACF2 lets you switch the active status from whichever structure is active to the non-active structure. You can switch structures with the following command:

```
F ACF2,SYSPLEX(SWITCH)
```

If the primary structure is active, this command will switch from primary to alternate. If the alternate structure is active, this command will switch from alternate to primary. You can redefine the non-active structure and activate it with the SWITCH option. This lets you make changes to the sysplex structure without having to stop and restart sysplex processing.

SHOW SYSPLEX Command

You can use the SHOW SYSPLEX subcommand to display information about the current settings for the SYSPLEX facility. It also displays the number of times that eTrust CA-ACF2 has used the XES feature and the number of times messages have been sent or retrieved through XCF. Here is an example of the SHOW command:

```
ACF
show sysplex

 -- SYSPLEX COUPLING FACILITY --
 OPTION: SYSPLEX
 CURRENT XES STATUS: ACTIVE
 CURRENT XCF STATUS: ACTIVE

 COUPLING FACILITY DATA:
  INFOSTORAGE:  INACTIVE
  LOGONIDS:     ACTIVE
  RULES:        ACTIVE

  XCF GROUP NAME: N/A
  MEMBER NAME: DAH1

  PRIMARY STRUCTURE ACTIVE

  PRIMARY STRUCTURE NAME:SECURITY1
  ALTERNATE STRUCTURE NAME: N/A
  CURRENT STRUCTURE SIZE= 10,240K
  MAX STRUCTURE SIZE= 10,240K
  NUMBER OF STRUCTURE ENTRIES= 0
  MAX NUMBER OF STRUCTURE ENTRIES= 8,433

  FULL THRESHOLD= 95%
  ACTION AT THRESHOLD= CLEAR
  # OF TIMES STRUCTURE CLEARED= 0
  # OF TIMES STRUCTURE ALTERED= 0

  NUMBER OF XES WRITES= 0
         LID DB WRITES= 0
        RULE DB WRITES= 0
      INFSTG DB WRITES= 0

  NUMBER OF XES READS= 0
  NUMBER OF XES DELETES= 0

  NUMBER OF XCF MESSAGES SENT= 0
  NUMBER OF XCF MESSAGE GETS= 0
```

Field Descriptions

OPTION
Indicates that the SYSPLEX option has been set in the GSO OPTS record.

CURRENT XES AND CURRENT XCF STATUS
Indicates whether these functions are currently active or not. This example shows both are ACTIVE, meaning that a START of the SYSPLEX has been completed.

COUPLING FACILITY DATA
Indicates which of the eTrust CA-ACF2 databases have been set to use XES services. This example shows that logonids and RULES are using XES services and INFOSTG is not.

XCF GROUP AND MEMBER NAME
Indicates the name of the group that has been set to join other eTrust CA-ACF2 systems for ACF2 operator commands. Also indicates the current member of the group.

XCF GROUP NAME
Indicates the name of the group that has been set to join other eTrust CA-ACF2 systems for ACF2 operator commands.

PRIMARY AND ALTERNATE STRUCTURE NAMES
Indicates the name of the SYSPLEX structure that eTrust CA-ACF2 will use for its XES services. This example shows that a primary name has been set and no alternate is being used.

FULL THRESHOLD
Indicates the full threshold value.

ACTION AT THRESHOLD
Indicates the action to be taken by eTrust CA-ACF2 when the full threshold is reached.

 # OF TIMES STRUCTURE CLEARED
Indicates the number of times the structure has been cleared for any of the following reasons:

- In a pre-release 8.0 eTrust CA-ACF2 system, the full threshold of 90% was reached
- In a r8.0 eTrust CA-ACF2 system, the full threshold value specified in the GSO SYSPLEX record was reached and the action to be taken was specified as “CLEAR” in the FULLACTN field of the GSO SYSPLEX record
- The F ACF2,SYSPLEX(CLEAR) command was used

 # OF TIMES STRUCUTURE ALTERED
Indicates the number of times the structure has been altered by IBM.

LID DB WRITES
Indicates the number of logonid records written to the structure.

RULES DB WRITES
Indicates the number of rule records written to the structure.

INFSTG DB WRITES
Indicates the number of infostorage records written to the structure.

NUMBER OF XES WRITES/READS/DELETES
Indicates the number of successful writes to, reads from, and deletes that have been performed by XES services since the SYSPLEX start was issued. These values are cleared each time the SYSPLEX is started.

NUMBER OF XCF MESSAGES SENT/GETS
Indicates the number of successful times a command was sent by this system using XCF services, as well as the number of times this system has received a message (ACF operator command) from another eTrust CA-ACF2 system.

Implementing SYSPLEX

Follow these steps to implement the SYSPLEX feature of eTrust CA-ACF2:

1. Determine which SYSPLEX services will be used by your eTrust CA-ACF2 systems.
   - If using XCF (message routing) for eTrust CA-ACF2 operator commands, you must set up your SYSPLEX environment to be able to process XCF.
   - If using XES (data sharing) for the eTrust CA-ACF2 databases, you must set up your SYSPLEX environment by defining the structures in the SYSPLEX Coupling Facility that eTrust CA-ACF2 is to use.
   - The size of the structure to be defined is dependent on a number of factors, some of which include:
     - The number of eTrust CA-ACF2 systems using the structure.
     - Which eTrust CA-ACF2 databases will be using the structure.
     - How busy the system is (the number of people using it).
     - How large are the Rules/Infostorage databases (4K-32K).
     - What other factors must be considered?

       Allocate the SYSPLEX to accommodate the maximum number of records.
     - See the IBM SYSPLEX manuals to learn how to specifically set up these environments.

2. Create the GSO SYSPLEX record on each system that is going to use the SYSPLEX environment.
   - If using XCF (message routing), make sure that each of the eTrust CA-ACF2 systems that are to communicate have the same group defined. Only the systems with the same XCFGROUP name on the GSO SYSPLEX record will receive or get messages from the other eTrust CA-ACF2 systems.
   - If using XES (data sharing), make sure that all of the eTrust CA-ACF2 systems using the same set of eTrust CA-ACF2 databases are using the same structure in the SYSPLEX Coupling Facility. Also, be sure that for all systems using the same structure each has defined the same databases to the Coupling Facility. (For example, if system A has logonids and RULES database defined but not INFOSTG, make sure that each of the other eTrust CA-ACF2 systems also has only logonids and RULES defined.)
   - Make sure that all systems sharing the same eTrust CA-ACF2 databases are using the same structure. **Do not** have shared database systems use different structures.
   - Determine whether to set MINLVL80 (the default) or NOMINLVL80 in the GSO SYSPLEX record. If you have a mixture of pre-release 8.0 eTrust CA-ACF2 systems and r8.0 eTrust CA-ACF2 systems leave MINLVL80. If you are certain that all systems connecting to the structure will be at eTrust CA-ACF2 r8.0 you may set NOMINLVL80.
   - Once the GSO SYSPLEX record is set, refresh this record with the following ACF operator command:

     ```
     F ACF2,REFRESH(SYSPLEX)
     ```
3. Update the GSO OPTS record on each system that will be using the SYSPLEX environment.
   - Whether using the XES or XCF feature, the SYSPLEX field must be turned on to start the SYSPLEX feature. If NOSYSPLEX is set, eTrust CA-ACF2 SYSPLEX support cannot start.
   - Once the GSO OPTS record is set, refresh this record with the following ACF operator command:

     ```
     F ACF2,REFRESH(OPTS)
     ```
4. When you are ready to use the SYSPLEX feature of eTrust CA-ACF2, use the modify ACF operator command:

   ```
   F ACF2,SYSPLEX(START)
   ```
5. If you must stop using the SYSPLEX feature of eTrust CA-ACF2, use the modify ACF operator command:

   ```
   F ACF2,SYSPLEX(STOP)
   ```
6. After the SYSPLEX is active, use the ACF SHOW SYSPLEX subcommand to monitor the usage of the SYSPLEX environment.

SYSPLEX Return and Reason Codes

When using the SYSPLEX XES feature, you might find that not always will the data be obtained from the Coupling Facility. When a request is made to the Coupling Facility and eTrust CA-ACF2 does not receive a good response, a message will be displayed to the console. This does not mean that eTrust CA-ACF2 has a problem using the Coupling Facility. It could simply mean that a record was not found on a read or delete request. In any case, message CASECCFS is displayed in the format:

```
CASECCFS - Function: xxxxxxxx Rc=xxxxxxxx  Rs=xxxxxxxx
```

Where Function is one of the following:

- READ
- WRITE
- DELETE
- LOCK
- CONNECT
- DISCONNECT
- MSGALL
- GETMSG

The *Rc* (Return code) and *Rs* (Reason code) come from the SYSPLEX Coupling Facility interface. These codes can be found in the *IBM z/OS V1R2.0 MVS Programming: Sysplex Services Reference* guide.

One of the most common times that this message will be displayed is when a read request fails and the record needs to be read directly from the database. After the read to the database the record will be written to the coupling facility. In this case, the return and reason codes mean that the entry was not found and the following message is displayed:

```
CASECCFS - Function: Read     Rc=xxxx0008  Rs=xxxx0825
```

When a return code and reason code other than the above for a read request (entry not found) is displayed, an additional eTrust CA-ACF2 message is displayed. It will be in the format:

```
ACFCD401-FUNC: xxxxxxxx CFS RC=xxxxxxxx RS=xxxxxxxx XES RC=xxxxxxxx RS=xxxxxxxx
```

The XES return and reason codes are again found in the *IBM z/OS V1R2.0 MVS Programming: Sysplex Services Reference* guide.

If you are having a problem or are still uncertain about a problem after reviewing these return and reason codes, contact the eTrust CA-ACF2 technical support group with these messages and return codes.

The CFS return code and reason code meanings are described in the following:

CFS Return Codes

The CFS return codes are:

- 0—Completed Successfully
- 4—Warning issued
- 8—Invalid plist passed to module CASECCFS
- 12—Input validation failed
- 16—Function failed

CFS Reason Codes

The CFS reason codes are described in the following table:

Code	Description
1	Invalid function code passed to module CASECCFS, request cannot be processed.
2	Invalid work area passed to module CASECCFS, request cannot be processed.
3	Invalid Token passed to module CASECCFS, request cannot be processed.
4	Getmain failed in module CASECCFS, request cannot be processed.
5	Invalid Strname passed to module CASECCFS, structure name not defined in the coupling facility.
6	Invalid Conname passed to module CASECCFS, connect to coupling facility can not be completed.
7	Invalid number of headers passed to module CASECCFS, maximum number of headers allowed per structure exceeded.
8	Invalid entry size passed to module CASECCFS, size exceeds maximum allowed or is not a multiple that is required.
9	Invalid name field passed to module CASECCFS, request cannot be processed.

Code	Description
10	Invalid buffer address passed to module CASECCFS, request cannot be processed.
11	Invalid buffer length passed to module CASECCFS, request cannot be processed.
12	Invalid option field passed to module CASECCFS, request cannot be processed.
13	IXLCONN failed, Sysplex services cannot be completed. See the XES return and reason codes for problem.
14	IXLDISC failed, Sysplex services cannot be completed. See the XES return and reason codes for problem.
15	IXLLIST READ failed, Sysplex services cannot be completed. See the XES return and reason codes for problem.
16	IXLLIST WRITE failed, Sysplex services cannot be completed. See the XES return and reason codes for problem.
17	IXLLIST DELETE failed, Sysplex services cannot be completed. See the XES return and reason codes for problem.
18	IXLLIST LOCK failed, Sysplex services cannot be completed. See the XES return and reason codes for problem.
19	Entry is locked, Sysplex services cannot be completed. See the XES return and reason codes for problem.
20	Entry not found, request must be issued against the eTrust CA-ACF2 database.
21	Lock not held, request must be issued against the eTrust CA-ACF2 database.
22	Lock held, request must be issued against the eTrust CA-ACF2 database.
23	Lock not set, request must be issued against the eTrust CA-ACF2 database.
24	Lock not reset, request must be issued against the eTrust CA-ACF2 database.
25	MSGALL request failed by XCF processing, See the XES return and reason codes for problem.
26	GETMSG request failed by XCF processing, See the XES return and reason codes for problem.
27	GETMSG no more messages are waiting to be processed, informational only.

Code	Description
28	Invalid ECB addr passed to module CASECCFS, request cannot be processed.
29	Invalid MSG address passed to module CASECCFS, request cannot be processed.
30	Invalid MSG length passed to module CASECCFS, request cannot be processed.
31	LXCF initialize failed, Sysplex services cannot be completed. See the XES return and reason codes for problem.
32	LXCF register failed, Sysplex services cannot be completed. See the XES return and reason codes for problem.
33	LXCF query failed, Sysplex services cannot be completed. See the XES return and reason codes for problem.
34	LXCF send message error, Sysplex services cannot be completed. See the XES return and reason codes for problem.
35	LXCF getevent error, Sysplex services cannot be completed. See the XES return and reason codes for problem.
36	LXCF endevent error, Sysplex services cannot be completed. See the XES return and reason codes for problem.
37	MSGALL no connected systems to send message to, informational only.
38	LXCF deregister error, Sysplex services cannot be completed. See the XES return and reason codes for problem.
39	LXCF terminate error, Sysplex services cannot be completed. See the XES return and reason codes for problem.
40	Storage pool init error in module CASECCFS, processing cannot be completed.
41	LXCF not initialized, request to use XCF services cannot be processed. Must start SYSPLEX with XCFGROUP defined.
42	Invalid alternate Strname passed to module CASECCFS, request cannot be processed.
43	Invalid GRPNAME passed to module CASECCFS, request cannot be processed.
44	Invalid DISCONNECT reason passed to CASECCFS, request cannot be processed.
45	Connect timed out for Sysplex services, must restart the SYSPLEX feature.

Code	Description
46	Connect timed out for Sysplex services, must restart the SYSPLEX feature.
47	Structure forced disconnect, must restart the SYSPLEX feature.
48	Error during recovery, must restart the SYSPLEX feature.
49	Connect failed - Recovery error, must restart the SYSPLEX feature.
50	XES not connected, request to use XES services cannot be processed. Must start SYSPLEX with valid structure name.
51	Structure out of space, it will be cleared so that further processing can take place, informational only.
52	IXLLIST DELETE_MULT failed, probably due to a current lock held on the structure. Will be retried later.
53	Delete function timed out probably due to a current lock held. Try function at a later time.
54	Lock error during Connect detected by Sysplex services, try to connect later.
55	Connect lock contention detected by Sysplex services, try to connect later.
56	Fatal XES error detected by SYSPLEX services, must restart the SYSPLEX feature.

Chapter

24

LDAP Directory Services (LDS)

X.500 Directories have become an integral part of corporate network infrastructures. They allow for the storing and retrieval of data that needs to be centrally located in the company. User information such as email address, office or cube number, and phone extensions are just some of the items that others in the company need to access.

Not only is more and more data being stored and accessed from directories, more and more applications are now using this information rather than keeping their own copies. These applications are also being extended to authenticate user IDs and passwords to these directories to provide a centralized password repository. eTrust CA-ACF2 can be utilized as this central repository using the eTrust LDAP Server.

When it is desired to host the respository on a platform other than the mainframe or to use a repository other than eTrust CA-ACF2 such eTrust Directory, Sun iPlanet, or Novell eDirectory, an issue that needs to be resolved is how to synchronize the data between the eTrust CA-ACF2 security database and this other repository.

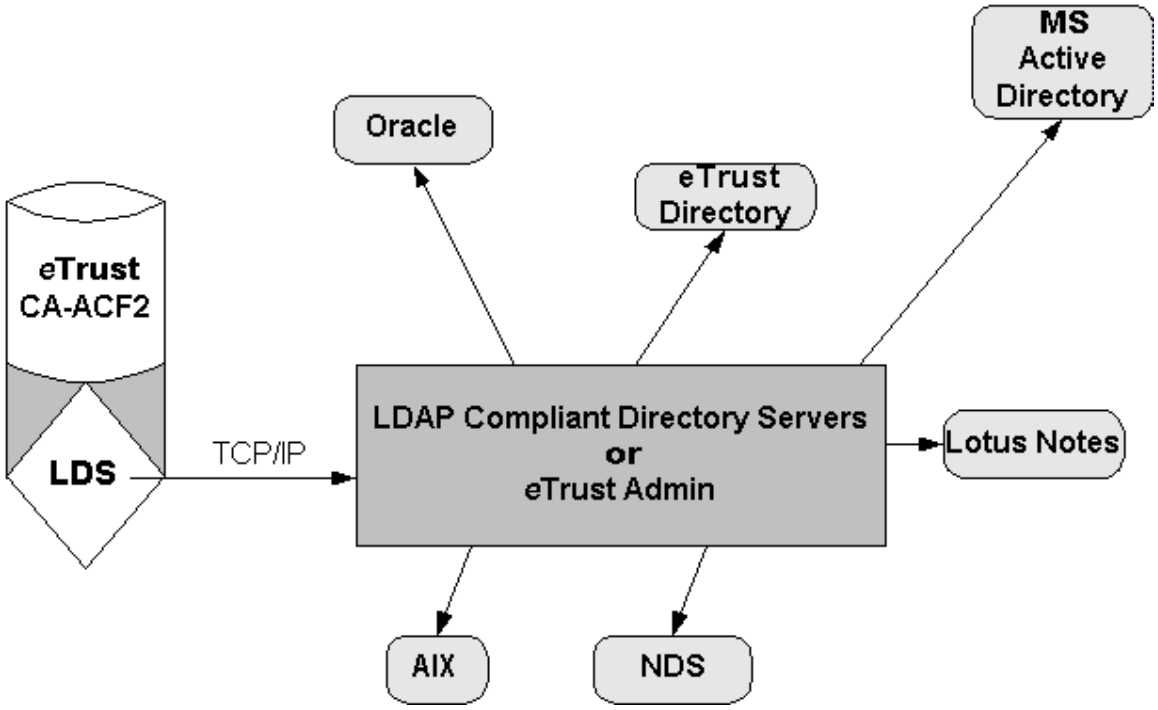
For transmitting the data from the other platform to the mainframe, the eTrust LDAP Server provides the interface that the other platform can use to send changes to eTrust CA-ACF2. For transmitting the data from the mainframe to the other respository, there is a built in feature called LDAP Directory Services (LDS). LDS is the functionality that allows eTrust CA-ACF2 to interface with and transmit data to the remote repository. Unlike other products and services that use pull technology on a scheduled basis; this is a proactive approach that pushes the change as it occurs, providing a real time update of the changed data.

How LDS Works

LDS uses the LDAP protocol and native TCP/IP to communicate the changes to the remote repository. Servers enabled with Secure Sockets Layer (SSL) technology protect unauthorized parties from viewing sensitive information during a secure session. Using the XREF mapping record, you configure which LID fields are to be sent to the remote repository and what the remote attribute name is.

Even though the field names in eTrust CA-ACF2 do not match to the remote directory attribute names, using this mapping table allows for integration with anything that supports the LDAP protocol. This eliminates the need for specialized interfaces designed for a specific product.

When eTrust CA-ACF2 uses LDS to connect to the remote LDAP directory, it is the client application to the remote LDAP Server. Using the standard LDAP protcol, eTrust CA-ACF2 formats the add, modify, or delete request and sends it to the remote LDAP Server through native TCP/IP. This remote directory can be anything that supports the LDAP APIs such as eTrust Admin, eTrust Directory, Sun iPlanet, or Novell eDirectory.



TCP/IP and SSL

eTrust CA-ACF2 LDS utilizes TCP/IP to propagate packets of information in clear text data over the internet and other TCIP/IP networks to LDAP servers that are listening. eTrust CA-ACF2 supports SSL technology and allows an SSL-enabled LDAP server to authenticate itself to eTrust CA-ACF2 and eTrust CA-ACF2 to authenticate itself to the SSL-enabled LDAP server. Highly sensitive information, such as passwords, are protected if your site chooses to exploit SSL technology.

Commands Valid for LDS

Administrative commands that INSERT, CHANGE, and DELETE logonid fields are eligible for LDS as well as password changes during system entry validation. Only logonid fields that are mapped to an LDAP directory server are eligible to be sent by LDS. All other eTrust CA-ACF2 data such as infostorage records, resource rules, and access rules are not eligible for LDS and are not transmitted to remote LDAP servers.

Note: LDS is not a synchronization facility for multi-valued field data. For example, if in a CHANGE command a REPLACE is issued for a multi-valued field, LDS will notify the LDAP directory server of the specific values INSERTED or DELETED from the multi-valued field list. If the data between the mapped multi-valued fields are not synchronized prior to the LDS update, it may remain as such. Therefore, a CHANGE request for a logonid field that contains multiple values may result in different values for the multi-valued field on the mapped LDAP directory entry.

Records Summary

Each record has a unique set of fields. These predefined record Ids are listed in the following table, together with their basic functions.

Record ID	Function
OPTIONS	Defines global LDS options available
LDAP	Contains LDAP server information
XREFLDAP	Allows mapping of LDAP attributes to eTrust CA-ACF2 logonid fields

Enabling the LDS Interface

The Control GSO OPTS record defines global options available to eTrust CA-ACF2. The LDS parameter on the Control GSO OPTS record indicates that the LDS interface can be used.

Modify the Control GSO OPTS Record

To change the Control GSO OPTIONS record to include LDS, use the following commands:

```
ACF
  ACF
SET CONTROL(GSO)
  CHANGE OPTS LDS
```

Refresh the Control GSO OPTS Record

To refresh the Control GSO OPTS record, use the following operator command:

```
F ACF2,REFRESH(OPTS),TYPE(GSO)
```

Maintaining Control LDS Records

The following explains how to create, modify, refresh, LDS records. It also describes how to start and stop LDS.

Control LDS OPTIONS Record

The Control LDS OPTIONS record defines global system options available to LDS. It also specifies the name of an outbound journal file and an attribute that specifies whether the journal process will start when LDS starts.

When LDS starts and DEBUG is indicated on the options record, eTrust CA-ACF2 spins off diagnostic output logs to the following DDNAMEs: CEEDUMP, SYSOUT, SYSPRINT, STDERR, and SYDOUT. The sysout class and destination for the diagnostic log output are specified by SYSCLASS and SYSDEST field respectively.

Note: The diagnostic output logs and the LDS Journal are dynamically allocated and opened by eTrust CA-ACF2 and should not be overridden in the eTrust CA-ACF2 started procedure.

Record ID	Fields
OPTIONS	DATASET(*journal file data set name*) DEBUG | NODEBUG JOURNAL | NOJOURNAL LDSRCVR(*recovery file data set name*) LDSRING(*keyring ringname*) RECOVERY | NORECOVERY RETRY(*nn*) SYSCLASS(*A* | *class*) SYSDEST(*LOCAL* | *destination*) TIMEOUT(*30* | *sss*)

Field Descriptions

DATASET(*journal file data set name*)
Defines the name of the journal file that contains the loggings of the LDS requests. The data set must be predefined and cataloged.

DEBUG | NODEBUG
Specifies that diagnostic messaging is captured during processing to the LDAP server and directed to diagnostic output logs. The default is NODEBUG.

JOURNAL | NOJOURNAL
Specifies whether the LDS Journal processing is started when LDS starts. The use of the LDS journal file is optional. You can choose to run without the journal file by setting the NOJOURNAL attribute on the Control LDS OPTIONS record. The specification here is a global setting. If you specify the JOURNAL option on the Control LDS LDAP record, you can control journaling for each server. The default is NOJOURNAL.

LDSRCVR(*recovery file data set name*)
Specifies the name of the file used by the LDS recovery process at LDS initialization. It is a 44 byte character data set name. The LDS recovery file is created, initialized, and catalogued prior to starting the LDS recovery processing. You may use the supplied INITLDSR job in the SAMPJCL library to create the LDS recovery file.

LDSRING(*keyring ringname*)
Specifies the ring name of the KEYRING record to which the CERTAUTH and PERSONAL certificate information for LDS processing is obtained. The userid associated with the KEYRING record must be ACF2. The ringname can be up to 237 characters in length. **Note:** This field is only valid when establishing SSL connections.

RECOVERY | NORECOVERY
Specifies that LDS recovery processing will be enabled at LDS intialization. LDSRCVR must also be specifies for recover to be enabled. The default is RECOVERY.

RETRY(*nn*)
Indicates the number of consecutive, unsuccessful attempts to deliver an LDS command to an ACTIVE LDAP server before the LDAP node will be DEACTIVATED by eTrust CA-ACF2. Valid values are 1 through 10. The default is 3.

Note: When an LDAP node exceeds the RETRY and TIMEOUT limits indicated in the GSO OPTS record, LDS will DISCONNECT from the LDAP server. Issue the LDS SHOW command to display the STATUS of the LDAP nodes in the system. If the STATUS is DISCONNECTED, you must issue the operator command 'F ACF2,LDS(ACTIVE),LDAPNODE(ldapnode)' to re-establish a connection handle to the LDAP server.

SYSCLASS(*A* | *class*)
Specifies the JES SYSOUT class for diagnostic log output. SYSCLASS is a 1-byte character alphanumeric field. This option cannot be refreshed and can only be activated at LDS startup. The default value is A.

SYSDEST(*LOCAL* | *destination*)
Specifies the JES SYSOUT destination for diagnostic log output. SYSDEST is a 1-8 byte character field consisting of alphanumeric or national characters. This option cannot be refreshed and can only be activated at LDS startup. The default value is LOCAL.

TIMEOUT(*30* | *sss*)
Indicates the number of seconds that eTrust CA-ACF2 will wait to receive a response from the LDAP server before the LDAP request will be considered as timed out. Valid values are 1 through 999. The default is 30 seconds.

Note: When an LDAP node exceeds the RETRY and TIMEOUT limits indicated in the GSO OPTS record, LDS will DISCONNECT from the LDAP server. Issue the LDS SHOW command to display the STATUS of the LDAP nodes in the system. If the STATUS is DISCONNECTED, you must issue the operator command 'F ACF2,LDS(ACTIVE),LDAPNODE(ldapnode)' to re-establish a connection handle to the LDAP server.

Note: A refresh of the LDS OPTIONS record is automatically issued during LDS initialization. You may issue a refresh of the LDS OPTIONS record at any time LDS is active but only DEBUG, RETRY, and TIMEOUT will be updated.

Create the Control LDS OPTIONS Record

Use the following commands to create the control LDS OPTIONS record:

```
ACF
  SET CONTROL(LDS) SYSID(TEST)
CONTROL
  insert options journal dataset(caldap.ldsjrnl) recovery ldsrcvr(caldap.ldsrcvr)
  debug timeout(15)
TEST / OPTIONS LAST CHANGED BY ADMIN01 ON 01/31/04-16:46
               DATASET(CALDAP.LDSJRNL) DEBUG JOURNAL
               LDSRCVR(CALDAP.LDSRCVR) RECOVERY RETRY(3) LDSRING()SYSCLASS(A)
               SYSDEST(LOCAL) TIMEOUT(15)
CONTROL
```

Refresh the Control LDS OPTIONS Record

LDS must be active to refresh LDS type records. To refresh the Control LDS OPTIONS record, use the following operator command:

```
F ACF2,REFRESH(OPTIONS),TYPE(LDS)
```

Note: When LDS starts up, all records are automatically refreshed.

LDS Journal

LDS uses the journal file to provide a historical record of outbound LDAP requests to LDAP servers. LDS uses one journal file for outbound requests to all LDAP servers. All requests and corresponding messages are journaled.

The LDS journal file is defined in the Control LDS OPTIONS record in the DATASET parameter. The journal file must be created, cataloged, and initialized prior to starting the LDS journal processing. Use the supplied INITLDSJ job in the SAMPJCL library to create the LDS journal files. The number of journal entries written to the LDS journal is limited by the size of the file. When the journal file becomes full, the LDS journal wraps to the top of the file and overwrites older journal entries.

The data set is dynamically allocated by eTrust CA-ACF2. Do not add it to the eTrust CA-ACF2 started procedure.

The use of LDS journal file is optional. You can choose to run without the journal file by setting the NOJOURNAL attribute on the LDS OPTIONS record or you may specify the NOJOURNAL option on the LDS LDAP RECORD. You may start and stop the journal file processing at any time as long as LDS is active.

Start and Stop the LDS Journal

Use the following operator command to start the LDS journal when LDS is active:

```
F ACF2,LDS(JOURNAL)
```

Use the following operator command to stop the LDS journal when LDS is active:

```
F ACF2,LDS(NOJOURNAL)
```

LDS Recovery

LDS uses the recovery file to store LDS request data for each defined LDAP server in the network. If an LDAP server fails to communicate with LDS due to the LDAP server being down, exceeding the configured time-out or a busy condition, LDS requests are saved in the recovery file for later transmission. As transaction data is processed, the data is removed from the recovery file and written to the LDS journal file if LDS JOURNAL is active.

To activate LDS recovery at LDS startup, the RECOVERY parameter must be indicated and the LDSRCVR parameter must specify a valid LDS recovery file on the LDS OPTIONS record. The recovery file must be created, cataloged, and initialized prior to starting LDS. Use the supplied INITLDSR job in the SAMPJCL library to create the LDS recovery file. The number of recovery entries written to the LDS recovery file is limited by the size of the file. When the recovery file becomes 80% full and 100% full, a message is displayed on the operator console. The LDS recovery file does not wrap to the top of the file and overlay older data. When the recovery file becomes full, transaction data can no longer be saved on the recovery file.

The use of the LDS recovery file is optional. You can choose to run without the recovery file by setting the NORECOVERY option on the LDS OPTIONS record. Recovery can only be activated at LDS initialization.

Note: The data set is dynamically allocated by eTrust CA-ACF2. Do not add it to the eTrust CA-ACF2 started procedure.

Removing LDS Recovery File Records

To delete LDS Recovery Records from the LDS Recovery File by LDAPNODE or a given date, enter the following command:

```
F ACF2,LDS(REMOVE), LDAPNODE(LDAP.xxxxxxxx), UNTIL(date)
```

The UNTIL parameter must be a valid date in the default date format specified by the eTrust CA-ACF2 options. The LDAPNODE parameter must indicate the complete LDS LDAP Record ID including suffix, for example:

```
F ACF2,LDS(remove), LDAPNODE(ldap.test), UNTIL(01/01/04)
```

Note: You may specify both LDAPNODE and UNTIL parameters, but at least one must be indicated. Also, LDS must be active for this command to be accepted.

LDSRPT Report

The LDSRPT report lists all LDS requests stored in the LDS Recovery File. LDS recovery retrieves records containing information pertaining to administrative commands that INSERT, CHANGE, and DELETE logonid fields as well as password changes that are eligible for LDS processing. There are no REPORT parameters for this program. For more information about the LDSRPT, see the *Reports and Utilities Guide*.

Control LDS LDAP Record

The Control LDS LDAP record defines the LDAP servers in the network and information required to appropriately communicate logonid administrative changes.

Information required to write, update, or delete entries in the LDAP directory is defined in the LDS LDAP record. This information includes:

- Administrator distinguished name (DN)
- Administrator password
- URL of the LDAP server
- Name of the objectClass of the entry on the LDAP directory
- Command type of logonid administration
- Map of the LDAP attributes to eTrust CA-ACF2 logonid record fields
- Additional options

Record ID	Fields
LDAP.*qualifier*	ACTIVE | NOACTIVE ADMINDN *(LDAP administrator distinguished name*) ADMPSWD (*LDAP administrator password*) APPLNAME (*application name*) BITDEFLT (*character/YN* | *field type/format*) BROADCST | NOBROADCST CHANGE | NOCHANGE DATEFMT(*date format* | *MMDDYYYY*) DEBUG | NODEBUG DELCHILD | NODELCHILD DELETE | NODELETE DELIMIT(*character*) INSERT | NOINSERT JOURNAL | NOJOURNAL LDSLABEL(*certificate label*) LDAPINST(*AttributeName/AttributeValue/AttributeType,…)* NEXTKEY(*qualifier of next LDS LDAP record*) OBJCLASS(*LDAP objectClass*) PSWDASIS | NOPSWDASIS URL(*Uniform Resource Locator*) USEEXTOP | NOUSEEXTOP USERDNS(*User Distinguished Name syntax*) XREF(*LIDfield1/LDAPattribute1Name/ LDAPattribute1FieldType/ LDAPattribute1DataFormat/LDAPattribute1Length, …*)

Field Descriptions

LDAP.*qualifier*

Must be 1 to 8 characters long. **Note:** The period **is** required.

ACTIVE | NOACTIVE

Indicates the LDAP record is active and communication with the LDAP server specified will be attempted. **Note:** To activate a LDAP record, INSERT, CHANGE, or DELETE must also be specified. The default is NOACTIVE.

ADMINDN(*LDAP administrator distinguished name*)
Indicates the LDAP administrator distinguished name used for binding to the LDAP server and administering the LDAP request. This is a required field. The following examples indicates the string substitutions allowed for this field:

Symbolic	String Substitution
%N	Substitutes the administrator's name (20 bytes) from the administrator's LID record
%L	Substitutes the administrator's logonid (8 bytes) from the administrator's LID record

Note: If there are embedded spaces or commas in the ADMINDN, enclose the entire string in single quotes.

Note: The LDAP administrator password or the APPLNAME field must be specified. Each LDAP request requires an administrator distinguished name and a password. There are two ways of providing the password:

- Specify the administrator password in the Control LDS LDAP record.
- Identify the administrator and the SSIGNON profile data record used for generating PassTickets by specifying the APPLNAME in the Control LDS LDAP record.

ADMPSWD(*LDAP administrator password*)
Indicates the LDAP administrator password used with the ADMINDN for binding to the LDAP server. This field cannot be modeled. This field is 128 bytes long and allows mixed case characters.

APPLNAME(*application name*)
Specifies the application name of the SSIGNON profile data record that contains the administrator's encryption key used for generating PassTickets. The administrator's userid is used with the PassTicket for binding to the LDAP server and administering the LDAP request. For more information about the SSIGNON profile data records, see the "Maintaining Profile Records" chapter in the Administrator Guide.

BITDEFLT(*field type/format*|character/*YN*)
Indicates the default field type and format for bit fields specified in the XREF field that will be sent to the LDAP server. **Note:** When a logonid bit field is turned off and the FLAGS attribute specifies NEVER, LIMIT or NULL in the ACCFDR @CFDE macro definition, LDS interprets this condition as a DELETE function of the bit field rather than a REPLACE. The default for this field is CHARACTER/YN. Available options are:

Options	
BINARY/01	Binary 1 or 0 when the bit is ON or OFF, respectively
BINARY/REV01	Binary 0 or 1 when the bit is ON or OFF, respectively
CHARACTER/01	1 or 0 when the bit is ON or OFF, respectively
CHARACTER/REV01	0 or 1 when the bit is ON or OFF, respectively
CHARACTER/YN	Y or N when the bit is ON or OFF, respectively
CHARACTER/REVYN	N or Y when the bit is ON or OFF, respectively
CHARACTER/TF	T or F when the bit is ON or OFF, respectively
CHARACTER/REVTF	F or T when the bit is ON or OFF, respectively

Note: This default can be overridden per bit field specified in the XREF parameter. See the XREF field for more information.

BROADCST|NOBROADCST
Specifies to capture all logonid administrator commands that are indicated for this LDAP record (for example, INSERT, CHANGE, and DELETE) regardless if the LID record affected is flagged for LDS propagation. The default is NOBROADCST.

CHANGE|NOCHANGE
Specifies that command LID change processing is propagated to the LDAP server. The default is NOCHANGE.

DATEFMT(*date format*|*MMDDYYYY*)
Indicates the default format for date fields specified in the XREF field. Valid date formats are MMDDYYYY, DDMMYYYY, YYYYMMDD, MMDDYY1, DDMMYY1, and YYMMDD1.

For example:

- *MM*–Represents a two-digit month.
- *DD*–Represents a two-digit day.
- *YY*–Represents a two-digit year.
- *YYYY*–represents a four-digit year.
- *Number 1*–Represents a '/' (forward slash) delimiter in the date field.

The default date format is MMDDYYYY. Year designations of 70-99 assume a date in the 20th century (1970-1999); year designations of 00-69 assume a date in the 21st century (2000-2069).

For example, if the ACTIVE date for the logonid record is April 7, 2001, the date field is converted to the following formats:

DATEFMT	Format Example
MMDDYYYY	04072001
DDMMYYYY	07042001
YYYYMMDD	20010407
MMDDYY1	04/07/01
DDMMYY1	07/04/01
YYMMDD1	01/04/01

Note: The default date format (MMDDYYYY) can be overridden per date field specified in the XREF parameter. See the XREF field for further information.

DEBUG|NODEBUG
Bit Option field turns tracing on during LDS processing to this node. The default is NODEBUG.

DELCHILD|NODELCHILD
Bit Option field indicates to delete child objects when a delete command of the base object is issued to this node. The default is NODELCHILD.

DELETE|NODELETE
Specifies that eTrust CA-ACF2 command LID delete processing will be propagated to the LDAP server. The default is NODELETE.

DELIMIT(*character*)
Indicates a one byte character date delimiter used in conjunction with the DATEFMT field to format additional output date value. Only date formats MMDDYY, DDMMYY, YYMMDD, MMDDYYYY, DDMMYYYY, and YYYYMMDD can be specified with one of the following valid DELIMIT characters: dash, colon, forward slash, period, coma, and space. The following table lists examples of the formatted output date for April, 28, 2001 with a valid DELIMIT and DATEFMT specified:

DELIMIT	DATEFMT	Example
-	MMDDYY	04-28-01
:	DDMMYY	28:04:01

DELIMIT	DATEFMT	Example
/	YYMMDD	01/04/28
.	MMDDYYYY	04.28.2001
,	DDMMYYYY	28,04,2001
‘ ‘	YYYYMMDD	2001 04 28

INSERT | NOINSERT
Specifies that eTrust CA-ACF2 command LID insert processing is propagated to the LDAP server. The default is NOINSERT.

JOURNAL | NOJOURNAL
Specifies that requests to this server will be journalled if LDS journal is active. The default is JOURNAL.The use of the LDS journal is optional.

LDSLABEL(*certificate label*)
Specifies the label of the PERSONAL certificate that is connected to the eTrust CA-ACF2 LDSRING indicated on the LDS OPTIONS record. The label can be up to 32 characters in length. It can contain blanks and mixed-case characters.

LDAPINST(*AttributeName/AttributeValue/AttributeType,...*)
Maps LDAP installation specific information that will be appended to all insert and change LDS requests only for this node. This is used when you always need to send an attribute/value pair no matter what was changed or inserted on the LID.

NEXTKEY(*qualifier of LDS XREFLDAP record*)
Specifies the qualifier name of the LDS XREFLDAP record that contains more XREF field values to be mapped to this LDAP server. The Control LDS XREFLDAP record provides the capability to define additional eTrust CA-ACF2 logonid fields and the cross-references to the LDAP attribute names.

OBJCLASS(*LDAP objectClass*)
When an INSERT is sent as an LDAP add, this specifies the objectClass of the entry that is to be added to the remote LDAP directory. The maximum length is 24 characters.

PSWDASIS | NOPSWDASIS
Indicates the case sensitivity format of the user's password to be propagated, if the password field is specified in the XREF field. If PSWDASIS is specified, the password is sent in the same case as typed by the user or administrator. In this case, the password can consist of mixed case text. If NOPSWDASIS is specified, the password will be propagated in uppercase format. The default is PSWDASIS.

Note: When a user changes a password during system entry validation, LDS automatically propagates the new password to the LDAP servers interested in the password field. The user receives no indication that LDS processing was involved. LDS must be active and the LDS option must be specified in the logonid record. See LDS Logonid Field.

URL(*Uniform Resource Locator*)
Specifies the Uniform Resource Locator (URL) used to identify the LDAP server. The syntax of the LDAP URL should be the following:

```
ldap://host.domain:port
```

or

```
ldaps://host.domain:port
```

ldap/ldaps
Specifies a connection using the LDAP protocol. 'ldap' starts an unencrypted connection, where 'ldaps' starts an encrypted connection.

host.domain
Specifies the name or IP address of the remote LDAP server. This defaults to localhost.

port
Specifies the port number of the remote LDAP server. This defaults to 389.

USEEXTOPT|NOUSEEXTOP
Use this option when a remote LDAP Server does not support the ldaps URL to establish an SSL connection. After configuring the URL with ldap above, enable this option and LDS will establish the unencrypted connection and then try to turn it into an encrypted connection using the LDAP extended operation function call. The default is NOUSEEXTOP.

USERDNS(*User Distinguished Name Syntax*)
Indicates the user distinguished name syntax that refers to the entry on the LDAP server where the changes will be applied. This is a required field. The maximum length is 255 characters.

The following indicates the string substitutions allowed for this field:

Symbolic	String Substitution
%N	Substitute the user's name (20 bytes) from the user's LID record For example: USERDNS('acf2lid=%N, host=prod, o=company, c=usa')
%L	Substitute the user's logonid (8 bytes) from the user's LID record. For example: USERDNS ('acf2lid=%L, host=prod, o=company, c=usa')

Note: If there are embedded spaces or commas in the USERDNS, enclose the entire string in single quotes.

XREF(*LIDfield1/LDAPattribute1Name/LDAPattribute1FieldType/LDAPattribute1DataFormat/LDAPattribute1Length, ...*)
Specifies the names of the eTrust CA-ACF2 logonid fields and the corresponding LDAP directory attributes to be synchronized to the LDAP directory for insert or change command processing. The XREF field must contain a valid mapping if INSERT or CHANGE is indicated on the LDAP record.

- LIDfield—Specifies the external logonid field name from eTrust CA-ACF2 logonid record that is mapped to the LDAP attribute name on the LDAP directory. For a list of external field names, see the "Maintaining Logonid Records" chapter.

  Note: The LIDfield is a required parameter and must be delimited by a '/' (forward slash).

The logonid record contains some fields that are used internally by eTrust CA-ACF2 processing and not modified directly by a security administrator or user. eTrust CA-ACF2 logonid fields that are not inserted or modified by eTrust CA-ACF2 command administration or by a user are not supported by LDAP and cannot be specified as an XREF field parameter. These fields include:

Logonid Field	Description
ACC-CNT	Count of system accesses
ACC-DATE	Date of last system access
ACC-SRCE	Source of last system access
ACC-TIME	Time of last system access
CSDATE	Cancel/suspend/mon date
CSWHO	User with set cancel/suspend/mon
HOMENODE	Node where lid is stored
PRVPSWD1	Password history 1
PRVPSWD2	Password history 2
PRVPSWD3	Password history 3
PRVPSWD4	Password history 4
PRV-TOD1	Date/Time of PRVPSWD1
PRV-TOD2	Date/Time of PRVPSWD2
PRV-TOD3	Date/Time of PRVPSWD3
PRV-TOD4	Date/Time of PRVPSWD4
PSWD-SRC	Password source

Logonid Field	Description
PSWD-TIM	Password time
PSWD-TOD	Password time of date
PSWD-XTR	Halfway encrypted password flag
UPD-TOD	Date of update

- **LDAPattributeName**—Specifies the LDAP attribute of the entry on the LDAP directory that corresponds to the logonid field name on the eTrust CA-ACF2 logonid record. The attribute name can contain only alphanumeric characters and dashes.

  Note: The LIDfield and LDAP attribute name are required parameters and must be delimited by '/' (forward slash).

The following parameters are optional and can also be specified.

- **LDAPattributeFieldType**—Specifies the field type of the LDAP attribute. Valid field types are BINARY, CHARACTER, and DATE. If this parameter is not specified, the default field type is CHARACTER. This field indicates to eTrust CA-ACF2 how to convert the field data before transmitting it to the LDAP server. For example, LID character fields are transmitted as character strings only. LID binary and LID hexadecimal fields can be transmitted as either character or binary data. LID bit fields can be transmitted as either character or binary data. LID date fields can be transmitted as character date fields.
- **LDAPattributeDataFormat**—Specifies the data format of the LDAP attribute name. If this parameter is not specified, the default data type is taken from the DATEFMT field if this is a LID date field or from the BITDEFLT field if this is a LID bit field. If this parameter is indicated, then LDAPattributeFieldType parameter must be specified.

The following are examples of LDAP attribute field type/data formats for eTrust CA-ACF2 LID date fields with a null DELIMIT field specified on LDS OPTIONS record:

LDAP Attribute Field Type	LDAP Attribute Data Format	Example Date
DATE	MMDDYYYY	April 7, 2001 converts to 04072001
DATE	DDMMYYYY	April 7, 2001 converts to 07042001
DATE	YYYYMMDD	April 7, 2001 converts to 20010407
DATE	MMDDYY1	April 7, 2001 converts to 04/07/01
DATE	DDMMYY1	April 7, 2001 converts to 07/04/01
DATE	YYMMDD1	April 7, 2001 converts to 01/04/07

Bit fields are ON if the specific privilege or authority has been granted and OFF if not specified or granted. The LID fields can be converted to character 'Y', 'N', 'T', 'F', '0', '1', binary 0 or 1.

The following are valid LDAP attribute field type/data formats for eTrust CA-ACF2 LID bit fields:

LDAP Attribute Field Type	LDAP Attribute Data Format	BIT is ON	BIT is OFF
CHARACTER	YN	'Y'	'N'
CHARACTER	REVYN	'N'	'Y'
CHARACTER	TF	'T'	'F'
CHARACTER	REVTF	'F'	'T'
CHARACTER	01	'1'	'0'
CHARACTER	REV01	'0'	'1'
BINARY	01	X'1'	X'0'
BINARY	REV01	X'0'	X'1'

- **LDAPattributeLength**—A number from 1 to 1024 that represents the maximum data length of the LDAP attribute value. If this is not specified, the default length is the eTrust CA-ACF2 logonid field length from which the LDAP attribute has been mapped.

  Note: Optional fields are positional and must be delimited by a '/' (forward slash).

Note the following:

If LDS is started and there are no LDAP records defined, LDS will automatically shutdown.

Likewise, if LDS is running and a refresh of LDAP records is issued with no LDAP records defined as active, LDS will automatically shutdown.

Create the Control LDS LDAP Record

The following is an example of creating an LDS LDAP record.

```
acf
  acf
set control(lds)
insert ldap.cpu2 active insert delete pswdasis objclass(acf2lid)
url('ldap://nnnn.nnn.nnn.nnn:389')
userdns('acf2lid=%1, host=cpu2, o=cai, c=usa')
admindn('acf2lid=admin') admpswd(password) xref(name/Name) nextkey(cpu2xrf)
 JGP  /  LDAP.CPU2 LAST CHANGED BY FRANK01 ON 11/27/01-15:42
               ACTIVE ADMINDN(acf2lid=admin) APPLNAME()
               BITDEFLT(CHARACTER/YN) NOBROADCST NOCHANGE DATEFMT(MMDDYYYY)
               DELETE INSERT LDSLABEL NEXTKEY(CPU2XRF) OBJCLASS(acf2lid) PSWDASIS
               URL(ldap://nnnn.nnn.nnn.nnn:389)
               USERDNS(acf2lid=%1, host=cpu2, o=cai, c=usa) XREF(NAME/Name)
```

Refresh the Control LDS LDAP Record

To activate the LDAP directories and refresh the Control GSO LDAP and Control LDS XREFLDAP records, issue the following command:

```
F ACF2,REFRESH(LDAP),TYPE(LDS)
```

Note: The REFRESH command works only when the LDS facility has been activated.

Control LDS XREFLDAP Record

The Control LDS XREFLDAP record defines additional eTrust CA-ACF2 logonid fields and the cross-references to the LDAP attribute names. Recall that the XREF field in the Control LDS LDAP record ties the eTrust CA-ACF2 logonid record field to LDAP attribute name in the LDAP directory.

The Control LDS XREFLDAP record is intended to be an extension of the XREF field of the Control LDS LDAP record. The XREF field ties the logonid record field to the attributes of objects in the LDAP directory.

The NEXTKEY field references the qualifier name of the Control LDS XREFLDAP record that indicates that additional fields are to be propagated. Create a Control LDS XREFLDAP record if the Control LDS LDAP record requires additional XREF field information.

Record ID	Fields
XREFLDAP.*qualifer*	NEXTKEY(*qualifier of next LDS XREFLDAP record*) XREF(*LIDfield1/LDAPattribute1Name/ LDAPattribute1FieldType/LDAPattribute1FieldFormat/ LDAPattribute1FieldLength*)

Field Descriptions

XREFLDAP.*qualifier*
Must start with a period and must be one to seven characters long.

NEXTKEY(***qualifier of next LDS XREFLDAP record***)
Specifies the qualifier of the next Control LDS XREFLDAP record that contains more XREF field values to be mapped to the requesting LDAP server.

XREF(*LIDfield/LDAPattribute1Name/LDAPattribute1FieldType/ LDAPattribute1DataFormat/LDAPattribute1Length*)
Specifies the name of the eTrust CA-ACF2 logonid fields and the corresponding LDAP directory attributes to be synchronized to the LDAP directory for insert or change eTrust CA-ACF2 logonid command processing. The LIDfield and LDAPattributeName are required parameters and must be delimited by '/' (forward slash). See the LDAP records section for more information.

The following parameters are optional and can also be specified:

- LDAPattributeFieldType
- LDAPattributeDataFormat
- LDAPattributeLength

Note: Optional fields are positional and must be delimited by a '/' (forward slash). In addition, there must be at least one XREF field parameter defined in the XREFLDAP record.

Create the Control LDS XREFLDAP Record

Use the following commands to create the XREFLDAP record:

```
acf
  acf
set control(lds)
insert xrefldap.cpu2xrf xref(password/userPassword, phone/Phone)
 JGP / XREFLDAP.CPU2XRF LAST CHANGED BY FRANK01 ON 11/27/01-15:45
                      NEXTKEY() XREF(PASSWORD/userPassword PHONE/Phone)
```

Refresh the Control LDS XREFLDAP Record

The LDAP refresh process builds the LDAP record table with Control GSO LDAP and Control LDS XREFLDAP records. LDAP records must meet the following criteria to propagate logonid changes to networked LDAP servers.

- LDAP records must be ACTIVE with the INSERT, CHANGE, or DELETE command indicated to propagate logonid administration. If INSERT or CHANGE is specified on the Control LDS LDAP record, the XREF field data must be specified for information to be propagated
- If the NEXTKEY field is specified on the Control LDS LDAP or Control LDS XREFLDAP record, the Control LDS XREFLDAP record with the qualifier specified in the NEXTKEY field must exist.
- The NEXTKEY field can chain a maximum of twenty-five different Control LDS XREFLDAP records to a single Control LDS LDAP record. NEXTKEY field values reference Control LDS XREFLDAP records and cannot be duplicated in the sequence.

To activate the LDAP directories and refresh the Control GSO LDAP and Control LDS XREFLDAP records, issue the following command:

```
F ACF2,REFRESH(LDAP),TYPE(LDS)
```

Note: The REFRESH command works only when the LDS facility has been activated.

Enabling SSL

If an SSL connection is to be established between eTrust CA-ACF2 and an LDAP server, ensure the following steps have been completed before starting LDS.

- Generate the digital certificates from a Certificate Authority such as eTrust CA-ACF2
- Configure eTrust CA-ACF2 to obtain the generated digital certificates utilized for SSL
- Configure the LDAP server for SSL with the generated digital certificates

This section provides a sample list of steps used to setup SSL support for eTrust CA-ACF2 LDS.

Step 1

The following is an example of the eTrust CA-ACF2 commands used to generate the certauth, sitecert and personal certificates used by the LDAP.TEST node for SSL:

```
acf

gencert certauth.acfca
  expire(12/12/12)
  label(acf2=certauth)
  subjsdn(cn=acf2-certauth,ou=esac,o=ca,c=us)

gencert sitecert.acf2ca
  expire(12/12/12)
  label(site-cert-ldap)
  signwith(certauth.acf2ca)
  subjsdn(cn=site-cert-ldap,ou=esac,o=ca,c=us)

gencert acf2.acf2ca
  expire(12/12/12)
  label(acf2-user-cert)
  signwith(certauth.acf2ca)
  subjsdn(cn=acf2-user-cert,ou=esac,o=ca,c=us)
```

Step 2

The following is an example of the eTrust CA-ACF2 commands used to export the digital certificates to a data set for SSL configuration on the LDAP server.

```
export acf2.acf2ca
  dsname('acf2ca.usercert')
  label(acf2-user-cert)
  password(password)

export certauth.acf2ca
  dsname('acf2ca.certauth')
  label(acf2-certauth)
  password(password)

export sitecert.acf2ca password(password)
  dsname('acf2ca.sitecert')
  label(site-cert-ldap)
  password(password)
```

Step 3

The following is an example of the eTrust CA-ACF2 commands used to create the ACF2.LDS KEYRING record and connect the digital certificates for SSL.

```
acf

set profile(user) div(keyring)

insert acf2.lds
 ringname(acf2-lds-keyring)

connect certdata(certauth.acf2ca)
 keyring(acf2.lds)

connect certdata(acf2.acf2ca)
 keyring(acf2.lds)

connect certdata(sitecert.acf2ca)
 keyring(acf2.lds)
```

Step 4

The following is an example of the eTrust CA-ACF2 commands used to create the LDS OPTIONS record that points to the ACF2.LDS KEYRING record for SSL.

```
insert options
 retry(3) sysclass(A) sysdest(LOCAL) timeout(30) ldsring(acf2-lds-keyring)
 journal dataset(uniscax1.ldsjrnl) recovery ldsrcvr(uniscax1.ldsrcvr)
```

Step 5

The following is an example of the eTrust CA-ACF2 commands used to create the LDS LDAP record LDAP.TEST that points to the digital certificate acf2-user-cert which is connected to the ACF2.LDS KEYRING record.

```
insert ldap.test
 active debug admindn('cn=Manager,c=us') admpswd(secret) insert change delete
 ldslabel(acf2-user-cert) objclass(inetOrgPerson)
 userdns(‘cn=%l,ou=esac,o=ca,c=us') url(LDAPS://123.456.78.99:636)
 xref(name/cn,NAME/sn,PASSWORD/userPassword)
```

Start LDS to obtain all the certificates connected to the ACF2.LDS KEYRING during initialization and establish an SSL encrypted connection to the LDAP.TEST server.

```
F acf2,lds(start)
```

Starting LDS

To start LDS, use the following command:

```
F ACF2,LDS(START)
```

Note: The Control LDS OPTIONS, LDAP, AND Control LDS XREFLDAP records are automatically refreshed during initialization of LDS.

Stopping LDS

To stop LDS, use the following command:

```
F, ACF2,LDS(STOP)
```

LDS Logonid Field

The LDS logonid field specifies whether logonid administrative changes for a user are propagated to the LDAP servers defined by Control LDS LDAP and Control LDS XREFLDAP records. The LDS field also indicates that an LDAP request will be created when the user changes their password during the signon process.

Create LDS Logonid Field

Use the following command to add LDS capability to a logonid:

```
acf
  acf
change user01 lds
```

To display the changed logonid, use the following command:

```
list user01
  USER01             USER01
  PRIVILEGES           ACCOUNT CICS JOB SECURITY TSO LDS
  ACCESS               ACC-CNT(4) ACC-DATE(11/19/01) ACC-SRCE(A55TG003)
                       ACC-TIME(15:43)
  PASSWORD             PSWD-DAT(00/00/00) PSWD-INV(0) PSWD-TOD(11/07/01-11:48)
                       PSWD-VIO(0) PSWD-XTR
  TSO                  CHAR(BS) DFT-PFX(FRANK01) DFT-SOUT(A) DFT-SUBM(A)
                       INTERCOM JCL LGN-ACCT LGN-MSG LGN-PERF LGN-PROC LGN-RCVR
                       LGN-TIME LGN-UNIT LINE(ATTN) MAIL MODE MSGID NOTICES
                       TSOFSCRN TSORBA(000096) TSORGN(4,096) TSOSIZE(4,096) WTP
  STATISTICS           SEC-VIO(0) UPD-TOD(11/19/01-15:43)
  CICS                 CICSCL(404040) CICSKEY(FFFFFF) CICSKEYX(FFFFFFFFFF)
  RESTRICTIONS         PREFIX(USER01)
```

Note: LDS is located in the privileges section.

SHOW LDS Subcommand

The SHOW LDS subcommand displays the LDS status and options, the active LDAP records, and the eTrust CA-ACF2 logonid field information that is propagated to an LDAP server.

The SHOW LDS command uses standard eTrust CA-ACF2 masking conventions to specify a range of active LDAP records. The following example shows the results after issuing the SHOW LDS command with a MASKED qualifier value.

```
acf
show lds(cpu2-)


-- LIGHTWEIGHT DIRECTORY ACCESS PROTOCOL --

    CURRENT LDS STATUS:          ACTIVE
    CURRENT LDS JOURNAL STATUS:  ACTIVE
    LDS JOURNAL DATASET NAME:    CALDAP.LDSJRNL
    CURRENT LDS RECOVERY STATUS: ACTIVE
    LDS RECOVERY DATASET NAME:   CALDAP.LDSRCVR

    CURRENT SYSID:               TEST
    LDSRING:
    OPTIONS:     DEBUG    RETRY(3)  SYSCLASS(A)  SYSDEST(LOCAL)    TIMEOUT(30)

    ACTIVE LDAP RECORD LIST:

    - LDAP.CPU2
      STATUS:    CONNECTED
      OPTIONS:   ACTIVE    BROADCST    CHANGE    DEBUG       DELCHILD
                 DELETE    INSERT      JOURNAL   PSWDASIS    USEEXTOP
      OBJCLASS:  ACF2LID
      NEXTKEY:   *** NO NEXTKEY DEFINED ***
      URL:       LDAP://xxx.xxx.xxx.xx:389
      USERDNS:   acf2lid=%l,host=xxx,o=xx,c=xx
      LDSLABEL:
      XREF:      LID:                ATTRIBUTE:
                 NAME                name
                 PASSWORD            USERPASSWORD
```

The following example shows what you might see when issuing the SHOW LDS command with an LDAP record that was deactivated.

Note: LDS deactivates the LDAP node record for several reasons, an incorrect administrator password was supplied to the LDAP server or the limit of consecutive LDAP requests that have timed-out was exceeded. Analyze the previous messages for the specific type of error. Verify the LDS LDAP record field values for the URL, the ADMINDN, ADMPSWD (LDAP administrator password), or the APPLNAME (application name) is correct. Verify the LDAP server and the communication services are running. Issue the REFRESH or LDS(ACTIVE) operator command for the LDS LDAP record to reactivate this node.

```
acf
show lds

-- LIGHTWEIGHT DIRECTORY ACCESS PROTOCOL --

    CURRENT LDS STATUS:          ACTIVE
    CURRENT LDS JOURNAL STATUS:  ACTIVE
    LDS JOURNAL DATASET NAME:    CALDAP.LDSJRNL
    CURRENT LDS RECOVERY STATUS: ACTIVE
    LDS RECOVERY DATASET NAME:   CALDAP.LDSRCVR
    CURRENT SYSID:               TEST
    LDSRING:                     NONE
    OPTIONS:     DEBUG    RETRY(3)  SYSCLASS(A)  SYSDEST(LOCAL)    TIMEOUT(30)

    ACTIVE LDAP RECORD LIST:

    - LDAP.CPU2
      STATUS:    DISCONNECTED
      OPTIONS:   NOACTIVE    BROADCST    CHANGE    DEBUG       DELCHILD
                 DELETE      INSERT      JOURNAL   PSWDASIS    USEEXTOP
      OBJCLASS:  ACF2LID
      NEXTKEY:   *** NO NEXTKEY DEFINED ***
      URL:       LDAP://xxx.xxx.xxx.xx:389
      USERDNS:   acf2lid=%l,host=xxx,o=xx,c=xx
      LDSLABEL:  NONE
      XREF:      LID:                ATTRIBUTE:
                 NAME                name
                 PASSWORD            USERPASSWORD
```

The following is an example of the SHOW LDS command when LDS is not active.

```
acf
show lds

-- LIGHTWEIGHT DIRECTORY ACCESS PROTOCAL --

   CURRENT LDS STATUS:         INACTIVE
   CURRENT LDS JOURNAL STATUS: INACTIVE
   LDS JOURNAL DATASET NAME:   NONE
   CURRENT LDS RECOVERY STATUS:INACTIVE
   LDS RECOVERY DATASET NAME:   NONE

   CURRENT SYSID:               NONE
   LDSRING:                     NONE
   OPTIONS:     NODEBUG  RETRY(0)  SYSCLASS()   SYSDEST()         TIMEOUT(0)

   ACTIVE LDAP RECORD LIST:

   THERE ARE NO ACTIVE LDAP RECORDS IN THIS SYSTEM

acf
```

The following is an example of the SHOW LDS command when LDS is active and the LDAP node is connected.

```
acf
show lds

-- LIGHTWEIGHT DIRECTORY ACCESS PROTOCOL --

    CURRENT LDS STATUS:          ACTIVE
    CURRENT LDS JOURNAL STATUS:  ACTIVE
    LDS JOURNAL DATASET NAME:    CALDAP.LDSJRNL
    CURRENT LDS RECOVERY STATUS: ACTIVE
    LDS RECOVERY DATASET NAME:   CALDAP.LDSRCVR

    CURRENT SYSID:               TEST
    LDSRING:                     NONE
    OPTIONS:     DEBUG    RETRY(3)  SYSCLASS(A)  SYSDEST(LOCAL)    TIMEOUT(30)

    ACTIVE LDAP RECORD LIST:

    - LDAP.CPU2
      STATUS:    CONNECTED
      OPTIONS:   ACTIVE    BROADCST    CHANGE    DEBUG       DELCHILD
                 DELETE    INSERT      JOURNAL   PSWDASIS    USEEXTOP
      OBJCLASS:  ACF2LID
      NEXTKEY:   *** NO NEXTKEY DEFINED ***
      URL:       LDAP://xxx.xxx.xxx.xx:389
      USERDNS:   acf2lid=%l,host=xxx,o=xx,c=xx
      LDSLABEL:  NONE
      XREF:      LID:                ATTRIBUTE:
                 NAME                name
                 PASSWORD            USERPASSWORD
```

ISPF Panels

This section has examples of the following ISPF panels:

- Insert Control LDS LDAP Record
- Insert Control LDS XREFLDAP Record
- Insert Control LDS OPTIONS Record
- Change Control LDS OPTIONS Record
- Add a new logonid
- Change a logonid

Control LDS LDAP Record Panels

Use the following panels to insert a Control LDS LDAP record.

From the eTrust CA-ACF2 Security ISPF Option Selection Menu, select option 20 - LDS – Process LDAP Directory Services. The following panel displays:

```
--------------------eTrust CA-ACF2 LDAP DIRECTORY SERVICES--------------------

OPTION===>  1

1  INSERT       - INITIALLY DEFINE LDS RECORDS

2  CHANGE       - CHANGE EXISTING LDS RECORDS

3  LIST         - DISPLAY LDS RECORDS AND PARAMETERS

4  DELETE       - DELETE AN ENTIRE LDS RECORD

5  SHOW ALL     - SHOW ALL ACFFDR AND LDS OPTIONS IN EFFECT

6  FIELDS       - SHOW LDS RECORD FIELD NAMES

7  TARGETS      - SET CPF TARGET NODES, DEFAULTS IN USE
```

Select 1 and press <ENTER> to continue.

```
-------------------------- INSERT AN LDS RECORD -----------------------------
COMMAND===>

  INSERT

  CHANGE TYPE   ===> ADD        (ADD)

  SYSID         ===> CPU1       SYSTEM ID FOR LDS RECORD

  USING SYSID   ===>            OPTIONAL SYSTEM ID FOR COPY OF EXISTING VALUES

  USING RECID   ===>                   OPTIONAL PROTOTYPE RECORD NAME

  RECID         ===> LDAP       (LDAP, OPTIONS, XREFLDAP)

SPECIFICATION OF RECID WILL RESULT IN DISPLAY OF APPROPRIATE PANEL FOR OPTION
SPEC
```

Panel Fields

CHANGE TYPE
Specifies the action. This field default is ADD.

SYSID
Specifies the system identification.

RECID
Specifies the type of record to insert.

```
------------------------- INSERT AN LDS RECORD -----------------------------
COMMAND===>

COMMAND:    INSERT   (1 OF 5)

Record ID:  LDAP      Qualifier ===> cpu2

ADMINDN ===> acf2lid=admin

Press ENTER to continue or END to cancel
```

Panel Fields

Qualifier
Must be 1 to 8 characters long. A period is automatically inserted between LDAP and the qualifier as part of the record id.

ADMINDN
Indicates the LDAP administrator distinguished name used for binding to the LDAP server and administering the LDAP request.

The following string substitutions are allowed for this field:

Symbolic	String Substitution
%N	Substitutes the administrator's name (20 bytes) from the administrator's LID record
%L	Substitutes the administrator's logonid (8 bytes) from the administrator's LID record

Note: If there are embedded spaces or commas in the ADMINDN, enclose the entire string in quotes.

```
------------------------- INSERT AN LDS RECORD ----------------------------
COMMAND ===>

  MODE=CONTROL  TYPE=LDS  FUNCTION=ADD  SYSID=CPU1

  COMMAND:   INSERT    (2 OF 5)

  RECORD ID: LDAP.CPU2

        Enter either Application Name or Administrative Password:
        APPLNAME  ===>
        ADMPSWD   ===>


        NEXTKEY   ===>

        Enter Y or N:
        ACTIVE    ===>        BROADCST  ===>        CHANGE    ===>
        DEBUG     ===>        DELCHILD  ===>        DELETE    ===>
        INSERT    ===>        JOURNAL   ===>        PSWDASIS  ===>
        USEEXTOP  ===>

  Press ENTER to continue or END to go back to the previous panel.
```

Panel Fields

APPLNAME
Indicates the application name. The application specifies the record ID of the SSIGNON profile data record that contains the administrator's encryption key used to generate PassTicket.

ADMPSWD
Indicates the LDAP administrator password. **Note:** It is required to define the application name (APPLNAM) or the administrator password (ADMPSWD) when creating or modifying the LDS LDAP record. These fields are mutually exclusive. The LDAP administrator distinguished name and the PassTicket or LDAP administrator password is used for authentication and authorization the LDAP server.

NEXTKEY
Specifies the qualifier name of the Control LDS XREFLDAP record that contains more XREF field values to be mapped to this LDAP

ACTIVE
Indicates this LDAP server is active and data requests are communicated. The default is NOACTIVE.

BROADCST
Specifies to capture all logonid administrator commands that are indicated for this record regardless if the LID record is flagged for LDS propagation.

CHANGE
Indicates eTrust CA-ACF2 command logonid change process is propagated to the LDAP server. The default is NOCHANGE.

DEBUG
Bit option field indicates diagnostic messages are captured during communication to this LDAP server. The default is NODEBUG.

DELCHILD
Bit option field indicates to delete child objects when a delete command of the base object is issued to this node. The default is NODELCHILD.

DELETE
Indicates eTrust CA-ACF2 command logonid delete process is propagated to the LDAP server. The default is NODELETE.

INSERT
Indicates eTrust CA-ACF2 command logonid insert process is propagated to the LDAP server. The default is NOINSERT.

JOURNAL
Indicates LDS requests to the LDAP server will be written to the LDS Journal file when JOURNAL is active. The default is JOURNAL.

PSWDASIS
Indicates how the LID's password is propagated to the LDAP server. **Note:** The LID password field must be specified in the XREF field. PSWDASIS propagates the current LID's password as defined and NOPSWDASIS propagates in all uppercase letters. The default is PSWDASIS.

USEEXTOP
Indicates this LDAP server requires the use of an extended operation. The default is NOUSEEXTOP.

```
------------------------  INSERT AN LDS LDAP  RECORD ---------------------------
 COMMAND ===>


   COMMAND:       (3 OF 5)
   RECORD ID:

     DATE DEFAULT ===>                    Current Default is 'MMDDYYYY'
     BIT DEFAULT  ===>                    Current Default is 'CHARACTER/YN'
     DELIMITER    ===>
     LDSLABEL     ===>

     OBJECT CLASS ===>
              URL  ===>



          USERDNS ===>


  Press ENTER to continue or END to go back to the previous panel.
```

Panel Fields

DATE DEFAULT

Indicates the default format of date fields referenced in the Reference Field of this Control LDS LDAP record. The default is MMDDYY. MM represents a two-digit month, DD represents a two-digit day, YY represents a two-digit year, and YYYY represents the four-digit year. For example, if the current month is 04, the day is 28, and the year is 2001, the date formats are as follows:

- MMDDYYYY – 04282001
- DDMMYYYY – 28042001
- YYYYMMDD – 20010428
- MMDDYY1 – 04/28/01
- DDMMYY1 – 28/04/01
- YYMMDD1 – 01/04/28

BIT DEFAULT

Indicates the default field type and format of bit fields referenced in the cross-reference field (XREF) of this Control LDS LDAP record. The default is CHARACTER/YN.

DELIMIT(*character*)

Indicates a one byte character date delimiter used in conjunction with the DATEFMT field to format additional output date value. Only date formats MMDDYY, DDMMYY, YYMMDD, MMDDYYYY, DDMMYYYY, and YYYYMMDD can be specified with one of the following valid DELIMIT characters: dash, semicolon, forward slash, period, coma, and space. The following table lists examples of the formatted output date for April, 28, 2001 with a valid DELIMIT and DATEFMT specified:

DELIMIT	DATEFMT	Example
-	MMDDYY	04-28-01
:	DDMMYY	28:04:01
/	YYMMDD	01/04/28
.	MMDDYYYY	04.28.2001
,	DDMMYYYY	28,04,2001
‘ ‘	YYYYMMDD	2001 04 28

LDSLABEL

Specifies the label of the PERSONAL certificate that is connected to the RINGNAME of the eTrust CA-ACF2 KEYRING indicated on the LDS OPTIONS record. The label can be up to 32 characters in length. It can contain blanks and mixed-case characters.

OBJECT CLASS

Defines the attributes the LDAP directory might contain. The default is ACF2LID.

URL

Identifies the Uniform Resource Locator (URL) used to identify the LDAP server. This field can be up 1 to up to 255 characters in length. The syntax must be as follows:

```
ldap://host.domain:port
```

or

```
ldaps://host.domain:port
```

ldap/ldaps

Specifies a connection using the LDAP protocol. 'ldap' starts an unencrypted connection, where 'ldaps' starts an encrypted connection.

host

Specifies the name or IP address of the LDAP server host.

port

Specifies the port number of the LDAP server.

USERDNS

Indicates the user distinguished name syntax that refers to the entry on the LDAP server where the changes will be applied. The maximum length is 255-characters.

The following indicates the string substitutions allowed for this field:

Symbolic	String Substitution
%N	User's name (20 bytes) from the user LID record
%L	User's LID (8 bytes) from the user LID record

Note: If there are embedded spaces or commas in the field, enclose the entire string in quotes.

```
------------------------  INSERT AN LDS LDAP RECORD -----------------------------
COMMAND ===>

  MODE=CONTROL  TYPE=LDS   FUNCTION=ADD   SYSID=CPU1

  COMMAND:     INSERT    (4 OF 5)

  RECORD ID:   LDAP.CPU2

  LDAPINST FIELDS:  -  ZERO LDAPINST FIELDS ARE ENTERED SO FAR
                       (Enter only one LDAPINST field entry at a time)

          ===>


  Press ENTER to continue or END to go back to previous panel.
```

Panel Fields

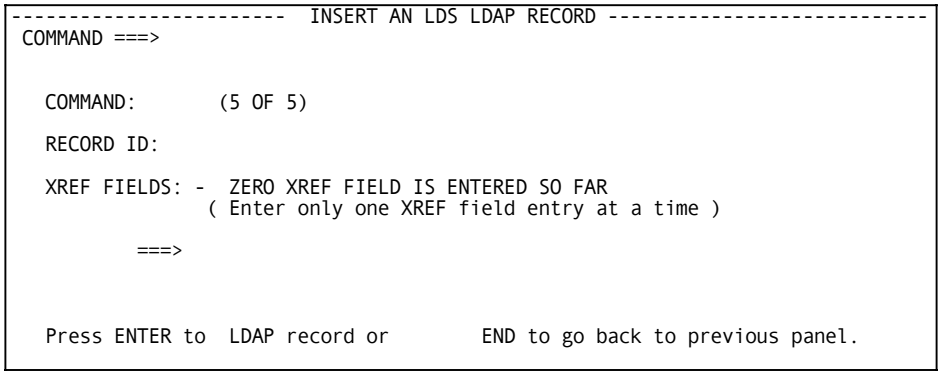
LDAPINST

Specifies installation specific information to be appended to LDS requests as hardcoded attribute name and attribute values. This field is optional. The syntax of the LDAPINST field is the following:

```
LDAPINST(AttributeName/AttributeValue/AttributeType,...)
```

Where:

- AttributeName, Value, and Type are to be delimited by a forward slash'/'.
- AttributeType is 'CHARACTER'.
- LDAPINST field length is from 3 to 255 characters in length.



Panel Fields

XREF FIELDS (*LIDfield1/LDAPattribute1Name/LDAPattribute1FieldType/LDAPattribute1FieldFormat/LDAPattribute1Fieldlength*)

Specifies the name of the eTrust CA-ACF2 logonid fields and the corresponding LDAP directory attribute fields to be synchronized to the LDAP directory for insert or change LID command processing. The LIDfield and LDAPattributeName are required parameters and must be delimited by a '/' (forward slash).

Insert Control LDS XREFLDAP Record Panels

Use the following panels to insert a Control LDS XREFLDAP record.

```
------------------eTrust CA-ACF2 LDAP DIRECTORY SERVICES----------------------
OPTION===>
1  INSERT       - INITIALLY DEFINE LDS RECORDS
2  CHANGE       - CHANGE EXISTING LDS RECORDS
3  LIST         - DISPLAY LDS RECORDS AND PARAMETERS
4  DELETE       - DELETE AN ENTIRE LDS RECORD
5  SHOW ALL     - SHOW ALL ACFFDR AND LDS OPTIONS IN EFFECT
6  FIELDS       - SHOW LDS RECORD FIELD NAMES
7  TARGETS      - SET CPF TARGET NODES
```

Select 1 and press <ENTER> to continue.

```
---------------------- INSERT AN LDS XREFLDAP RECORD --------------------------
COMMAND===>
  INSERT
  CHANGE TYPE ===> ADD       (ADD)
  SYSID       ===> CPU1      SYSTEM ID FOR LDS RECORD
  USING SYSID ===>           OPTIONAL SYSTEM ID FOR COPY OF EXISTING VALUES
  USING RECID ===>              OPTIONAL PROTOTYPE RECORD NAME
  RECID       ===> xrefldap  (LDAP, OPTIONS, XREFLDAP)
```

```
---------------------- INSERT AN LDS XREFLDAP RECORD --------------------------
COMMAND===>

   COMMAND:   INSERT    (1 OF 2)
   Record ID: XREFLDAP
     Qualifier  ===> cpu2
       NextKey  ===> cpu3

   Press ENTER to continue or END to cancel
```

Panel Fields

Qualifier
Must start with a period and must be one to seven characters long.

NextKey
Specifies the qualifier of the next Control LDS XREFLDAP record that contains more XREF field values to be mapped to the requesting LDAP server.

```
---------------------- INSERT AN LDS XREFLDAP RECORD --------------------------
COMMAND===>

   MODE=CONTROL   TYPE=GSO   FUNCTION=ADD   SYSID=CPU1

       COMMAND:   INSERT     (2 OF 2)

    Record ID:    XREFLDAP.CPU2

  XREF FIELDS:    - ZERO XREF FIELD IS ENTERED SO FAR
                  (  Enter only one XREF field entry at a time  )

          ===>    password/ldappassword

  Press ENTER to INSERT XREFLDAP record or END to go back to previous panel
```

Panel Fields

XREF FIELDS (*LIDfield1/LDAPattribute1Name/LDAPattribute1FieldType/LDAPattribute1FieldFormat/LDAPattribute1Fieldlength*)
Specifies the name of the eTrust CA-ACF2 logonid fields and the corresponding LDAP directory attribute fields to be mapped to the LDAP directory for insert or change LID command processing. The LIDfield and LDAPattributeName are required parameters and must be delimited by a '/' (forward slash).

Insert Control LDS OPTIONS Record Panels

Use the following panels to insert a Control LDS OPTIONS record.

```
----------------------eTrust CA-ACF2 LDAP DIRECTORY SERVICES--------------------

OPTION===>  1

1  INSERT       - INITIALLY DEFINE LDS RECORDS

2  CHANGE       - CHANGE EXISTING LDS RECORDS

3  LIST         - DISPLAY LDS RECORDS AND PARAMETERS

4  DELETE       - DELETE AN ENTIRE LDS RECORD

5  SHOW ALL     - SHOW ALL ACFFDR AND LDS OPTIONS IN EFFECT

6  FIELDS       - SHOW LDS RECORD FIELD NAMES

7  TARGETS      - SET CPF TARGET NODES
```

Select 1 and press <ENTER> to continue.

```
------------------------- INSERT AN LDS RECORD -----------------------------
COMMAND===>

  INSERT

  CHANGE TYPE   ===> ADD        (ADD)

  SYSID         ===> CPU1       SYSTEM ID FOR LDS RECORD

  USING SYSID   ===>            OPTIONAL SYSTEM ID FOR COPY OF EXISTING VALUES

  USING RECID   ===>                   OPTIONAL PROTOTYPE RECORD NAME

  RECID         ===> OPTIONS    (LDAP, OPTIONS, XREFLDAP)


SPECIFICATION OF RECID WILL RESULT IN DISPLAY OF APPROPRIATE PANEL FOR OPTION
SPECIFIED
```

Panel Fields

SYSID
Specifies the system identification.

RECID
Specifies the type of record to insert.

```
---------------------------- INSERT AN LDS RECORD -----------------------------
COMMAND ===>


  COMMAND:

  RECORD-ID:

  FIELDS   ENTER Y OR N:
           DEBUG    ===>     JOURNAL  ===>     RECOVERY ===>

           SPECIFY A DATA SET NAME:
           DATASET  ===>
           LDSRCVR  ===>

           SPECIFY A VALID VALUE:
           RETRY    ===>           (VALID VALUES ARE 1 THROUGH 10)
           SYSCLASS ===>           (VALID SYSOUT CLASS)
           SYSDEST  ===>           (VALID SYSOUT DESTINATION)
           TIMEOUT  ===>           (VALID VALUES ARE 1 THROUGH 999)
```

Panel Fields

DEBUG | NODEBUG
Specifies that diagnostic messaging is captured during processing to the LDAP server and directed to diagnostic output logs. This option cannot be refreshed and can only be activated at LDS startup. The default is NODEBUG.

JOURNAL | NOJOURNAL
Specifies whether the LDS Journal processing is started when LDS starts. The use of the LDS journal file is optional. You can choose to run without the journal file by setting the NOJOURNAL attribute on the Control LDS OPTIONS record. The specification here is a global setting. If you specify the JOURNAL option on the Control LDS LDAP record, you can control journaling for each server. The default is NOJOURNAL.

RECOVERY
Specifies that LDS recovery processing can be started when LDS starts.

DATASET
Specifies the name of the file used by the LDS journal process. It is a 44 byte character data set name. The LDS journal file is created and initialized prior to starting the LDS journal processing. You may use the supplied INITLDSJ job in the SAMPJCL library to create the LDS journal file.

LDSRCVR
Specifies the name of the file used by the LDS recovery process. It is a 44 byte character data set name. The LDS recovery file is created and initialized prior to starting the LDS recovery processing. You may use the supplied INITLDSR job in the SAMPJCL library to create the LDS journal file.

RETRY
Indicates the number of consecutive unsuccessful attempts to deliver an LDS command to an ACTIVE LDAP server before the LDAP node will be DEACTIVATED by eTrust CA-ACF2 Security. Valid values are 1 through 10. The default is 3.

SYSCLASS(A)
Specifies the JES SYSOUT class for diagnostic log output. SYSCLASS is a 1-byte character alphanumeric field. This option cannot be refreshed and can only be activated at LDS startup. The default value is A.

SYSDEST(LOCAL)
Specifies the JES SYSOUT destination for diagnostic log output. SYSDEST is a 1-8 byte character field consisting of alphanumeric or national characters. This option cannot be refreshed and can only be activated at LDS startup. The default value is LOCAL.

TIMEOUT(30)
Indicates the number of seconds that eTrust CA-ACF2 will wait to receive a response from the LDAP server before the LDAP request will be considered as timed out. This option cannot be refreshed and can only be activated at LDS startup. Valid values are 1 through 999. The default is 30 seconds.

```
---------------------------- INSERT AN LDS RECORD -----------------------------
 COMMAND ===>


   COMMAND:

   RECORD-ID:

   SPECIFY A VALID VALUE:
            LDSRING  ===>
```

LDSRING
Specifies the ringname of the KEYRING record to which all the CERTAUTH and PERSONAL certificate information to establish SSL connections are obtained. The userid associated with the KEYRING record must be eTrust CA-ACF2. The ringname can be up to 237 characters in length.

Change Control LDS OPTIONS Record Panels

Use the following panels to change a Control LDS OPTIONS record.

```
-------------------eTrust CA-ACF2 LDAP DIRECTORY SERVICES------------------

OPTION===>  2

1  INSERT       - INITIALLY DEFINE LDS RECORDS

2  CHANGE       - CHANGE EXISTING LDS RECORDS

3  LIST         - DISPLAY LDS RECORDS AND PARAMETERS

4  DELETE       - DELETE AN ENTIRE LDS RECORD

5  SHOW ALL     - SHOW ALL ACFFDR AND LDS OPTIONS IN EFFECT

6  FIELDS       - SHOW LDS RECORD FIELD NAMES

7  TARGETS      - SET CPF TARGET NODES
```

Select 2 and press <ENTER> to continue.

```
------------------------- CHANGE AN LDS RECORD -----------------------------
COMMAND===>

  CHANGE       ===> ADD

  CHANGE TYPE  ===>          (ADD, DEL, REP)

  SYSID        ===> CPU1     SYSTEM ID FOR LDS RECORD

  MASK SYSID   ===>          MASKED SYSTEM ID FOR LDS RECORD

  MASK RECID   ===>                RECID MASKED VALUE

  RECID        ===> OPTIONS  (LDAP, OPTIONS, XREFLDAP)

SPECIFICATION OF RECID WILL RESULT IN DISPLAY OF APPROPRIATE PANEL FOR OPTION
SPECIFIED
```

Panel Fields

SYSID
Specifies the system identification.

RECID—Specifies the type of record to insert.

```
---------------------------  CHANGE AN LDS RECORD -----------------------------
 COMMAND ===>


   COMMAND:

   RECORD-ID:

   FIELDS   ENTER Y OR N:
            DEBUG    ===>     JOURNAL  ===>     RECOVERY ===>

            SPECIFY A DATA SET NAME:
            DATASET  ===>
            LDSRCVR  ===>

            SPECIFY A VALID VALUE:
            RETRY    ===>           (VALID VALUES ARE 1 THROUGH 10)
            SYSCLASS ===>           (VALID SYSOUT CLASS)
            SYSDEST  ===>           (VALID SYSOUT DESTINATION)
            TIMEOUT  ===>           (VALID VALUES ARE 1 THROUGH 999)
```

Panel Fields

DEBUG|NODEBUG
Specifies that diagnostic messaging is captured during processing to the LDAP server and directed to diagnostic output logs. This option cannot be refreshed and can only be activated at LDS startup. The default is NODEBUG.

JOURNAL
Specifies whether the LDS journal option is active.

RECOVERY
Specifies that LDS recovery processing can be started when LDS starts.

DATASET
Specifies the name of the file used by the LDS journal process. It is a 44 byte character data set name. The LDS journal file is created and initialized prior to starting the LDS journal processing. You may use the supplied INITLDSJ job in the SAMPJCL library to create the LDS journal file.

LDSRCVR
Specifies the name of the file used by the LDS recovery process. It is a 44 byte character data set name. The LDS recovery file is created and initialized prior to starting the LDS recovery processing. You may use the supplied INITLDSR job in the SAMPJCL library to create the LDS journal file.

RETRY
Indicates the number of consecutive unsuccessful attempts to deliver an LDS command to an ACTIVE LDAP server before the LDAP node will be DEACTIVATED by eTrust CA-ACF2 Security. Valid values are 1 through 10. The default is 3.

SYSCLASS
Specifies the JES SYSOUT class for diagnostic log output. SYSCLASS is a 1-byte character alphanumeric field. This option cannot be refreshed and can only be activated at LDS startup. The default value is A.

SYSDEST
Specifies the JES SYSOUT destination for diagnostic log output. SYSDEST is a 1-8 byte character field consisting of alphanumeric or national characters. This option cannot be refreshed and can only be activated at LDS startup. The default is LOCAL.

TIMEOUT
Indicates the number of seconds that eTrust CA-ACF2 will wait to receive a response from the LDAP server before the LDAP request will be considered as timed out. This option cannot be refreshed and can only be activated at LDS startup. Valid values are 1 through 999. The default is 30 seconds.

```
---------------------------  CHANGE AN LDS RECORD ----------------------------
 COMMAND ===>


   COMMAND:

   RECORD-ID:


  SPECIFY A VALID VALUE:
            LDSRING  ===>
```

LDSRING
Specifies the ringname of the KEYRING record to which all the CERTAUTH and PERSONAL certificate information to establish SSL connections are obtained. The userid associated with the KEYRING record must be eTrust CA-ACF2. The ringname can be up to 237 characters in length.

Add New Logonid

Use the following panel to add LDAP privileges to a new logonid.

```
------------------------------ Add a New Logonid -----------------------------
 COMMAND ===>
 Privileges Section for USER01
   ACTIVE   ===>             DSNSCOPE ===>             EXPIRE   ===>
   LIDSCOPE ===>             NOSPOOL  ===>             PROGRAM  ===>
   SCPLIST  ===>             SYNCNODE ===>             SYNERR   ===>
   UIDSCOPE ===>
 Enter Y to allow the following privileges or N to disallow. N is the default
  ACCOUNT  ==>   AUDIT    ==>   AUTOALL  ==>   AUTODUMP ==>   AUTONOPW  ==>
  AUTOONLY ==>   BDT      ==>   CICS     ==>   CMD-PROP ==>   CONSULT   ==>
  DG84DIR  ==>   DIALBYP  ==>   DUMPAUTH ==>   GRP-OPT  ==>   GRPLOGON  ==>
  IDMS     ==>   IMS      ==>   JOB      ==>   JOBFROM  ==>   LDS       ==>
  LDEV     ==>   LEADER   ==>   LOGSHIFT ==>   MAINT    ==>   MUSASS    ==>
  NO-INH   ==>   NO-OMVS  ==>   NO-SMC   ==>   NO-STORE ==>   NOMAXVIO  ==>
  NON-CNCL ==>   PPGM     ==>   PRIV-CTL ==>   READALL  ==>   REFRESH   ==>
  RESTRICT ==>   RSRCVLD  ==>   RULEVLD  ==>   SECURITY ==>   SRF       ==>
  STC      ==>   SUBAUTH  ==>   TAPE-BLP ==>   TAPE-LBL ==>   TDISKVLD  ==>
  TSO      ==>   VLDVMACT ==>   VM       ==>   VMD4AUTH ==>   VMD4RSET  ==>
  VMD4TARG ==>   VMSAF    ==>   VMSFS    ==>   VMXA     ==>   VSESRF    ==>
 Press ENTER to display next panel or END to redisplay previous panel.
  F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
  F7=UP        F8=DOWN      F9=SWAP      F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

LDS

Specifies whether Logonid administrative changes for this user will be propagated to the active LDAP servers in the network. Enter Y (yes) to activate.

Change Logonid

Use the following panel to change the LDAP privileges for a logonid.

```
------------------------------ Change A Logonid ------------------------------
 COMMAND ===>

 Privileges section for USER01
   ACTIVE   ===>             DSNSCOPE ===>             EXPIRE   ===>
   LIDSCOPE ===>             NOSPOOL  ===>             PROGRAM  ===>
   SCPLIST  ===>             SYNCNODE ===>             SYNERR   ===>
   UIDSCOPE ===>
 Enter Y or N to change privilege or blank to leave privilege unaltered.
  ACCOUNT  ==>   AUDIT    ==>   AUTOALL  ==>   AUTODUMP ==>   AUTONOPW  ==>
  AUTOONLY ==>   BDT      ==>   CICS     ==>   CMD-PROP ==>   CONSULT   ==>
  DG84DIR  ==>   DIALBYP  ==>   DUMPAUTH ==>   GRP-OPT  ==>   GRPLOGON  ==>
  IDMS     ==>   IMS      ==>   JOB      ==>   JOBFROM  ==>   LDS       ==>
  LDEV     ==>   LEADER   ==>   LOGSHIFT ==>   MAINT    ==>   MUSASS    ==>
  NO-INH   ==>   NO-OMVS  ==>   NO-SMC   ==>   NO-STORE ==>   NOMAXVIO  ==>
  NON-CNCL ==>   PPGM     ==>   PRIV-CTL ==>   READALL  ==>   REFRESH   ==>
  RESTRICT ==>   RSRCVLD  ==>   RULEVLD  ==>   SECURITY ==>   SRF       ==>
  STC      ==>   SUBAUTH  ==>   TAPE-BLP ==>   TAPE-LBL ==>   TDISKVLD  ==>
  TSO      ==>   VLDVMACT ==>   VM       ==>   VMD4AUTH ==>   VMD4RSET  ==>
  VMD4TARG ==>   VMSAF    ==>   VMSFS    ==>   VMXA     ==>   VSESRF    ==>

 Press ENTER to display next panel or END to redisplay previous panel.
  F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
  F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

LDS

Specifies whether Logonid administrative changes for this user will be propagated to the active LDAP servers in the network. Enter Y (yes) to activate or N (no) to deactivate.

Chapter

25

Digital Certificate Support

An X.509 Digital Certificate provides a secure method for identifying a user, typically through a Web-based application. As an alternative to requesting userid and password information, a z/OS Web server can authenticate users based upon their digital certificates. Digital certificates provide a means of authentication through the use of public-key cryptography and a trusted third party, known as a Certification Authority. A digital certificate is generated by the Certification Authority and is identified uniquely by its serial number and by the associated distinguished name of the Certification Authority ("issuer's distinguished name").

If MVS resources are accessed, the certificate is presented to eTrust CA-ACF2. Using the certificate serial number and the issuer's distinguished name, eTrust CA-ACF2 associates the certificate with an MVS userid. An MVS security environment is then created for that user. By using authenticated certificates, passwords are not sent through the network.

eTrust CA-ACF2 provides complete functionality to generate, install, and maintain digital certificates, key rings, and digital certificate mappings, including the following:

- Generate a digital certificate and a public/private key pair
- Create a PKCS #12 certificate package
- Create a PKCS #10 certificate request
- Export a digital certificate or certificate package and private key from eTrust CA-ACF2 to a z/OS dataset
- Display a certificate that is in a z/OS dataset and determine if it is associated with an eTrust CA-ACF2 user
- Display a certificate registered with eTrust CA-ACF2
- Automatically register a digital certificate with eTrust CA-ACF2
- Associate an eTrust CA-ACF2 user with a digital certificate
- Change, display, and delete information about a digital certificate for an eTrust CA-ACF2 user
- Create, change, display, and delete a key ring
- Add and remove a certificate from a key ring

- Assign an eTrust CA-ACF2 user to a group of certificates via user ID mapping
- Assign an eTrust CA-ACF2 user to a group of certificates based on system ID, application ID, or application-defined variables
- Change, delete, and display an eTrust CA-ACF2 user ID mapping

The following Table summarizes all of the eTrust CA-ACF2 commands that can be issued to generate, install and maintain digital certificates, key rings, and digital certificate mappings.

eTrust CA-ACF2 Commands

Command	Function	ACF Setting/ Component	Syntax
CHKCERT	eTrust CA-ACF2 displays information about an X.509 certificate in a CERTDATA profile record or a z/OS data set (including whether it is registered with eTrust CA-ACF2).	ACF COMMON SUBCOMMAND	CHKcert {*logonid* Label(*label*)| logonid.*suffix*|Dsname(*data-set-name*)} [Password(*password*)] [Nolist] [Dump]
CONNECT	eTrust CA-ACF2 associates a certificate with a key ring.	ACF COMMON SUBCOMMAND	CONnect Certdata(*userid1.suffix*) Keyring(*userid2.suffix*) [Ringname(*ringname*)] [Label(*label*)] [Usage(PERSONAL|CERTAUTH|SITE)] [DEFAULT]
EXPORT	eTrust CA-ACF2 exports an X.509 digital certificate from the eTrust CA-ACF2 database and puts it into a z/OS data set.	ACF COMMON SUBCOMMAND	Export {*logonid*|*logonid.suffix*} Dsname(*data-set-name*) [Label(*label*)] [Format(CERTDER|CERTB64| PKCS12DER|PKCS12B64| PKCS7DER|PKCS7B64)] [Password(*password*)]

Command	Function	ACF Setting/ Component	Syntax
GENCERT	eTrust CA-ACF2 generates a digital certificate and inserts a CERTDATA profile record into the eTrust CA-ACF2 infostorage database.	ACF COMMON SUBCOMMAND	GENCErt {*logonid*|*logonid.suffix*| CERTAUTH|CERTAUTH.*suffix*| SITECERT|SITECERT.*suffix* } [Label(*label*)] [Dsname(*data-set-name*)] [SUbjsdn([CN=*common name*] [T=*title*] [OU=*organizational-unit-name*] [O=*organization-name*] [L=*locality*] [S=*state-or-province*| SP=*state-or-province*| ST=*state-or-province*] [C=*country*])] [SIZe({*key-size*|1024})] [ICSF|PCICC] [ACtive({*date-or-date-time*| *current-date-000000*| *current-date-time*})] [Expire({*date-or-date-time*| *current-date-000000*| *current-date-time*})] [SIGnwith({CERTAUTH Label(*label-name*)| SITECERT Label(*label-name*)| CERTAUTH.*suffix*|SITECERT.*suffix*)| Label(*label-name*)}] [Keyusage([HANDSHAKE] [DATAENCRYPT][DOCSIGN] [CERTSIGN])] ALtname([IP=*numeric-ip-address*] [DOMAIN=*internet-domain-name*] [EMAIL=*email-address*] [URI=*universal-resource-identifier*])
GENREQ	eTrust CA-ACF2 generates a certificate request (PKCS #10) to be sent to a Certification Authority.	ACF COMMON SUBCOMMAND	GENReq {*logonid*|*logonid.suffix*} Dsname(*data-set-name*) [Label(*label*)]
REKEY	eTrust CA-ACF2 generates a new certificate with a new key pair using the contents of an existing certificate.	ACF COMMON SUBCOMMAND	REKey{logonid|logonid.suffix|CERTAUTH| CERTAUTH.suffix|SITECERT|SITECERT.suffix} [Label(existing-certificate-label)] [WITHLbl(new-certificate-label)] [WITHSfx(new-certificate-suffix)] [SIZe({key-size|1024})] [ICSF|PCICC] [ACtive({date-or-date-time| current-date-000000| current-date-time})] [Expire({date-or-date-time| current-date-000000| current-date-time})]

Command	Function	ACF Setting/ Component	Syntax
REMOVE	eTrust CA-ACF2 disassociates a certificate from a key ring.	ACF COMMON SUBCOMMAND	REMove Certdata(*userid1.suffix*) Keyring(*userid2.suffix*) [Ringname(*ringname*)] [Label(*label*)]
ROLLOVER	eTrust CA-ACF2 rolls over a certificate by removing the old private key, reconnecting the new certificate to the old key rings and updates the serial number base.	ACF COMMON SUBCOMMAND	ROllover{logonid|logonid.suffix|CERTAUTH| CERTAUTH.suffix|SITECERT| SITECERT.suffix} [Label(old-certificate-label)] [NEWLabel(new-certificate-Label)] [NEWSufx(new-certificate-suffix)] [Force]
INSERT	eTrust CA-ACF2 reads an X.509 digital certificate from a z/OS data set and inserts it, along with data from the command input line, into a CERTDATA profile record, which associates a user with a certificate.	PROFILE USER RECORD (CERTDATA)	Insert {*logonid*|*logonid.suffix*| CERTAUTH|CERTAUTH.*suffix*|SITECERT| SITECERT.*suffix* } [Active(*date*)] DSN(*data-set-name*) [Expire(*date*)] [Label(*label*)] [Password(*password*)] [HITRUST|TRUST|NOTRUST] [ICSF]
CHANGE	eTrust CA-ACF2 accepts data from the command input line and, accordingly, changes the CERTDATA profile record(s), which associates a user(s) with a certificate(s).	PROFILE USER RECORD (CERTDATA)	CHAnge {*logonid*|*logonid.suffix*| CERTAUTH|CERTAUTH.*suffix*|SITECERT| SITECERT.*suffix* } [Active(*date*)] [Expire(*date*)] [NEWLABEL(*label*)] [HITRUST|TRUST|NOTRUST] CHAnge *userid* ISSUERDN(*dn*) SERIAL#(*serial-number*) [Active(*date*)] [Expire(*date*)] [NEWLABEL(*label*)] [HITRUST|TRUST|NOTRUST]
DELETE	eTrust CA-ACF2 deletes the CERTDATA profile record(s), which associates a user(s) with a certificate(s).	PROFILE USER RECORD (CERTDATA)	DELete {*logonid* LABEL(*label*)| *logonid.suffix*| CERTAUTH LABEL(*label*)|CERTAUTH.*suffix*| SITECERT LABEL(*label*)|SITECERT.*suffix*} DELete *userid* ISSUERDN(*dn*) SERIAL#(*serial-number*)

Command	Function	ACF Setting/ Component	Syntax
LIST	eTrust CA-ACF2 displays the CERTDATA profile record(s), which associates a user(s) with a certificate(s).	PROFILE USER RECORD (CERTDATA)	List {*logonid* LABEL(*label*)|*logonid.suffix*| CERTAUTH LABEL(*label*)|CERTAUTH.*suffix*| SITECERT LABEL(*label*)|SITECERT.*suffix*} List *userid* ISSUERDN(*dn*) SERIAL#(*serial-number*)
INSERT	eTrust CA-ACF2 inserts a KEYRING profile record, which associates one or more certificates with a single user (logonid).	PROFILE USER RECORD (KEYRING)	Insert {*recid*|*recid.suffix* } [Default(*userid.suffix*)] Ringname(*ringname*)
CHANGE	eTrust CA-ACF2 accepts data from the command input line and, accordingly, changes the KEYRING profile record, which associates one or more certificates with a single user (logonid).	PROFILE USER RECORD (KEYRING)	CHAnge {*recid*|*recid.suffix* } [Default(*userid.suffix*)] [NEWNAME(*ringname*)]
DELETE	eTrust CA-ACF2 deletes the KEYRING profile record, which associates one or more certificates with a single user (logonid).	PROFILE USER RECORD (KEYRING)	DELete {*recid*|*recid.suffix* }
LIST	eTrust CA-ACF2 displays the KEYRING profile record, which associates one or more certificates with a single user (logonid).	PROFILE USER RECORD (KEYRING)	List {*recid* |*recid.suffix* }

Command	Function	ACF Setting/ Component	Syntax
INSERT	eTrust CA-ACF2 inserts a CERTMAP GSO record, which defines the IDN (issuer's distinguished name) or SDN (subject's distinguished name) filters used to assign a specific logonid to a group of certificates.	CONTROL GSO RECORD (CERTMAP)	Insert CERTMAP.*recid* [SDNFILTR(*subject's-dist-name-filter*)] [IDNFILTR(*issuer's-dist-name-filter*)] [DSN(*data-set-name*)] [CRITERIA(*criteria-name-template*)] [LABEL(*label*)] [TRUST|NOTRUST] [USERID(*userid-to-map-to*) [MULTIID|NOMULTIID]
CHANGE	eTrust CA-ACF2 accepts data from the command input line and, accordingly, changes the CERTMAP GSO record, which defines the IDN (issuer's distinguished name) or SDN (subject's distinguished name) filters used to assign a specific logonid to a group of certificates.	CONTROL GSO RECORD (CERTMAP)	CHAnge CERTMAP.*recid* [SDNFILTR(*subject's-dist-name-filter*)] [IDNFILTR(*issuer's-dist-name-filter*)] [DSN(*data-set-name*)] [CRITERIA(*criteria-name-template*)] [LABEL(*label*)] [TRUST|NOTRUST] [USERID(*userid-to-map-to*)] [MULTIID|NOMULTIID]

Command	Function	ACF Setting/ Component	Syntax
DELETE	eTrust CA-ACF2 deletes the CERTMAP GSO record, which defines the IDN (issuer's distinguished name) or SDN (subject's distinguished name) filters used to assign a specific logonid to a group of certificates.	CONTROL GSO RECORD (CERTMAP)	`DELete CERTMAP.`*`recid`*
LIST	eTrust CA-ACF2 displays the CERTMAP GSO record, which defines the IDN (issuer's distinguished name) or SDN (subject's distinguished name) filters used to assign a specific logonid to a group of certificates.	CONTROL GSO RECORD (CERTMAP)	`List CERTMAP.`*`recid`*
SHOW	eTrust CA-ACF2 displays information contained in CERTMAP records as laid out in the internal CERTMAP table.	ACF COMMON SUBCOMMAND	`SHow CERTMAP`

Command	Function	ACF Setting/ Component	Syntax

INSERT	eTrust CA-ACF2 inserts a CRITMAP GSO record, which is used in conjunction with the CRITERIA parameter of the CERTMAP GSO record, to assign a specific logonid to a group of certificates based on the system ID, application ID, or application-defined variables specified in the CRITMAP GSO record.	CONTROL GSO RECORD (CRITMAP)	`Insert CRITMAP.recid`
			`[APPLID(application-name)]`
			`[SYSTEMID(sysid)]`
			`[APPLVAR(site-variable-list)]`
			`USERID(userid-to-map-to)`

CHANGE	eTrust CA-ACF2 accepts data from the command input line and, accordingly, changes the CRITMAP GSO record, which is used in conjunction with the CRITERIA parameter of the CERTMAP GSO record, to assign a specific logonid to a group of certificates based on the system ID, application ID, or application-defined variables specified in the CRITMAP GSO record.	CONTROL GSO RECORD (CRITMAP)	`CHAnge CRITMAP.recid`
			`[APPLID(application-name)]`
			`[SYSTEMID(sysid)]`
			`[APPLVAR(site-variable-list)]`
			`USERID(userid-to-map-to)`

Command	Function	ACF Setting/ Component	Syntax
DELETE	eTrust CA-ACF2 deletes the CRITMAP GSO record, which is used in conjunction with the CRITERIA parameter of the CERTMAP GSO record, to assign a specific logonid to a group of certificates based on the system ID, application ID, or application-defined variables specified in the CRITMAP GSO record.	CONTROL GSO RECORD (CRITMAP)	`DELete CRITMAP.`*`recid`*
LIST	eTrust CA-ACF2 displays the CRITMAP GSO record, which is used in conjunction with the CRITERIA parameter of the CERTMAP GSO record, to assign a specific logonid to a group of certificates based on the system ID, application ID, or application-defined variables specified in the CRITMAP GSO record.	CONTROL GSO RECORD (CRITMAP)	`List CRITMAP.`*`recid`*
SHOW	eTrust CA-ACF2 displays information contained in CRITMAP records as laid out in the internal CRITMAP table.	ACF COMMON SUBCOMMAND	`SHow CRITMAP`

Processing Digital Certificates with ACF Subcommands

After you enter the ACF command, you can use the subcommands listed below to create, display, and export digital certificates, create certificate requests, and associate certificates with or disassociate certificates from key rings. These subcommands are common to all settings. A detailed description of each subcommand follows:

- CHKCERT
- EXPORT
- GENCERT
- GENREQ
- CONNECT
- REKEY
- REMOVE
- ROLLOVER

CHKCERT Subcommand

The CHKCERT subcommand is used to display information about an X.509 certificate in a CERTDATA Profile record or a z/OS data set, including whether it is registered with eTrust CA-ACF2. The certificate can be in DER-encoded or base-64-encoded form (PEM). It can also be a PKCS #12 certificate, which includes the private key associated with the certificate.

The CHKCERT subcommand can be issued in any mode of the ACF command. It has the following syntax:

```
CHKcert {logonid Label(label) |logonid.suffix | Dsname(data-set-name)}
[Password(password)]
[Nolist]
[Dump]
```

Parameter Descriptions

***logonid* Label(*label*)**

Indicates the eTrust CA-ACF2 user whose label is used to specify which CERTDATA record containing the certificate is displayed. **Note:** For every one apostrophe desired in the Label value, two consecutive apostrophes must be specified. For example, the Label value, Frank's Certificate, should be specified as, Frank''s Certificate. If a single apostrophe is specified in the Label value, the value will be considered invalid.

logonid.suffix
Indicates the eTrust CA-ACF2 record id of the CERTDATA record containing the certificate that is displayed.

Dsname(*data-set-name*)
Indicates the name of the data set containing the certificate to be checked. If the data set name is enclosed in single quotes, it is considered fully qualified and is used as specified. Otherwise, the user's prefix, as specified by the TSO PROFILE PREFIX command (or defaulted from the DFT-PFX field of the logonid record) is added to the front of the data set name.

Password(*password*)
The password associated with a PKCS #12 certificate. This must be the same password that was specified when the certificate was exported.

Nolist
Indicates that the eTrust CA-ACF2 CERTDATA profile record should not be listed, even if the certificate is registered with eTrust CA-ACF2.

Dump
Indicates to display a hex dump of the certificate prior to listing the contents of the certificate.

Examples

The following displays information about the certificate in the FRANK01.MYCERT data set.

```
CHKCERT DSN('frank01.mycert')
```

Since the certificate is registered with eTrust CA-ACF2 and the User Profile Directory has been rebuilt, the CERTDATA profile record is listed.

```
Data set name:
    FRANK01.MYCERT

Certificate ID:
    4gjDxdnjweTjyNeBmZKiQMPB

Serial number:
    02
Issuer's distinguished name:
    CN=Blue Lock Company Certificate Authority
    OU=Auditing Department
    O=Blue Lock Company
    C=US
Subject's distinguished name:
    CN=Frank Schwinger
    OU=Sales Department
    O=Blue Lock Company
    C=US

Not valid before:
    2002/02/02  00:00:00 UTC
Not valid after:
    2003/12/31  00:00:00 UTC
Private Key Type:
    ICSF(NON-ICSF)
Private key bit size:
    1024

This certificate is registered with eTrust CA-ACF2

The CERTDATA record key is FRANK01.MYCERT

CERTDATA / FRANK01.MYCERT LAST CHANGED BY CUNKE01 ON 02/02/02-11:23
                   ISSUERDN(CN=Blue Lock Company Certificate
                   Authority.OU=Auditing Department.O=Blue Lock Company.C=US)
                   KEYSIZE(1,024) LABEL(Frank01 Cert) SERIAL#(02) SUBJDN(CN=Frank
                   Schwinger.OU=Sales Department.O=Blue Lock Company.C=US) TRUST

Certificate is not connected to any keyrings
```

The following are examples of the CHKCERT command. It can be used to display a certificate contained in a data set or a certificate contained in an eTrust CA-ACF2 User Profile CERTDATA record:

```
CHKCERT DSN('frank01.mycert') NOLIST
CHKCERT frank01.cert DUMP
CHKCERT certauth label(Audit CA)
```

While only a few common attributes usually appear in distinguished names, there are many possible attributes. Attributes that are not recognized by eTrust CA-ACF2 are displayed in the notation recommended by Internet RFC 1779, *A string representation of distinguished names*. For example, the last attribute of the subject's distinguished name is listed as OID.2.5.4.20==#3535352031323132. It is possible to look up the Object ID (OID) at an OID locator site, such as http://www.alvestrand.no/domen/objectid/top.html and determine that the OID represents a telephone number. The value of the field is displayed in hexadecimal and it is possible to see that it is the ASCII representation of the telephone number: 555 1212.

EXPORT Subcommand

The EXPORT subcommand is used to export an X.509 digital certificate from the eTrust CA-ACF2 database and put it into a z/OS data set. The data set can be used to insert the certificate in another system, or can be downloaded to a personal computer and installed in a web browser. If you send the exported certificate to others that receive messages from you signed with your private key, they can use the public key in the exported certificate to validate those messages, but they cannot forge messages from you because they do not have your private key. Your private key can be exported using the PKCS12DER or PKCS12B64 format options. Using these options will generate a PKCS #12 certificate package containing the user certificate, its private key, and all certificate-authority certificates necessary to complete the chain of certificates from user certificate to root certificate-authority certificate.

The EXPORT command can be issued in any mode of the ACF command. It has the following syntax:

```
Export  {logonid|logonid.suffix}
        Dsname(data-set-name)
        [Label(label)]
        [Format(CERTDER|CERTB64|PKCS12DER|PKCS12B64|PKCS7DER|PKCS7B64)]
        [Password(password)]
```

Parameter Descriptions

logonid*|*logonid.suffix

The record key of the certificate to be exported. This can be a one to eight-character logonid or a logonid, a dot, and a one to-eight character suffix. If LABEL is specified in addition to suffix, then both suffix and the label must refer to the same CERTDATA record.

Dsname(*data-set-name*)

The name of the data set into which the certificate is exported. The data set must not already exist. If the data set name is enclosed in single quotes, it is considered fully qualified and is used as specified. Otherwise, the user's prefix, as specified by the TSO PROFILE PREFIX command (or defaulted from the DFT-PFX field of the logonid record) is added to the front of the data set name.

Label(*label*)

The label of the certificate to be exported. Logonid must also be specified to indicate which logonid the label is associated with. **Note:** For every one apostrophe desired in the Label value, two consecutive apostrophes must be specified. For example, the Label value, Frank's Certificate, should be specified as, Frank"s Certificate. If a single apostrophe is specified in the Label value, the value will be considered invalid.

Password(*password*)

A password used to encrypt the private key and certificates in a PKCS #12 package. This password does not conform to normal eTrust CA-ACF2 password syntax and can be mixed case and up to 255 bytes in length. If password is specified, a PKCS #12 package is generated with the user certificate, private key and CA certificates. If format is not specified, format will default to PKCS12B64. Note that eTrust CA-ACF2 only supports PKCS #12 certificates that adhere to the PKCS #12 v1.0 standard published by RSA. These certificates are defined with a 3 in the version number of the PKCS #12 certificate package.

Format(CERTDER)

Indicates that the exported certificate should be encoded using the X.509 Distinguished Encoding Rules (DER). This is the standard form of an X.509 certificate. It is a binary file, so if it is being transferred using FTP, BINARY or IMAGE mode must be used.

Format(CERTB64)

Indicates that the exported certificate should be encoded using base-64 encoding. This encoding is applied to the standard X.509 certificate to make it possible to ship the certificate through systems, such as E-mail systems, that cannot handle binary files. This is a text file, so if it is being transferred using FTP, ASCII or TEXT mode must be used. Format(CERTB64) is the default if no format is specified.

Format(PKCS12DER)
Specifies a DER-encoded PKCS#12 certificate package. If this option is selected, a PASSWORD must also be supplied. Format PKCS12DER must be used if you need to import a PKCS#12 certificate package on Windows, since Windows cannot directly import a PKCS12B64 format PKCS#12 package.

Format(PKCS12B64)
Specifies a DER-encoded then base-64 encoded PKCS #12 certificate package. If this option is selected, a PASSWORD must also be supplied. Format (PKCS12B64) is the default if a password has been specified but no format is specified.

Format(PKCS7DER)
Specifies a DER encoded PKCS 7 certificate package. This will export a certificate and it's CA chain. If a certificate in the chain cannot be found under the CERTAUTH ID or the certificate is expired, an informational message will be issued and an incomplete PKCS 7 package will be created. eTrust CA-ACF2 will still be able to process the incomplete package but it may not be useful to OEM products.

Format(PKCS7B64)
Specifies a base-64 encoded PKCS 7 certificate package. This will export a certificate and it's CA chain. If a certificate in the chain cannot be found under the CERTAUTH ID or the certificate is expired, an informational message will be issued and an incomplete PKCS 7 package will be created. eTrust CA-ACF2 will still be able to process the incomplete package but it may not be useful to OEM products.

Examples

The following exports the certificate with a record key of FRANK01.CERT into a data set named FRANK01.MYCERT, using base-64 encoding.

```
EXPORT FRANK01.CERT DSNAME(MYCERT)
```

The following exports the certificate belonging to user FRANK01, with a label of "Frank01 Cert," into a data set named PUBLIC.TESTCERT, using DER encoding.

```
EXPORT FRANK01 LABEL(Frank01 Cert) DSNAME('PUBLIC.TESTCERT') FORMAT(CERTDER)
```

The following exports the certificate with a record key of CQLRG.TESTCERT into a data set named CQLRG.TESTCERT.CER, using base-64 encoding.

```
EXPORT CQLRG.TESTCERT DSNAME(TESTCERT.CER) FORMAT(CERTB64)
```

The following exports the certificate and private key with a logonid of FRANK01 and a Label of "Frank01 Cert" into a data set named FRANK01.CERT.P12, using base-64 encoding. This PKCS #12 package will also contain the CA certificates that are in the signing chain.

```
EXPORT FRANK01 DSNAME('FRANK01.CERT.P12') LABEL(Frank01 Cert)
PASSWORD(This is my secret and you don't know it)
```

GENCERT Subcommand

The GENCERT subcommand is used to generate a digital certificate and insert a CERTDATA profile record into the eTrust CA-ACF2 infostorage database.

The GENCERT subcommand can be issued in any mode of the ACF command. It has the following syntax:

```
GENCert { logonid | logonid.suffix | CERTAUTH | CERTAUTH.suffix | SITECERT |

          SITECERT.suffix }
        [Label(label)]
        [Dsname(data-set-name)]
        [SUbjsdn([CN=common-name]
                 [T=title]
                 [OU=organizational-unit-name]
                 [O=organization-name]
                 [L=locality]
                 [{S=state-or-province |
                  SP=state-or-province |
                  ST=state-or-province}]
                 [C=country])]
        [SIZe({key-size|1024})]
        [ICSF]
        [PCICC]
        [ACtive({date-or-date-time|current-date-000000|
                 current-date-time})]
        [Expire({date-or-date-time|current-date-000000|
                 current-date-time})]
        [SIGnwith({ CERTAUTH Label(label-name) | CERTAUTH.suffix |
        SITECERT Label(label-name) | SITECERT.suffix) | Label(label-name)})]
        [Keyusage([HANDSHAKE][DATAENCRYPT]
                 [DOCSIGN][CERTSIGN])]
        [ALtname([IP=numeric-ip-address]
                 [DOMAIN=internet-domain-name]
                 [EMAIL=email-address]
                 [URI=universal-resource-identifier])]
```

Parameter Descriptions

***logonid*|CERTAUTH|SITECERT**

The logonid specifies the user associated with the certificate. It may be a one to eight-character logonid, or a logonid, a dot, and a one to eight-character suffix. If label is specified, logonid, rather than logonid.suffix, must be specified, and indicates the logonid with which the label is associated.

Using CERTAUTH in place of a logonid designates the certificate as a Certification Authority certificate.

Using SITECERT in place of a logonid designates the certificate as a site certificate.

Label(*label*)

Specifies a 1 to 32-character label that the new certificate will have. The label can contain blanks and mixed-case characters. If a label is not specified, the label field defaults to the upper-case version of the logonid or logonid.suffix that was specified. **Note:** For every one apostrophe desired in the Label value, two consecutive apostrophes must be specified. For example, the Label value, Frank's Certificate, should be specified as, Frank"s Certificate. If a single apostrophe is specified in the Label value, the value will be considered invalid.

Dsname(*data-set-name*)

Specifies the data set that contains a PKCS #10 certificate request, as generated by the GENREQ subcommand of the ACF command, or by other certificate management software. If the data set name is enclosed in single quotes, it is considered fully qualified and is used as specified. Otherwise, the user's prefix, as specified by the TSO PROFILE PREFIX command (or defaulted from the DFT-PFX field of the logonid record) is added to the front of the data set name.

DSNAME is optional. If specified, GENCERT will not generate a public/private key pair. If specified, SIGNWITH must also be specified because the PKCS #10 request does not contain a private key. If SUBJSDN is also specified, the subject's distinguished name is generated from the SUBJSDN that was entered on the command.

If DSNAME is specified and extensions are present within the PKCS #10 request, and they are not overridden by the other GENCERT keywords, they are copied to the new certificate.

DSNAME is mutually exclusive with the SIZE keyword.

SUbjsdn(*...attributes...*)

Specifies the subject's distinguished name. The attributes can consist of the following fields. Except as otherwise noted, valid characters for the values of the attributes are A through Z, a through z, 0 through 9, space, ',(,),+,comma,-,.,/,:,=,and ?. Values containing spaces must be enclosed in single quotes. Any apostrophes should be doubled. For example, a common name of John T. O'Reiley would be specified as CN='John T. O''Reiley'. Also, unless otherwise specified, each attribute can be specified only once.

Note the following:

- A space is the only valid delimiter between specified attributes. The maximum length for this parameter for a self-signed certificate is 229.
- The maximum length for this parameter for a non self-signed certificate is 255.
- The maximum length for each attribute of this parameter is 64. Multiple blanks are not removed and are included in the lengths.
- If the DSNAME keyword is also present, the subject's distinguished name from the SUBJSDN keyword is used instead of the subject's distinguished name from the PKCS #10 certificate request. If neither DSNAME nor SUBJSDN is specified, the subject's distinguished name is generated with CN='ACF2 USER:*logonid*'.
  - CN=*common-name*—Specifies the subject's regular name. For example, Sam Smith would be specified as CN='Sam Smith'. An '*' wilcard character may be used as the leftmost byte of the CN attribute, as in CN='*.example.com'.
  - T=*title*—Specifies the person's job title. For example, T='Software Developer'.
  - OU=*organizational-unit-name*—Specifies the department or group. This can be specified multiple times to indicate a hierarchy. For example, OU=Accounting,OU='Accounts Payable'.
  - O=*organization-name*—Specifies the name of the company. For example, O='Blue Lock Company'.
  - L=*locality*—Specifies the city. For example, L='Tom"s River'.
  - S=*state-or-province*, SP=*state-or-province*, ST=*state-or-province*—Specifies the state or province. All three keywords mean the same thing. When the distinguished name is displayed, state or province is displayed using 'ST='. State or province must be expressed using the same abbreviations used in mail addresses, for example, ST=IL for Illinois.
  - C=*country*—Specifies the country. This must be specified using the two-character ISO 3166 country code. For example, C=US for the United States of America, or C=VA for Vatican City. The codes are available at the ISO 3166 Maintenance Agency web site at:

    http://www.din.de/gremien/nas/nabd/iso3166ma/codlstpl/en_listpl.html

SIZe(*{key-size|1024}*)
Specifies the size of the private encryption key to be generated, in bits.

- 512—Specifies a low-strength key
- 768—Specifies a medium-strength key
- 1024—Specifies a high-strength key
- 2048—Specifies a very high strength key (only available using the PCICC keyword)

This keyword is mutually exclusive with the DSNAME keyword.

ICSF
Indicates that the generated private key will be placed in ICSF. If the DSN parameter was also specified and an existing certificate is to be replaced, the certificate will also be placed in ICSF.

ICSF must be active and configured for PKA operations. If it is not an error message will be displayed when attempting to insert or use the private key.

PCICC
Specifies that the key pair should be generated using the PCI Cryptographic Coprocessor and that the private key should be stored in ICSF. When PCICC is not specified, the key pair is generated using software. PCICC cannot be used with the DSN parameter or with the ICSF parameter. If a PCI cryptographic coprocessor is not present or operational, or if ICSF is not active or configured for PKA operations, an error message will be displayed and processing will terminate.

ACtive({*date-or-date-time*|*current-date-000000*|*current-date-time*})
Indicates the date and time that the certificate becomes active, for example 04/30/01-154403. If no time is specified, it defaults to 000000. If no date is specified, it defaults to the current day and time. **Note:** The year specified must fall within the range, 1950 - 2049. If an expire date is not also specified, the active year specified must fall within the range, 1950 - 2048, since the expire date defaults to the active day and time plus one year.

Valid date formats include: YY/MM/DD, MM/DD/YY, DD/MM/YY, and DD/MM/YYYY.

Expire({*date-or-date-time*|*current-date-000000*|*current-date-time*})**
Indicates the date and time that the certificate expires, for example MM/DD/YY. If no time is specified, it defaults to 000000. If no date is specified, it defaults to the active day and time plus 1 year. The year specified must fall within the range, 1950-2049. **Note:** The maximum value that can be specified for a certificate expiration date is December 31, 2049 when using eTrust CA-ACF2. Other platforms may have maximum expiration date values that are less than the maximum value that can be set by eTrust CA-ACF2. Use caution when setting an expiration date far into the future. If you will be passing such a certificate to another platform, be sure the expiration date falls within the guidelines of the other platform.

Valid date formats include: YY/MM/DD, MM/DD/YY, DD/MM/YY, and DD/MM/YYYY.

SIGnwith({CERTAUTH Label*(label-name)*|SITECERT Label*(label-name)*}), SIGnwith({CERTAUTH.*suffix* |SITECERT.*suffix*}), SIGnwith(Label*(label-name)*))
Specifies the certificate used to sign the new certificate. If SIGNWITH is not specified, a self-signed certificate is generated.

If SIGNWITH contains CERTAUTH or SITECERT, a suffix or Label value is used to specify which CERTAUTH or SITECERT certificate is used to sign the certificate.

If CERTAUTH or SITECERT are not specified, Label must be specified and the label will identify the user certificate that will sign the new certificate. The user id associated with the label is the user generating the certificate. This option cannot be specified if the certificate being generated is for a CERTAUTH or SITECERT id.

Note: For every one apostrophe desired in the Label value, two consecutive apostrophes must be specified. For example, the Label value, Frank's Certificate, should be specified as, Frank''s Certificate. If a single apostrophe is specified in the Label value, the value will be considered invalid.

Keyusage
Specifies the values of the keyUsage certificate extension. The default for certificate authority certificates is CERTSIGN. CERTSIGN is always set for certificate authority certificates even if not specified. There is no default for non-certificate authority certificates.

- HANDSHAKE—Sets the digitalSignature and keyEncipherment bits in the keyUsage extension. This allows identification and key exchange during security handshakes such as SSL.
- DATAENCRYPT—Sets the dataEncipherment bit in the keyUsage extension. This allows the certificate to be used for data encryption.
- DOCSIGN—Sets the nonRepudiation bit in the keyUsage extension. This allows the certificate to be used in a legally binding signature.
- CERTSIGN– Sets the keyCertSign and cRLSign bits in the keyUsage extension. This lets the certificate sign other digital certificates and CRLs.

  When KEYUSAGE is specified and the target ID is CERTAUTH and keyUsage is present in the PKCS #10 request specified in the DSNAME keyword, the request will fail if the certSign bit is turned off in the PKCS #10 request. Otherwise, the keyUsage extension is generated as indicated by the KEYUSAGE parameter. In addition, the certSign and cRLSign bits are turned on if not already specified in the KEYUSAGE parameter.

When KEYUSAGE is specified and the target ID is CERTAUTH but keyUsage is not specified in a PKCS #10 request, the extension are generated as indicated by the KEYUSAGE parameter. In addition, the certSign and cRLSign bits are turned on if not already specified in the KEYUSAGE parameter.

When KEYUSAGE is specified and the target ID is not CERTAUTH, the keyUsage extension is generated using the attributes specified in the KEYUSAGE parameter.

When KEYUSAGE is not specified and the target id is CERTAUTH and keyUsage is present in the PKCS #10 request specified in the DSNAME keyword, the request will fail if the certSign bit is turned off in the PKCS #10 request. Otherwise, the extension is generated using the attributes specified in the keyUsage extension in the PKCS #10 request.

When KEYUSAGE is not specified and the target id is CERTAUTH and the DSNAME keyword is not specified, the keyUsage extension is generated by turning on the certSign and cRLSign bits.

When KEYUSAGE is not specified and the target id is not CERTAUTH, the extension is generated using the attributes specified in the keyUsage extension in the PKCS #10 request, if present. If the keyUsage extension is not present in the PKCS #10 request or the DSNAME keyword was not specified, the keyUsage extension is not created.

ALtname

Specifies the values for the subjectAltName extension. One or more of the values might be specified. The parameter is optional and there is no default. If required, the entered values are converted to ASCII.

- IP=*numeric-ip-address*—Specifies a string containing a fully qualified numeric-ip-address in IPV4 dotted decimal format, which is four decimal numbers between 0 and 255 separated by periods:

  141.202.1.255

  The maximum field size is 15 bytes.

- DOMAIN=*internet-domain-name*—Specifies a fully qualified internet domain name, such as www.ca.com. The validity of this value is not checked. The maximum field size is 255 bytes.

- EMAIL=*email-address*—Specifies a fully qualified e-mail address such as frank01@bluelock.com. The maximum field size is 255 bytes.

- URI=*universal-resource-identifier*—Specifies a universal resource identifier such as http://www.ca.com The validity of this field is not checked. The maximum field size is 255 bytes.

Certificate Extensions

When a certificate is generated, certain extensions are created. If a PKCS #10 request is passed as input to GENCERT using the DSNAME parameter, certain extensions are copied from the PKCS #10 request. The logic for the keyUsage extension was listed previously under the KEYUSAGE parameter. The following is the logic for the other extension settings:

subjectKeyIdentifer

When DSNAME is specified, the subjectKeyIdentifier is copied from the PKCS #10 request, if it is present.

If DSNAME is not specified or if the PKCS #10 request does not contain a subjectKeyIdentifer, this extension is created according to Public Key Infrastructure Standards.

authorityKeyIdentifier

If SIGNWITH is specified, this extension is created using the subjectKeyIdentifier value in the SIGNWITH certificate.

If SIGNWITH is not specified or the SIGNWITH certificate does not contain a subjectKeyIdentifier extension, the authorityKeyIdentifier extension is not created.

basicConstraints

When the target ID is CERTAUTH, and basicConstraints is present in the PKCS #10 request, the command fails if the cA Boolean value is false. Otherwise the extension is generated turning the cA bit on. Pathlength is not included.

When the target ID is CERTAUTH and basicConstraints is not present in the PKCS #10 request, the extension is generated by turning the cA bit on. Pathlength is not included.

When the target ID is not CERTAUTH, and basicConstraints is coded in the PKCS #10 request, basicConstraints is generated using the values set in the PKCS #10 request including the pathLength. If basicConstraints was not present in a PKCS #10 request, the extension is not generated.

subjectAltName

When ALTNAME is specified the subjectAltName extension is generated using the ALTNAME values.

When ALTNAME is not specified and subjectAltName is present in a PKCS #10 request, the subjectAltName extension is generated using the values present in the PKCS #10 request. If subjectAltName is not present in a PKCS #10 request, the subjectAltName extension is not generated.

issuerAltName

When SIGNWITH is specified, the extension is generated using the subjectAltName value of the signing certificate if the extension is present. Otherwise the extension is not created.

When SIGNWITH is not specified, the issuerAltName extension is not created.

GENCERT Examples

1. Generate a self-signed certificate authority certificate from a PKCS #10 request.

```
gencert certauth.bluelock Subj(CN='Blue Lock Company Certificate Authority'
OU='Auditing Department' O='Blue Lock Company' C=US)
label(Audit CA) expire(12/31/2020)
```

This command will generate a self-signed certificate authority certificate. A label, Audit CA, was specified for the record and an expiration date of 12/31/2020. Be sure to place an expiration date on your CERTAUTH certificates. The default is one year from the day you create the record. Subsequent certificates signed with the certificate might get warning messages if the expiration date of the CERTAUTH certificate is not more than a year away. The subject's distinguished name as well as the issuer's distinguished name are generated using the inputted SUBJSDN value. Since the certificate is self-signed, the TRUST flag is automatically set and the serial number is 00.

The resulting CERTDATA record would look like this:

```
CERTDATA / CERTAUTH.BLUELOCK LAST CHANGED BY FRANK01 ON 02/02/02-10:34
          ISSUERDN(CN=Blue Lock Company Certificate
          Authority.OU=Auditing Department O=Blue Lock
          Company.C=US) CERTNSER(0000000000000001) KEYSIZE(1,024)
          LABEL(Audit CA) SERIAL#(00) SUBJDN(CN=Blue Lock Company Certificate
          Authority.OU=Auditing Department.O=Blue Lock
          Company.C=US) TRUST
Certificate is not connected to any key rings
```

2. Generate a SITECERT certificate for the company's web server.

```
gencert sitecert.blcweb Subj(CN='Blue Lock Web Server Certificate'
OU='Auditing Department'O='Blue Lock Company' C=US) label(BL Web Server)
signwith(certauth Label(Audit CA)) expire(12/31/2020)
```

This command generated a certificate signed by the CERTAUTH certificate we just created. The serial number and issuer's distinguished name were taken from the signing certificate. We expect the web server to be in place for quite some time so a generous expiration date was also supplied. The resulting CERTDATA record would look like this:

```
CERTDATA / SITECERT.BLCWEB LAST CHANGED BY FRANK01 ON 02/02/02-11:08
                 ISSUERDN(CN=Blue Lock Company Certificate
                 Authority.OU=Auditing Department.O=Blue Lock
                 Company.C=US) KEYSIZE(1,024)LABEL(BL Web Server)
                  SERIAL#(01) SUBJDN(CN=Blue Lock Web Server
                 Certificate.OU=Auditing Department.O=Blue Lock
                 Company.C=US)TRUST
Certificate is not connected to any key rings
```

3. Generate a user certificate, signed with a certificate authority certificate.

```
gencert frank01.cert Subj(cn='Frank Schwinger' OU='Sales Department' O='Blue
Lock Company' C=US) label(Frank01 Cert) signwith(certauth Label(Audit CA))
expire(12/31/2003)
```

This user certificate is also signed with the Audit CA CERTAUTH certificate. This certificate was given a less generous expiration date. We can renew Frank's certificate if he's still with the company next year. The certificate is trusted because the signing certificate is trusted.

```
CERTDATA / FRANK01.CERT LAST CHANGED BY CUNKE01 ON 02/02/02-11:23
               ISSUERDN(CN=Blue Lock Company Certificate
               Authority.OU=Auditing Department.O=Blue Lock
               Company.C=US) KEYSIZE(1,024) LABEL(Frank01 Cert)
               SERIAL#(02) SUBJDN(CN=Frank Schwinger.OU=Sales Department.
               O=Blue Lock Company.C=US) TRUST
Certificate is not connected to any key rings
```

GENREQ Subcommand

The GENREQ command is used to generate a certificate request to be sent to a Certification Authority. GENREQ extracts the subject's distinguished name and public key from an existing certificate, packages it in PKCS #10 format, signs it with the certificate's private key, base-64 encodes the result, and writes it to a data set. This request is sent to a Certification Authority, which will create a new certificate with the same distinguished name and public key, but issued and signed by the Certification Authority.

The GENREQ command can be issued in any mode of the ACF command. It has the following syntax:

```
GENReq {logonid|logonid.suffix}
       Dsname(data-set-name)
       [Label(label)]
```

logonid|logonid.suffix
Specifies the record key of the certificate to use to obtain the distinguished name and public key for the request, if Label is not also specified. This is a one- to eight-character logonid, or a logonid, a dot, and a one- to eight-character suffix.

If Label is also specified, logonid, rather than logonid.suffix, must be specified, and indicates the logonid that the label is associated with.

Dsname(*data-set-name***)**
Specifies the name of the data set into which the certificate request is written. The data set must not already exist. If the data set name is enclosed in single quotes, it is considered fully qualified and is used as specified. Otherwise, the user's prefix, as specified by the TSO PROFILE PREFIX command (or defaulted from the DFT-PFX field of the logonid record) is added to the front of the data set name.

Label(*label***)**
Specifies the label of the certificate used to obtain the distinguished name and public key for the request. Logonid must also be specified to indicate which logonid the label is associated with. **Note:** For every one apostrophe desired in the Label value, two consecutive apostrophes must be specified. For example, the Label value, Frank's Certificate, should be specified as, Frank''s Certificate. If a single apostrophe is specified in the Label value, the value will be considered invalid.

Examples

```
genreq frank01.cert dsn(TESTREQ.REQ)

genreq frank01 label(Frank01 Cert) dsn('joseph01.testreq2.req')
```

This command, issued by user FRANK01, generates a certificate request based on the FRANK01.CERT certificate and writes it to a data set named FRANK01.TESTREQ.REQ. The second command shows how a label is used to indicate which one of Frank's certificates is used in the genreq request.

The following is an example of a generated request.

```
Server:  CA-SAF   REL 1.3

Subject's distinguished name:

CN=Frank Schwinger
OU=Sales Department
O=Blue Lock Company
C=US

-----BEGIN NEW CERTIFICATE REQUEST-----
MIIBkwI4AsBf9R0wGwYJKoZIhvcNAQkBFg5kbWFnZWVAbWliLmNvbTEVMBMGA1UE
AxMMRGVubmlzIE1hZ2VlMQwwCgYDVQQFEwMwMDExDDAKBgNVBAogA01JQjELMAkG
A1UEBjCBnTANBgkqhkiG9w0BAQEFAAOBiwAwgYcCgYEA6SSBPS7HrK1WAOaU3QeN
g+F85qvzyPh+VZLhihFR6IdX149OtAIhQFG+479EnpW2prJyjFr2Xd19jV4QxCHZ
q8RYeVzU0+lrJPPRHLQGYUdx/lYvGv/LzwZOiWn+OwRdTqkxKTPr/IH0weIXW0Xg
j2rhi1YQK8xpm7IpdwEw+eECAQOgADqxBgkqhkiG9w0BAQQDgYEALsTCqYSqfLXH
9aZ8lx1tj0pBcsSIgqKf9BF2KxM2i9PfTxuqnuLt3dQcM/MBJp0oKvFlNaUfevkt
4eoljTkZZ+WBq4s9Lwx7c/K6O9CMGG59j2VvhxRBIbhhzQN1SoOX/tf50y6kQmMP
cnUi93gpQIaopR/zQvjJhUN7TZAwUJE
-----END NEW CERTIFICATE REQUEST-----
```

CONNECT Subcommand

The CONNECT subcommand is used to associate certificate information with a key ring. The CONNECT subcommand has the following syntax:

```
CONnect  Certdata(userid1.suffix) Keyring(userid2.suffix)
         [Ringname(ringname)]
         [Label(label)]
         [Usage({PERSONAL|CERTAUTH|SITE})]
         [DEFAULT]
```

Parameter Descriptions

CERTDATA(*userid1.suffix*)

Specifies the record key of a CERTDATA record to associate with a key ring. The userid1 is a one to eight-character userid associated with the CERTDATA record. The suffix is one to eight characters used to make the record key unique. The suffix is separated from the userid by a period. If LABEL is specified in addition to suffix, then both suffix and the label must refer to the same CERTDATA record.

KEYRING(*userid2.suffix*)

Specifies the record key of a KEYRING record to which the certificate information is to be associated. The userid2 is a one to eight-character userid associated with the KEYRING record. The suffix is one to eight characters used to make the record key unique. The suffix is separated from the userid by a period, and **cannot** be specified when the RINGNAME parameter is used.

RINGNAME(*ringname*)
Specifies the ring name of a KEYRING record to which the certificate information is to be associated. The ringname can be up to 237 characters in length.

LABEL(*label*)
Specifies the label of a CERTDATA record to associate with a key ring. The label can be up to 32 characters in length. It can contain blanks and mixed-case characters. **Note:** For every one apostrophe desired in the Label value, two consecutive apostrophes must be specified. For example, the Label value, Frank's Certificate, should be specified as, Frank"s Certificate. If a single apostrophe is specified in the Label value, the value will be considered invalid.

USAGE({PERSONAL|CERTAUTH|SITE})
Specifies how the certificate is to be used within the key ring. If USAGE is omitted, the usage associated with the certificate that is being connected is used.

- **PERSONAL**—Specifies that the certificate is to be used as a user certificate.
- **CERTAUTH**—Specifies that the certificate is to be used as a certificate authority certificate.
- **SITE**—Specifies that the certificate is to be used as a site certificate.

DEFAULT
Specifies that the certificate is to be the default certificate for the key ring. A key ring can have only one default certificate.

Examples

```
acf
connect certdata(user02.cert1) keyring(user01.ring) usage(site) default

acf
connect certdata(user02) label(user02 certificate) keyring(user01)
    ringname(user01 key ring)
```

REKEY Subcommand

The REKEY subcommand is used to create a new certificate from an existing certificate with a new public/private key pair. The REKEY command is the first step of a rekey rollover process to retire the use of an existing private key. The REKEY subcommand will also copy the subject's distinguished name, key usage and subject alternate name from the existing certificate. The new certificate is self-signed and saved under the same logonid or CERTAUTH or SITECERT.

```
REKey {logonid|logonid.suffix|CERTAUTH|CERTAUTH.suffix|SITECERT|SITECERT.suffix}
       [Label(existing-certificate-label)]
       [WITHLbl(new-certificate-label)]
       [WITHSfx(new-certificate-suffix)]
       [SIZe({key-size|1024})]
       [ICSF|PCICC]
       [ACtive({date-or-date-time|current-date-000000|current-date-time})]
       [Expire({date-or-date-time|current-date-000000|current-date-time})]
```

Parameter Descriptions

***logonid*|logonid.suffix|CERTAUTH|SITECERT**

The logonid specifies the user associated with the certificate. It may be a one to eight-character logonid, or a logonid, a dot, and a one to eight-character suffix. If label is specified, logonid, rather than logonid.suffix, must be specified, and indicates the logonid with which the label is associated.

Label(*label*)

Specifies a 1 to 32-character label of the existing certificate. The label can contain blanks and mixed-case characters. **Note:** For every one apostrophe desired in the Label value, two consecutive apostrophes must be specified. For example, the Label value, Frank's Certificate, should be specified as, Frank''s Certificate. If a single apostrophe is specified in the Label value, the value will be considered invalid.

WITHLbl(*label*)

Specifies a 1 to 32-character label that the new certificate will have. The WITHLBL value can contain blanks and mixed-case characters. It must be unique to the logonid with which the new certificate is associated. If WITHLBL is not specified, the label field of the new certificate defaults to the upper-case version of the logonid or logonid.suffix that was specified. **Note:** For every one apostrophe desired in the WITHLBL value, two consecutive apostrophes must be specified. For example, the WITHLBL value, Frank's Certificate, should be specified as, Frank''s Certificate. If a single apostrophe is specified in the WITHLBL value, the value will be considered invalid.

WITHSufx(*record suffix*)

Specifies a 1 to 8-character suffix of the new certificate. The new suffix can contain mixed-case characters but will be folded to upper case. The new suffix must be unique to the logonid with which the new certificate is associated. The suffix will be appended to the record key with a dot (.) preceding the suffix. If a suffix is not specified, the suffix will be in the format of AUTOnnn, where nnn is a number from 001 to 999.

SIZe({*key-size*|*1024*})
Specifies the size of the private encryption key to be generated, in bits.

- 512—Specifies a low-strength key
- 768—Specifies a medium-strength key
- 1024—Specifies a high-strength key

ICSF
Indicates that the generated private key will be placed in ICSF. ICSF must be active and configured for PKA operations. If it is not an error message will be displayed when attempting to insert or use the private key.

PCICC
Specifies that the key pair should be generated using the PCI Cryptographic Coprocessor and that the private key should be stored in ICSF. When PCICC is not specified, the key pair is generated using software. PCICC cannot be used with the ICSF parameter. If a PCI cryptographic coprocessor is not present or operational, or if ICSF is not active or configured for PKA operations, an error message will be displayed and processing will terminate.

If neither ICSF nor PCICC is specified, the key pair is generated using software and stored in the eTrust CA-ACF2 database.

ACtive({*date-or-date-time*|*current-date-000000*|*current-date-time*})
Indicates the date and time that the certificate becomes active, for example 04/30/01-154403. If no time is specified, it defaults to 000000. If no date is specified, it defaults to the current day and time. Note: The year specified must fall within the range, 1950 - 2049. If an expire date is not also specified, the active year specified must fall within the range, 1950 -2048, since the expire date defaults to the active day and time plus one year.

Valid date formats include: YY/MM/DD, MM/DD/YY, DD/MM/YY, and DD/MM/YYYY.

Expire({*date-or-date-time*|*current-date-000000*|*current-date-time*})
Indicates the date and time that the certificate expires, for example MM/DD/YY. If no time is specified, it defaults to 000000. If no date is specified, it defaults to the active day and time plus 1 year. Note: The year specified must fall within the range, 1950 - 2049.

Valid date formats include: YY/MM/DD, MM/DD/YY, DD/MM/YY, and DD/MM/YYYY.

Examples

```
REKEY CERTAUTH.LOCALCA WITHLBL(Local CA 2004) SIZE(1024) EXPIRE(12/31/14)

REKEY CERTAUTH LABEL(Local CA) WITHLBL(Local CA 2004) EXPIRE(12/31/19)

REKEY CERTAUTH LABEL(Local CA) WITHLBL(Local CA 2004) WITHSUFX(LOCAL04)
EXPIRE(12/31/19)
```

REMOVE Subcommand

The REMOVE subcommand is used to disassociate a certificate from a key ring. The REMOVE subcommand has the following syntax:

```
REMove  Certdata(userid1.suffix) Keyring(userid2.suffix)
        [Ringname(ringname)]
        [Label(label)]
```

Parameter Descriptions

CERTDATA(*userid1.suffix*)
Specifies the record key of a CERTDATA record to remove from a key ring. The userid1 is a one to eight character userid associated with the CERTDATA record. The suffix is one to eight characters used to make the record key unique. The suffix is separated from the userid by a period. If LABEL is specified in addition to suffix, then both suffix and the label must refer to the same CERTDATA record.

KEYRING(*userid2.suffix*)
Specifies the record key of a KEYRING record from which to remove the certificate. The userid2 is a one to eight character userid associated with the KEYRING record. The suffix is one to eight characters used to make the record key unique. The suffix is separated from the userid by a period, and **cannot** be specified when RINGNAME is used.

RINGNAME(*ringname*)
Specifies the ring name of a KEYRING record from which to remove the certificate. The ringname can be up to 237 characters in length.

LABEL(*label*)
Specifies the label of a CERTDATA record from which to disassociate a key ring. The label can be up to 32 characters in length. It can contain blanks and mixed-case characters. **Note:** For every one apostrophe desired in the Label value, two consecutive apostrophes must be specified. For example, the Label value, Frank's Certificate, should be specified as, Frank''s Certificate. If a single apostrophe is specified in the Label value, the value will be considered invalid.

Examples

```
acf
remove certdata(user02.cert1) keyring(user01.ring)
```

```
acf
remove certdata(user02) label(user02 certificate) keyring(user01)
    ringname(user01 key ring)
```

ROLLOVER Subcommand

The ROLLOVER subcommand is the final step in the rekey – rollover process. ROLLOVER specifies the old certificate that is to be superceded by the new certificate. The ROLLOVER subcommand will perform the following actions:

- Delete the private key of the old certificate (as specified by the LABEL keyword), so that it can no longer be used to sign or encrypt and documents or certificates.
- Replaces the old certificate with the new certificate (as specified by the NEWLABEL keyword) in every key ring that the old certificate is connected to.
- Copies the serial number base from the old certificate to the new certificate.

When the rollover is complete, the new certificate is used as if it were the old certificate. The old certificate will still be available to verify signatures and decrypt data, but can no longer be used to sign or encrypt.

```
ROllover{logonid|logonid.suffix|CERTAUTH|CERTAUTH.suffix|SITECERT|
SITECERT.suffix}
        [Label(old-certificate-label)]
        [NEWLabel(new-certificate-label)|NEWSufx(new-certificate-suffix)]
[Force]
```

Parameter Descriptions

***logonid*|logonid.suffix|CERTAUTH|SITECERT**

The logonid specifies the user associated with the certificate. It may be a one to eight-character logonid, or a logonid, a dot, and a one to eight-character suffix. If label is specified, logonid, rather than logonid.suffix, must be specified, and indicates the logonid with which the label is associated.

Label(*label*)

Specifies a 1 to 32-character label of the old (source) certificate. This is the certificate that will have it's private key deleted. The label can contain blanks and mixed-case characters. **Note:** For every one apostrophe desired in the Label value, two consecutive apostrophes must be specified. For example, the Label value, Frank's Certificate, should be specified as, Frank''s Certificate. If a single apostrophe is specified in the Label value, the value will be considered invalid.

NEWLabel(*label*)

Specifies a 1 to 32-character label of the new (target) certificate. This is the certificate that will replace the old certificate in all the key rings that had the old certificate connected. The NEWLABEL value can contain blanks and mixed-case characters. It must be unique to the logonid with which the new certificate is associated. If a NEWLABEL is not specified, then NEWSUFX must be specified. **Note:** For every one apostrophe desired in the NEWLABEL value, two consecutive apostrophes must be specified. For example, the NEWLABEL value, Frank's Certificate, should be specified as, Frank''s Certificate. If a single apostrophe is specified in the NEWLABEL value, the value will be considered invalid.

NEWSufx(*record suffix*)

Specifies a 1 to 8-character suffix of the new (target) certificate. The new suffix can be used in place of the NEWLABEL field.

Force

Specifies that eTrust CA-ACF2 should bypass the following checks and perform the rollover unconditionally.

1. The values of LABEL and NEWLABEL must be different. If NEWSUFX is specified instead of NEWLABEL, the label of the new certificate must be different than the LABEL value.
2. The certificates identified by LABEL and NEWLABEL (or NEWSUFX) must both have private keys associated with them.
3. The certificate identified by NEWLABEL (or NEWSUFX) must have never been the target of a previously issued ROLLOVER subcommand and never used to sign other certificates.

When the FORCE keyword is specified, the previous three checks are not performed.

Note: The ROLLOVER subcommand has a degenerative feature where the private key of the certificate is deleted if both LABEL and NEWLABEL are the same and the FORCE keyword is also used.

Examples

```
ROLLOVER CERTAUTH.LOCALCA NEWLABEL(Local CA 2004)
ROLLOVER CERTAUTH LABEL(Local CA) NEWLABEL(Local CA 2004)
```

Associating a Digital Certificate with a User

Certificates are associated to eTrust CA-ACF2 users through the use of profile records. The CERTDATA segment of the USER profile identifies an X.509 digital certificate associated with a user. A user can have more than one certificate, but a single certificate cannot be used by more than one user. Profile records are inserted using the userid as the record key with or without a specified label (if no label is specified, one is set by default). The record key can also contain a userid with a distinguished suffix if a user is defined with more than one certificate.

CERTDATA Profile Data Records

A description of the CERTDATA profile data record fields follows.

Record ID	Fields
recid	ACTIVE(*date*) CERTNSER(*hex*) DSN(*data-set-name*) EXPIRE(*date*) ICSF ISSUERDN(*dn*) LABEL(*label*) NEWLABEL(*label*) PASSWORD(*password*) SERIAL#(*serial-number*) HITRUST | TRUST | NOTRUST

Field Descriptions

recid

Specifies the userid that is to be associated with the certificate. It can be a one to eight-character logonid, or a logonid, a dot, and a one to eight-character suffix. If label is also specified, logonid, rather than logonid.suffix, must be specified, and indicates the logonid with which the label is associated.

Using CERTAUTH in place of a logonid designates the certificate as a Certification Authority certificate.

Using SITECERT in place of a logonid designates the certificate as a site certificate.

Active(*date*)

Specifies an optional activation date in the format "mm/dd/yy", "dd/mm/yy", or "yy/mm/dd". The format you specify is determined by the DATE field of the GSO OPTS record. This date is not the same as the not-before validity date in the certificate itself. This date gives the security administrator the ability to specify when the profile record associating the user to the certificate becomes active. This date must fall within the range of the certificate's not-before and not-after validity dates and must be earlier than the CERTDATA record expiration date, if one exists.

CERTNSER(*hex value*)

Indicates the next serial number to be used by this certificate when signing another certificate. This field cannot be altered, which prevents generation of duplicate certificates. All detected duplicate certificates will be unusable on z/OS systems.

DSN(*data-set-name*)

Specifies the z/OS dataset that contains the digital certificate that is inserted into a CERTDATA profile record. The data set must be defined as physical sequential (DSORG=PS) and variable-blocked (RECFM=VB) and must be catalogued. If the data set name is enclosed in single quotes, it is considered fully qualified and is used as specified. Otherwise, the user's prefix, as specified by the TSO PROFILE PREFIX command (or defaulted from the DFT-PFX field of the logonid record) is added to the front of the data set name.

The data set specified may contain a single certificate or a certificate package. A PKCS7 certificate package contains a certificate and a chain of CA certificates. A PKCS12 certificate package contains a certificate, a private key, and a chain of CA certificates.

You may now insert all of the certificates contained in a PKCS7 certificate package. When a certificate is inserted from a dataset set and the DSN parameter contains the name of a PKCS7 certificate package, each of the CA certificates will be added to the database from the highest CA certificate in the chain to the lowest certificate in the chain. The trust status of the first CA certificate added will take the value specified on the insert command. The other CA certificates that are added will take the trust value of the signing certificate. If a certificate is expired or it's validity period is not complete within the validity of it's signing certificate or if the signing certificate of the certificate is not in the input certificate package or in eTrust CA-ACF2, then the certificate is added with a trust status of NOTRUST. If a CA certificate is already known to eTrust CA-ACF2, the certificate will retain it's trust status. A label of AUTOxxx will be generated for each CA certificate added where xxx is an available number between 0 and 1000.

If an error occurs during the addition of certificates from a PKCS 7 or PKCS 12 certificate package, any CERTAUTH certificates already added will not be removed.

Expire(*date*)
Specifies an optional expiration date in the format "mm/dd/yy", "dd/mm/yy", or "yy/mm/dd". The format you specify is determined by the DATE field of the GSO OPTS record. This date is not the same as the not-after validity date in the certificate itself. This date gives the security administrator the ability to specify when the profile record associating the user to the certificate expires. This date must fall within the range of the certificate's not-before and not-after validity dates and must be later than the CERTDATA record activation date, if one exists.

ICSF
Indicates that the generated private key will be placed in ICSF. ICSF must be active and configured for PKA operations. If it is not, an error message will be displayed when attempting to insert the private key.

ISSUERDN(*dn*)
Specifies that the ISSUERDN is the certification authority's distinguished name as extracted from the certificate. This field cannot be altered. However, it can be used with the SERIAL# operand to list, change, or delete a CERTDATA record. **Note:** ISSUERDN cannot be abbreviated.

Label(*label*)
Specifies a 32-character label to be associated with the certificate. The label can contain blanks and mixed-case characters. If a label is not specified, the label field will default to the upper-case version of the logonid or logonid.suffix that was specified. **Note:** For every one apostrophe desired in the Label value, two consecutive apostrophes must be specified. For example, the Label value, Frank's Certificate, should be specified as, Frank''s Certificate. If a single apostrophe is specified in the Label value, the value will be considered invalid. To change the label in a record, the NEWLABEL operand must be used. **Note:** LABEL cannot be abbreviated on a LIST or DELETE command.

NEWLABEL(*label*)
Specifies the 32-character label, which is to replace an existing label associated with a certificate. The label can contain blanks and mixed-case characters. **Note:** For every one apostrophe desired in the NEWLABEL value, two consecutive apostrophes must be specified. For example, the NEWLABEL value, Frank's Certificate, should be specified as, Frank''s Certificate. If a single apostrophe is specified in the NEWLABEL value, the value will be considered invalid. NEWLABEL can only be specified on a CHANGE command. If Label rather than NEWLABEL is encountered in the CHANGE command input, a CHANGE using Label request is assumed to be in effect.

Password(*password*)
Specifies the password that was used to encrypt the PKCS #12 certification package.

SERIAL#(*serial_number*)
Specifies the serial number as extracted from the certificate. This field cannot be altered. However, it can be used with the ISSUERDN operand to list, change or delete a CERTDATA record. **Note:** SERIAL# cannot be abbreviated.

HITRUST | TRUST |NOTRUST
Specifies a trust status for the certificate. If a trust status is not specified, it is set by default based on the validity of the certificate (self-signed) or, if the certificate was signed by another certificate, the validity of the signing certificate and the private key.

- HITRUST indicates that the certificate is both highly trusted and trusted. Any certificate usage applying to trusted certificates applies to highly trusted certificates. However, only certificate authority certificates (CERTAUTH) can be highly trusted.
- TRUST indicates that the certificate is trusted, i.e., that the certificate is valid for the user, site or certificate authority, and the private key has not been compromised.

  If no trust status has been specified and the certificate is self-signed, by default, the status will be set to TRUST.

  If no trust status has been specified and the certificate was signed by another certificate, by default, the status will be set to TRUST only if the following conditions are met:

  - The signing certificate can be located in the database.
  - The signing certificate is a Certification Authority (CERTAUTH).
  - The signing certificate is not expired.
  - The signing certificate's signature is valid.
  - The validity dates of the certificate being added fall within the range of the signing certificate's validity dates.

  Otherwise, the certificate will be inserted as not trusted (NOTRUST) and an informational message will be issued stating the reason why. However, if the signing certificate's signature is invalid, the certificate is not inserted.

  USER certificate—TRUST indicates that the certificate can be used to authenticate a userid.

  SITECERT certificate—TRUST indicates that the certificate can be used without authenticating it.

  CERTAUTH certificate—TRUST indicates that the certificate can be used to authenticate other certificates.

- NOTRUST indicates that the certificate is not trusted. If NOTRUST is specified or set by default, when the CERTDATA record is displayed, no trust value will be displayed; the trust field will remain blank.

```
change frank01.mycert NOTRUST
   CERTDATA / FRANK01.mycert LAST CHANGED BY FRANK01 ON 01/14/02-15:41
          ISSUERDN(CN=this is a CA.OU=acf2) LABEL(CERTAUTH.CERT4) SERIAL#(00)
          SUBJDN(CN=this is a CA.OU=acf2)
```

Creating CERTDATA Profile Records

As part of the INSERT command processing, the certificate is read from the z/OS data set. eTrust CA-ACF2 decodes the certificate to extract the serial number, issuer's distinguished name, and other information and inserts some of this information along with information from the command input into a CERTDATA profile data record. This information can be displayed with the LIST command.

Use the following command to enter profile administration mode:

```
set profile(user) div(certdata)
```

Use the following commands to create a CERTDATA profile data record:

```
insert frank01.mycert dsn('frank01.mycert') active(02/02/02)
expire(02/02/03) trust

CERTDATA / FRANK01.MYCERT LAST CHANGED BY FRANK01 ON 02/02/02-17:24
                  ACTIVE(02/02/02) EXPIRE(02/02/03) ISSUERDN(CN=CA cert20)
                  LABEL(FRANK01.MYCERT) SERIAL#(01)
                  SUBJDN(CN=frank01.mycert) TRUST
PROFILE
```

To create a record using logonid and label:

```
insert frank01 label(mycert) dsn('frank01.mycert') active(02/02/02)
expire(02/02/03) trust

CERTDATA / FRANK01.MYCERT LAST CHANGED BY FRANK01 ON 02/02/02-17:24
                  ACTIVE(02/02/02) EXPIRE(02/02/03) ISSUERDN(CN=CA cert20)
                  LABEL(MYCERT) SERIAL#(01)
                  SUBJDN(CN=frank01.mycert) TRUST
PROFILE
```

Certificate Replacement ("Renewal")

As part of the INSERT command processing, a certificate can be replaced without deleting and reinserting it. To replace an existing certificate, make sure that one of the following three cases is satisfied:

Case 1. The certificate being added is a duplicate of the existing certificate (i.e., has the same serial number and issuer's distinguished name) and the labels and record keys of both certificates are the same;

Case 2. The certificate being added is NOT a duplicate of the existing certificate, has the same subject's distinguished name, issuer's distinguished name, and public key as the existing certificate, the end date and time on the certificate being added is later than on that of the existing certificate, the existing certificate is not expired, and the record keys of both certificates are the same;

Case 3. The certificate being added is NOT a duplicate of the existing certificate, has the same public key as the existing certificate, there is a private key associated with the existing certificate in the database, the existing certificate is NOT expired, and the record keys of both certificates are the same.

Activating New CERTDATA Profile Records

Use the following console command to activate a newly created CERTDATA profile data record:

```
f acf2,rebuild(usr),class(p)
f acf2,omvs
```

Changing CERTDATA Profile Records

Use the following command to enter profile administration mode.

```
set profile(user) div(certdata)
```

Use the CHANGE command to change **only** the following fields in the CERTDATA profile record:

- active date - [Active(*date*)]
- expire date - [Expire(*date*)]
- label - [NEWLABEL(*label*)]
- trust status - [HITRUST | TRUST | NOTRUST]

Use the following command to change a single record using record ID:

```
change frank01.mycert active(02/10/02) expire(02/20/03) newlabel(new certificate)
notrust

CERTDATA / FRANK01.MYCERT LAST CHANGED BY FRANK01 ON 02/02/02-17:24
                  ACTIVE(02/10/02) EXPIRE(02/20/03) ISSUERDN(CN=CA cert20)
                  LABEL(new certificate) SERIAL#(01)
                  SUBJDN(CN=frank0l.mycert)
PROFILE
```

Use the following command to change a single record using record ID and label:

```
change frank01 label(FRANK01.MYCERT) active(02/10/02) expire(02/20/03)
newlabel(new certificate) trust

CERTDATA / FRANK01 LAST CHANGED BY FRANK01 ON 02/02/02-17:24
                  ACTIVE(02/10/02) EXPIRE(02/20/03) ISSUERDN(CN=CA cert20)
                  LABEL(new certificate) SERIAL#(01)
                  SUBJDN(CN=frank0l.mycert) TRUST
PROFILE
```

Use the following command to change a single record using SERIAL # and ISSUERDN:

```
change userid SERIAL#(serial_number) ISSUERDN(dn) active(date) expire(date)
newlabel(label) [hitrust|trust|notrust]
```

Use the following command to change multiple records:

```
change like(frank01.-) active(02/10/02) expire(02/20/03) trust
  ACF6D071 2 RECORDS CHANGED
PROFILE
```

Note: The label cannot be changed on a multiple (masked) record request.

Activating Changed CERTDATA Profile Records

Use the following console command to activate a newly changed CERTDATA profile data record:

```
f acf2,rebuild(usr),class(p)
f acf2,omvs
```

Viewing CERTDATA Profile Records

Use the following command to enter profile administration mode.

```
set profile(user) div(certdata)
```

Use the LIST command as follows to view CERTDATA profile records.

To list a single record using record ID:

```
list frank01.mycert

CERTDATA / FRANK01.MYCERT LAST CHANGED BY FRANK01 ON 02/02/02-17:24
                  ACTIVE(02/10/02) EXPIRE(02/20/03) ISSUERDN(CN=CA
                        cert20) LABEL(FRANK01.MYCERT) SERIAL#(01)
                        SUBJDN(CN=frank0l.mycert)
PROFILE
```

To list a single record using record ID and label:

```
list frank01 label(MYCERT)

CERTDATA / FRANK01 LAST CHANGED BY FRANK01 ON 02/02/02-17:24
                  ACTIVE(02/10/02) EXPIRE(02/20/03) ISSUERDN(CN=CA)
                   LABEL(MYCERT) SERIAL#(01)
                  SUBJDN(CN=frank0l) TRUST
PROFILE
```

Use the following command to list a single record using SERIAL # and ISSUERDN:

```
list userid SERIAL#(serial_number) ISSUERDN(dn)
```

Use the following command to list multiple records:

```
list like(frank01.-)

CERTDATA / FRANK01.MYCERT1 LAST CHANGED BY FRANK01 ON 02/02/02-17:24
                  ACTIVE(02/10/02) EXPIRE(02/20/03) ISSUERDN(CN=CA
                   cert20) LABEL(FRANK01.MYCERT1) SERIAL#(01)
                  SUBJDN(CN=frank0l.mycert1) TRUST
Certificate is not connected to any key rings


CERTDATA / FRANK01.MYCERT2 LAST CHANGED BY FRANK01 ON 02/02/02-17:24
                  ACTIVE(02/10/02) EXPIRE(02/20/03) ISSUERDN(CN=CA cert2)
                   LABEL(FRANK01.MYCERT2) SERIAL#(01)
                  SUBJDN(CN= frank0l.mycert2) TRUST
Certificate is not connected to any key rings


CERTDATA / FRANK01.MYCERT3 LAST CHANGED BY FRANK01 ON 02/02/02-17:24
                  ACTIVE(02/10/02) EXPIRE(02/20/03) ISSUERDN(CN=CA cert21)
                   LABEL(FRANK01.MYCERT3) SERIAL#(01)
                  SUBJDN(CN=frank0l.mycert3) TRUST
Certificate is not connected to any key rings
PROFILE
```

Deleting CERTDATA Profile Records

Use the following command to enter profile administration mode.

```
set profile(user) div(certdata)
```

Use the DELETE command as follows to delete a specific CERTDATA profile record.

Use the following command to delete a single record using record ID:

```
delete frank01.mycert

ACF6D073 CERTDATA / FRANK01.MYCERT RECORD DELETED
PROFILE
```

Use the following command to delete a single record using record ID and label:

```
delete frank01 label(MYCERT)
ACF6D073 CERTDATA / FRANK01 RECORD DELETED
PROFILE
```

Use the following command to delete a single record using SERIAL # and ISSUERDN:

```
delete userid SERIAL#(serial_number) ISSUERDN(dn)
```

Use the following command to delete multiple records:

```
delete like(userid.-)
ACF6D072 2 RECORDS CHANGED
PROFILE
```

Automatic Registration of Digital Certificates

CERTDATA profile records can also be dynamically inserted or deleted through a process known as Automatic Registration of Digital Certificates. An installation-written or vendor-provided CGI program requests automatic registration by invoking a z/OS UNIX callable service. A program of this sort would typically validate a user's digital certificate and then prompt the user for an ID and password, which are then validated by eTrust CA-ACF2. If the validation is successful, the certificate is presented to eTrust CA-ACF2 and a CERTDATA profile record is dynamically created and associated with that user.

Dynamically inserted profile records can be distinguished in two ways: first, the record key contains the word AUTO*nnn* as the suffix, where *nnn* is a numeric value; second, the DSN field is blank since the certificate was not read from a z/OS data set.

A user must be authorized through the SAF FACILITY class before a profile record is dynamically inserted or deleted. The FAC resource rule $KEY values to allow dynamic INSERT and DELETE are:

```
IRR.DIGTCERT.ADD
IRR.DIGTCERT.DELETE
```

The following is a rule example:

```
SET RESOURCE(FAC)

COMPILE

$KEY(IRR) TYPE(FAC)
 DIGTCERT.ADD UID(userid) ALLOW
 DIGTCERT.DELETE UID(userid) ALLOW

STORE
```

Note: Ensure that you do not inadvertently allow access to these resources because of masking in your resource rules.

Associating Multiple Digital Certificates with a User

z/OS supports key rings for digital certificates. A key ring lets you associate one or more digital certificates to a particular user (logonid). These are certificates that can be used by the user. The application using the z/OS Unix System Services R_datalib function can request that eTrust CA-ACF2 pass back one or more trusted digital certificates based on the key ring value passed to it. The application asks for a trusted certificate by supplying a certificate label or the subject's distinguished name, or by requesting the default certificate. If the caller is authorized to make the request, eTrust CA-ACF2 passes back a trusted certificate and the application uses this to validate the user. If the validation fails, the caller can process each certificate in the ring in turn until it finds one that is accepted, or until it has processed all of the certificates in the specified key ring.

eTrust CA-ACF2 administers key ring support using profile records. The KEYRING segment of the USER Profile lets you create, change, display, or delete a key ring for a user using the normal INSERT, CHANGE, LIST, and DELETE commands. The key of the record consists of the logonid of the user for whom the key ring is created, and a qualifier to make the record unique.

KEYRING Profile Records

A description of the KEYRING profile data record fields follows:

Record ID	Fields
Recid	DEFAULT(*userid.suffix*) RINGNAME(*ringname*)

recid
This is a one to eight-character userid that is to be associatd with the key ring. A one to eight-character suffix can be appended to the userid to create a unique record key. The suffix must be separated from the userid with a period.

DEFAULT(*userid.suffix*)
Specifies a default certificate for this key ring. The field contains the name of a CERTDATA profile record used to register the digital certificate. Only one default certificate can be specified. The field is optional. The length of the field is 16-characters. **Note:** If a default certificate is specified when the KEYRING is being inserted, that certificate must be connected first before attempting to connect any other certificates.

RINGNAME(*ringname*)
Specifies the name of the key ring. This name is used to connect CERTDATA records to the key ring or to remove CERTDATA records from the key ring. The length of the field is 237 characters. This field is required.

The following is an example of creating a key ring profile record:

```
SET PROFILE(USER) DIV(KEYRING)
INSERT usera.ring DEFAULT(usera.cert1) RINGNAME(usera's keyring)

KEYRING  /  USERA.RING LAST CHANGED BY SECID01 ON 04/01/02 - 14:46
        DEFAULT(USERA.CERT1) RINGNAME(USERA'S KEYRING)
```

Once created, the key ring profile can have certificates associated with it. The method of controlling these certificates is by using the CONNECT and REMOVE commands. To add a certificate to a key ring, use the CONNECT command. To delete a certificate from a key ring, use the REMOVE command. There are two required fields used with these commands:

CERTDATA
Specifies the key of a CERTDATA record for a specific certificate. This field is required.

KEYRING
Specifies the key of the KEYRING profile record being operated on. This field is required.

The following is an example of adding a certificate to a key ring and then deleting it from the key ring:

```
CONNECT CERTDATA(usera.cert2) KEYRING(usera.ring)

ACF68011 CERTIFICATE SUCCESSFULLY CONNECTED TO THE KEY RING

REMOVE CERTDATA(usera.cert2) KEYRING(usera.ring)

ACF68015 CERTIFICATE SUCCESSFULLY REMOVED FROM THE KEY RING
```

Mapping Multiple Digital Certificates to a User ID

Digital certificates are being used more frequently as a means of identifying users in today's computing environment. As the number of people using certificates grows, so does the amount of work required to manage certificates. One method to manage this workload is the ability to map multiple certificates to a single logonid. This process, referred to as "digital certificate name filtering", is available at z/OS and is supported in eTrust CA-ACF2.

The CERTMAP and CRITMAP data records in the Global Systems Options (GSO) Record allow mapping of multiple certificates to a single eTrust CA-ACF2 logonid and are discussed below.

Digital Certificate Name Filtering

The filtering process lets a site map any number of certificates to a single logonid using two pieces of information contained in a certificate:

- Issuer's Distinguished Name (IDN)
- Subject's Distinguished Name (SDN)

The issuer is the certificate authority (CA) that issued the certificate. The subject is the individual to whom the certificate is issued. A distinguished name is the field that contains information that relates to the identity of the issuer or the subject. The identifiers in a distinguished name can include any or all of the following from least specific to most specific (The abbreviation used in the certificate is in parentheses):

- Country (C=)
- State or province (ST=)
- City or locality (L=)
- Postal code (PC=)
- Street (STREET=)
- Company or organization (O=)
- Division or organizational unit (OU=)
- Email address (E=)
- Title (T=)
- Common name (CN=)

The filter process lets you define which logonid to assign a particular certificate based on the information contained in a CERTMAP record. For example, assume that we have four certificates created by the same issuer with the IDN containing the values:

```
OU=CA Class 1.O=CA.L=internet
```

Assume also that the SDN for each certificate has values as follows:

```
CN=Bob.OU=LVL2.OU=Support.OU=ACF2.O=CA, Inc
CN=Earl.OU=LVL1.OU=Support.OU=ACF2.O=CA, Inc
CN=Pearl.OU=LVL1.OU=Support.OU=ACF2.O=CA, Inc
CN=Tony.OU=Devlopment.OU=ACF2.O=CA, Inc
```

If you want to assign a logonid for all level 1 personnel in support for access to the system, you could create a CERTMAP record referencing the logonid TECH01 that contains the values:

```
OU=LVL1.OU=Support.OU=ACF2.O=CA, Inc
```

Then, when Earl or Pearl access the system, the filtering process based on the described CERTMAP record would assign the logonid TECH01 as specified in the CERTMAP record.

Filtering Logic Processing

When a user attempts to access the system using a digital certificate, initACEE processing gets invoked. This processing attempts to make the most specific match it can of a certificate to a logonid. The logic of this processing is as follows:

1. Attempt to match the certificate to an existing CERTDATA record.
2. If no CERTDATA record matches, use the best-fit CERTMAP record that matches both the full IDN and a full or partial SDN.
3. If no CERTMAP record matches step 2, use the best-fit CERTMAP record that matches a full or partial SDN.
4. If no CERTMAP record matches step 3, use the best-fit CERTMAP record that matches a full or partial IDN.

Partial means removing levels of the distinguished name starting with the common or user name (CN=) and working through the filter until a match is found. The IDN or SDN are listed from most specific to least specific.

In addition to mapping to a specific logonid, you can use a CERTMAP record to map to multiple logonids. If the filtering process encounters a MULTIID value in the CERTMAP record, this indicates that multiple logonids are involved and determines the actual logonid to be assigned by using system or user variables such as SYSID, APPLID, or an application dependent variable list.

The logonid used in this case is found in the CRITMAP record. The processing logic for this is as follows:

1. Find the CERTMAP record as normal.
2. If MULTIID is specified on the CERTMAP record, use system variables (SYSID or APPLID) and/or application defined variables passed on the initACEE call to find a matching CRITMAP record.
3. Assign the logonid specified in the CRITMAP record.

For example, using the certificates that were previously described, assume we want to assign logonids based on the application that the users invoke when they access our site over the internet with a certificate provided by the CA (certificate authority). The internet applications are the WEBINSUR and WEBBANK pages. Using a CERTMAP record and two CRITMAP records, you can assign a different logonid based on the application:

```
CERTMAP data: IDNFILTR('L=Internet.O=CA') MULTIID CRITERIA(APPLID=&applid)
CRTIMAP data: APPLID(WEBBANK) USER(BANKU)
CRITMAP data: APPLID(WEBINSUR) USER(INSURU)
```

Using these records, eTrust CA-ACF2 assigns the logonid BANKU for the WEBBANK application and the logonid INSURU for the WEBINSUR application.

Certificate Name Filtering Options (CERTMAP)

Certificate Name Filtering allows the mapping of multiple digital certificates to a single eTrust CA-ACF2 logonid. Optionally, a single digital certificate can be mapped to one of a number of eTrust CA-ACF2 logonids based on the system ID, application ID, or application defined variables.

Record ID	Fields
CERTMAP.*recid*	SDNFILTR(*subject's-dist-name-filter*) IDNFILTR(*issuer's-dist-name-filter*) DSN(*data-set-name*) CRITERIA(*criteria-name-template*) LABEL(*32-byte-label*) TRUST | NOTRUST USERID(*userid-to-map-to*) MULTIID | NOMULTIID

Field Descriptions

recid
Specifies the unique one to eight-character name of the record ID.

SDNFILTR('*subject's-distinguished-name-filter*')
Specifies the "significant" portion of the subject's distinguished name. This is the part of the name that is used as a filter when associating a logonid with a certificate. If this field contains blanks, the entire filter must be encased in quotes, otherwise they are not needed.

When the CERTMAP record is inserted or changed without the DSN parameter, you must specify the entire portion of the distinguished name to be used as the filter.

The format of the *subject's-distinguished-name-filter* variable is similar to the output displayed when a certificate is displayed with the LIST command. It is an X.509 distinguished name in an address type format:

```
component.component.component.component...
```

Or, more specifically:

```
Qualifier1=node1.qualifier2=node2....qualifiern=noden
```

For example:

```
SDNFILTR('CN=John Jones.OU=Accounts Payable.O=My Company.L=internet')
```

The value specified for SDNFILTR must begin with a prefix found in the following list, followed by an equal sign (X'7E'). Each component should be separated by a period (X'4B'). The case, blanks, and punctuation displayed when the digital certificate information is listed must be maintained in the SDNFILTR. Since digital certificates only contain characters available in the ASCII character set, the same characters should be used for the SDNFILTR value. Valid prefixes are:

```
COUNTRY             Specified as C=
STATE/PROVINCE      Specified as ST=
LOCALITY            Specified as L=
ORGANIZATION        Specified as O=
ORGANIZATIONAL UNIT Specified as OU=
TITLE               Specified as T=
COMMON NAME         Specified as CN=
STREET Address      Specified as STREET=
POSTAL CODE (Zip)   Specified as PC=
```

A maximum of 255 characters can be entered for SDNFILTR.

SDNFILTR is optional if IDNFILTR is specified. If SDNFILTR is not specified, only the issuer's name is used as a filter. SDNFILTR must not be specified with IDNFILTR unless the value of IDNFILTR results in the entire issuer's name being used in the filter. Note that subject's name can be partial but cannot be used in a filter that contains only a partial issuer's name.

IDNFILTR('*issuer's-distinguished-name-filter*')

Specifies the "significant" portion of the issuer's distinguished name that is used as a filter when associating a logonid with a certificate. If this field contains blanks, the entire filter must be encased in quotes; otherwise they are not needed.

When the CERTMAP record is inserted or changed without the DSN parameter, you must specify the entire portion of the distinguished name to be used as the filter.

The format of the *issuer's-distinguished-name-filter* variable is the same as the subject's-distinguished-name-filter variable in the SDNFILTR field. See the SDNFILTR field description for a more complete description of the values that are permitted in the filter parameters.

A maximum of 255 characters can be entered for IDNFILTR.

IDNFILTR is optional if SDNFILTR is specified. If IDNFILTR is not specified, only the subject's name is used as a filter. If IDNFILTR is specified and only a portion of the issuer's name is used as the filter, SDNFILTR must not be specified.

If both IDNFILTER and SDNFILTR are specified, the IDNFILTR value does not need to begin with a valid prefix from the list above. This allows the use of certificates from a certificate authority that chooses to include non-standard data in the issuer's distinguished name.

DSN(*data-set-name*)

When IDNFILTR, SDNFILTR, or both are specified along with DSN on the CERTMAP record, the value in the filter fields must correspond to a starting point within the respective distinguished name found in the certificate contained in the data set. You should specify enough of the name to precisely identify the starting point for the filter. For example, if the certificate in the data set has the following subject:

```
CN=John Jones.OU=Accounts Payable.O=My Company.L=internet
```

and you want all certificates for anyone in:

```
Accounts Payable
```

to be selected by this filter. You must specify:

```
SDNFILTR('OU=Acc')
```

Without the data set containing the certificate, you need to enter the following to produce the same result:

```
SDNFILTR(OU=Accounts Payable.O=My Company.L=internet')
```

When a starting point value is specified for a certificate contained in a data set, there cannot be more than 255 characters between the starting point and ending point of the subject's name in the certificate.

CRITERIA(*criteria-name-template*)

When specified with the MULTIID field, CRITERIA indicates a dynamic user ID mapping. The user ID associated with this mapping profile is based not only on the issuer's distinguished name and the subject's distinguished name found in the certificate, but also on additional criteria. CRITERIA specifies the criteria in the form of one or more variable names, separated by freeform text. These variable names begin with an ampersand (&) and are separated by periods. The freeform text should identify the variables contained in the template:

```
variable-name1=&name1.variable-name2=&name2
```

For example, if the application ID and system ID are to be considered in determining the user ID associated with this mapping, the CRITERIA parameter should be specified as follows:

```
CRITERIA(APPLID=&APPLID.SYSID=&SYSID)
```

The eTrust CA-ACF2-defined criteria are the application ID (APPLID) and the system ID (SYSID). When a user presents a certificate to the system for identification, the identity of the application (as well as the system the user is trying to access) becomes part of the criteria. The application passes its identity to eTrust CA-ACF2 which determines the system identifier. The system ID is the 4-character value specified for the SID parameter of the SMFPRMxx member of SYS1.PARMLIB. This value is substituted for &SYSID in the criteria.

Once the substitution is made, the fully expanded criteria field is used to find a matching profile defined in the CRITMAP GSO records. For example, APPLID=BANKU.SYSID=SYSA is the definition if the application being accessed is BANKU on the SYSA system. You should create a CRITMAP GSO record for this profile. The logonid to be associated with these certificates must be specified as the USERID. The APPLID and SYSID values of the CRITMAP record can be masked so that a record containing APPLID(BANKU) SYSID(*) allows the certificates to be used on any system, rather than just system SYSA. While masking characters can be used in the CRITMAP parameters, they should not be specified in CRITERIA keyword on the CERTMAP GSO record.

Criteria names other than APPLID and SYSID are allowed, but are effective in certificate name filtering if the application supplies these criteria names and their associated values to eTrust CA-ACF2 when the user attempts to access the application using a certificate. SYSID is determined by eTrust CA-ACF2 but APPLID must be specified with the initACEE callable service. Criteria names, such as APPLID and SYSID, should only be specified if the application instructs you to do so.

A maximum of 255 characters can be entered when specifying the CRITERIA keyword. The values can be entered in any case, but are made upper case by eTrust CA-ACF2 because they must match upper case names in the CRITMAP record to be effective. When specifying the criteria value, note that the maximum length of the APPLVAR field on the CRITMAP record is also 254.

The CRITERIA keyword can only be set for MULTIID.

LABEL(*label-name*)

Up to 32 characters can be specified for label-name. It can contain embedded blanks and mixed-case characters. The LABEL field can be used as a record identifier when changing or deleting CERTMAP records. For example, when changing the CERTMAP.INTERNET record, which has a LABEL of A1 to TRUST status, either of the following commands would work.

```
Change certmap.internet trust
Change certmap label(A1) trust
```

TRUST | NOTRUST

When TRUST or NOTRUST is specified it indicates whether this mapping can be used to associate a logonid to a certificate presented by a user accessing the system. If neither is specified, the default is NOTRUST.

USERID(logonid)

Specifies the logonid of the user that is used when a certificate matches this record.

MULTIID | NOMULTIID

MULTIID tells eTrust CA-ACF2 to use the values specified for CRITERIA. If MULTIID is specified, then CRITERIA must be specified. If MULTIID is not specified, CRITERIA is ignored and USERID must be specified. The default is MULTIID.

Example

Example 1 shows entering the filter information manually:

```
SET CONTROL(GSO)
INSERT CERTMAP.lvl1 SDNFILTR(OU=LVL1.OU=SUPPORT.OU=ACF2.O=CA, Inc) USER(tech01)

SYSA     /    CERTMAP.LVL1 LAST CHANGED BY SSDRCM ON 02/15/00 - 20:04
        SDNFILTR('OU=LVL1.OU=SUPPORT.OU=ACF2.O=CA, Inc') USER(TECH01) NOTRUST
```

Example 2 shows entering the same information, but using the DSN field with the filter showing the starting point:

```
SET CONTROL(GSO)
INSERT CERTMAP.lvl1 SDNFILTR(OU=LVL1) USERID(tech01) DSN(bobs.cert)

SYSA     /    CERTMAP.LVL1 LAST CHANGED BY SSDRCM ON 02/15/00 - 20:04
        SDNFILTR('OU=LVL1.OU=SUPPORT.OU=ACF2.O=CA, Inc') USERID(TECH01) NOTRUST
```

SHOW CERTMAP

Displays information contained in CERTMAP records as laid out in the internal CERTMAP table. The display first shows data from records that contain both IDNFILTER and SDNFILTER, followed by records with only SDNFILTER, and finally records with only IDNFILTER.

```
show certmap

 -- CERTMAP FILTERING TABLES --

 IDN/SDN FILTERS
 ---------------
                                                  IDN FILTER
                                                  SDN FILTER
 Label                             TRUST   USER   CRITERIA
 ================================= ===== ======== ==============================
 ACF2 DEVELOPMENT                    Y   ACF2DEVL CN=CAI CERT AUTHORITY.OU=ACF2 D
                                                  EVELOPMENT.O=COMPUTER ASSOCIATE
                                                  S.L=LISLE.ST=ILLINOIS.C=US

                                                  OU=ACF2.OU=DEVELOPMENT.OU=COMPU
                                                  TER ASSOCIATES.L=LISLE.ST=ILLIN
                                                  OIS.C=US


 JASMINE II DEVELOPMENT              Y   JASMINE  CN=CAI CERT AUTHORITY.OU=ACF2 D
                                                  EVELOPMENT.O=COMPUTER ASSOCIATE
                                                  S.L=LISLE.ST=ILLINOIS.C=US

                                                  OU=JASMINE II.OU=DEVELOPMENT.OU
                                                  =COMPUTER ASSOCIATES.L=ISLANDIA
                                                  .ST=NEW YORK.C=US
```

```
UNICENTER TNG DEVELOPMENT          Y   TNG      CN=CAI CERT AUTHORITY.OU=ACF2 D
                                                EVELOPMENT.O=COMPUTER ASSOCIATE
                                                S.L=LISLE.ST=ILLINOIS.C=US

                                                OU=UNICENTER TNG.OU=DEVELOPMENT
                                                .OU=COMPUTER ASSOCIATES.L=ISLAN
                                                DIA.ST=NEW YORK.C=US


TOP SECRET DEVELOPMENT             Y   TSSDEVL  CN=CAI CERT AUTHORITY.OU=ACF2 D
                                                EVELOPMENT.O=COMPUTER ASSOCIATE
                                                S.L=LISLE.ST=ILLINOIS.C=US

                                                OU=TOP SECRET.OU=DEVELOPMENT.OU
                                                =COMPUTER ASSOCIATES.L=PRINCETO
                                                N.ST=NEW JERSEY.C=US


 SDN FILTERS
-----------

                                                SDN FILTER
Label                          TRUST   USER     CRITERIA
============================== ===== ======== ==============================
ISLANDIA DEVELOPMENT               Y   DEVL     OU=DEVELOPMENT.OU=COMPUTER ASSO
                                                CIATES.L=ISLANDIA.ST=NEW YORK.C
                                                =US


LISLE DEVELOPMENT                  N   DEVL     OU=DEVELOPMENT.OU=COMPUTER ASSO
                                                CIATES.L=LISLE.ST=ILLINOIS.C=US


DALLAS DEVELOPMENT                 Y   DEVL     OU=DEVELOPMENT.OU=COMPUTER ASSO
                                                CIATES.L=DALLAS.ST=TEXAS.C=US


IDN FILTERS
-----------

                                                IDN FILTER
Label                          TRUST   USER     CRITERIA
============================== ===== ======== ==============================
PRIVATE USERS                      Y            OU=CLASS 2 PUBLIC PRIMARY CERTI
                                                FICATION AUTHORITY.O=VERISIGN,
                                                INC.C=US

                                                APPLID=&APPLID.COMPANY=&COMPANY

PUBLIC USERS                       Y            OU=CLASS 1 PUBLIC PRIMARY CERTI
                                                FICATION AUTHORITY.O=VERISIGN,
                                                INC.C=US

                                                APPLID=&APPLID
```

CERTMAP GSO Records

The CERTMAP record defines the IDN or SDN filters used to assign a specific logonid to a group of certificates. There is a SHOW CERTMAP command that displays the information contained in the CERTMAP record. See R_cacheserv Cache Names (CACHESRV).

Examples

Example 1 shows entering the filter information manually:

```
SET CONTROL(GSO)
INSERT CERTMAP.lvl1 SDNFILTR(OU=LVL1.OU=SUPPORT.OU=ACF2.O=CA, Inc) USER(tech01)

SYSA     /    CERTMAP.LVL1 LAST CHANGED BY SSDRCM ON 02/15/02 - 20:04
        SDNFILTR('OU=LVL1.OU=SUPPORT.OU=ACF2.O=CA, Inc') USER(TECH01) NOTRUST
```

Example 2 shows entering the same information, but using the DSN field with the filter showing the starting point:

```
SET CONTROL(GSO)
INSERT CERTMAP.lvl1 SDNFILTR(OU=LVL1) USERID(tech01) DSN(bobs.cert)

SYSA     /    CERTMAP.LVL1 LAST CHANGED BY SSDRCM ON 02/15/02 - 20:04
        SDNFILTR('OU=LVL1.OU=SUPPORT.OU=ACF2.O=CA, Inc') USERID(TECH01) NOTRUST
```

Certificate Name Filtering Criteria Mapping (CRITMAP)

Digital certificates can be mapped to one of a number of eTrust CA-ACF2 logonids based on the system ID, application ID, or application-defined variables specified in the CRITMAP record. These records are used in conjunction with the CRITERIA parameter of the CERTMAP GSO records.

Record ID	Fields
CRITMAP.*recid*	APPLID(*application-name*) SYSTEMID(*sysid*) APPLVAR(*site-variable-list*) USERID(*userid-to-map-to*)

Field Descriptions:

recid
Specifies the unique one to eight-character name of the record ID.

APPLID(*application-name*)
Specifies an application ID specified by the application.

SYSTEMID(*sysid*)
Specifies a system ID. The system ID is obtained from the 4-character SID parameter of the SMFPRMxx member of SYS1.PARMLIB.

APPLVAR(*site-variable-list*)
Specifies a list of application-defined variables. For example, (BOBSAPPL=BANKAPP.LEVEL=HIGH). The application-defined variables are passed to eTrust CA-ACF2 on the initACEE callable service along with the certificate identifying the user. APPLVAR has a maximum length of 254.

USERID(*userid-to-map-to*)
Specifies the eTrust CA-ACF2 logonid assigned when matching this CRITMAP record.

Example 1

We want all users whose certificate contains an issuer's distinguished name ending in L=Internet to have their logonid selected based on the application that they are accessing. Users accessing the WEBBANK application will be assigned the WEBBANK logonid. Users accessing the WEBINS application will be assigned the INSUREU logonid.

```
Insert CERTMAP.APPLS IDNFILTR(L=Internet) CRITERIA(APPLID=&APPLID) TRUST

Insert CRITMAP.BANK APPLID(WEBBANK) USERID(WEBBANK)
Insert CRITMAP.INS  APPLID(WEBINS)  USERID(INSUREU)
```

Example 2

We want all users whose certificate contains an issuer's distinguished name ending in L=Internet to have their logonid selected based on the current SYSID and application variable called LEVEL. If the application specified that LEVEL=HIGH and it is running on system A, we'll assign logonid TOPDOG. If the application specified that LEVEL=LOW on system A, we'll assign logonid WORKER.

If the request is running on system B, all users should be assigned the logonid named GENERAL.

```
Insert CERTMAP.APPL2 IDNFILTR(L=Internet) CRITERIA(SYSID=&SYSID.LEVEL=&LEVEL)
TRUST

Insert CRITMAP.SYSA1 SYSID(SYSA) APPLVAR(LEVEL=HIGH) USER(TOPDOG)
Insert CRITMAP.SYSA2 SYSID(SYSA) APPLVAR(LEVEL=LOW) USER(WORKER)
Insert CRITMAP.SYSB  SYSID(SYSB) APPLVAR(LEVEL=*) USER(GENERAL)
```

Show CRITMAP

Displays information contained in CRITMAP records as laid out in the internal CRITMAP table. The display shows the record id, SYSID, APPLID, USERID, and associated application variables.

```
sho critmap

 -- CRITERIA TABLE --

 Record key        SYSTEMID  APPLID   USERID   APPLICATION VARS
 ================= ======== ======== ======== ================================
 CRITMAP.PUBLIC2   *        CICSAPPL PUBLIC2

 CRITMAP.PLATINUM  *        HRAPPL   PLATUSER COMPANY=PLATINUM

 CRITMAP.UCCEL     *        HRAPPL   UCCUSER  COMPANY=UCCEL

 CRITMAP.PUBLIC1   *        WEBAPPL  PUBLIC1
```

RACF to eTrust CA-ACF2 Translation

RACF uses the RACDCERT command to administer digital certificates. This command allows authorized users to add, list, modify, generate and delete certificates. It can also be used to generate certificate requests. The following are examples of RACDCERT commands and their translation to eTrust CA-ACF2 commands.

RACDCERT Command

Example 1

The following RACDCERT command adds a digital certificate to the RACF database:

```
RACDCERT ID(JSMITH) ADD('JSMITH.CLIENT1.P12') WITHLABEL('Certificate for Jimmy
Smith') TRUST PASSWORD('Jacksonville')
```

In eTrust CA-ACF2 you would add the same digital certificate in the following manner:

```
Set profile(user) div(certdata)
Insert jsmith.client dsn('jsmith.client1.p12') label(Certificate for Jimmy Smith)
trust
Password(Jacksonville)
```

Example 2

The following RACDCERT command displays the contents of the certificate in the JSMITH.CLIENT1.P12 data set.

```
RACDCERT CHECKCERT('JSMITH.CLIENT1.P12') PASSWORD('Jacksonville')
```

In eTrust CA-ACF2 you would display the same digital certificate in the following manner:

```
Chkcert dsn('jsmith.client1.p12') password(Jacksonville)
```

Example 3

The following RACDCERT command displays the certificate that was added to the RACF data base:

```
RACDCERT ID(JSMITH) LIST(LABEL('Certificate for Jimmy Smith'))
```

In eTrust CA-ACF2 you would display the CERTDATA record in the following manner:

```
Set profile(user) div(certdata)
List jsmith.client
```

The contents of the certificate contained in the CERTDATA record can be displayed as follows:

```
Chkcert jsmith.client
```

Example 4

The following RACDCERT command modifies the certificate that was added to the RACF data base:

```
RACDCERT ID(JSMITH) ALTER(LABEL('Certificate for Jimmy Smith'))
NEWLABEL('Jimmy Smith Certificate') TRUST
```

In eTrust CA-ACF2 the same change would be made in the following manner:

```
Set profile(user) div(certdata)
Change jsmith.client newlabel(Jimmy Smith Certificate) trust
```

Example 5

The following RACDCERT command deletes the certificate that was added to the RACF data base:

```
RACDCERT ID(JSMITH) DELETE(LABEL('Jimmy Smith Certificate'))
```

In eTrust CA-ACF2 the record would be deleted in the following manner:

```
Set profile(user) div(certdata)
Delete jsmith.client
```

or

```
delete jsmith label(Jimmy Smith Certificate)
```

Example 6

The following RACDCERT command generates a self-signed CA certificate to be added to the RACF database:

```
RACDCERT CERTAUTH GENCERT SUBJECTSDN(CN('My Company z/OS CA') o('My Company')
ou('My Dept') l('My location') sp('Illinois') c('US')) size(512) withlabel('CA
certificate for My Company') NOTBEFORE(DATE(2001-09-04))
NOTAFTER(DATE(2005-10-12))
```

In eTrust CA-ACF2, the following command will generate a self-signed CA certificate that is added to the CA-ACF2 database:

```
Gencert certauth.cert1 subjsdn(cn='My Company z/OS CA' o='My Company' ou='My
Dept' l='My location' sp=Illinois c=US) size(512) label(CA certificate for My
Company) active(2001-09-04) expire(2005-10-12)
```

Example 7

The following RACDCERT command generates a SITE certificate signed by the CA certificate generated in the previous step:

```
RACDCERT SITECERT GENCERT SUBJECTSDN(CN('server.my.company.com') o('My Company')
ou('My Dept') l('My location') sp('Illinois') c('US')) size(512)
withlabel('Certificate for my server') SIGNWITH(CERTAUTH LABEL('CA certificate
for My Company'))
```

In eTrust CA-ACF2, the following command will generate the same SITE certificate:

```
Gencert sitecert.server subjsdn(cn='server.my.company.com' o='My Company' ou='My
Dept' l='My location' sp=Illinois c=US) size(512) label(Certificate for my
server) signwith(certauth label(CA certificate for My Company)
```

Example 8

The following RACDCERT command generates a client certificate signed by an internal CA certificate:

```
RACDCERT ID(JSMITH) GENCERT SUBJECTSDN(CN('Jimmy Smith') o('My Company') ou('My
Dept') l('My location') sp('Illinois') c('US')) size(512) withlabel('Jimmy Smith
Cert') SIGNWITH(CERTAUTH LABEL('CA certificate for My Company'))
```

In eTrust CA-ACF2, the following command will generate the same client certificate:

```
Gencert jsmith.cert subjsdn(cn='Jimmy smith' o='My Company' ou='My Dept' l='My
location' sp=Illinois c=US) size(512) label(Jimmy Smith Cert) signwith(certauth
label(CA certificate for My Company))
```

Example 9

The following RACDCERT command generates a certificate request.

```
RACDCERT ID(JSMITH) GENREQ(LABEL('Jimmy Smith Cert')) DSN('JSMITH.CERT.REQ')
```

In eTrust CA-ACF2, the following command will generate the same certificate request:

```
Genreq jsmith.cert dsn('jsmith.cert.req')
```

Example 10

The following RACDCERT command generates a certificate from a certificate request.

```
RACDCERT ID(JSMITH) GENCERT('JSMITH.CERT.REQ') WITHLABEL('Jimmy Smith Cert from
req') SIGNWITH(CERTAUTH LABEL('CA certificate for My Company'))
```

In eTrust CA-ACF2, the following command will generate the same certificate from a certificate request:

```
Gencert jsmith.cert2 dsn('jsmith.cert.req') label(Jimmy Smith Cert from req)
signwith(certauth.cert1)
```

Example 11

The following RACDCERT command exports a certificate and private key to a dataset in DER format.

```
RACDCERT ID(JSMITH) EXPORT(LABEL('Jimmy Smith Cert')) DSN('JSMITH.CERT.P12')
FORMAT(PKCS12DER) PASSWORD('JimmySmith')
```

In eTrust CA-ACF2, the following command will export the same certificate to the same dataset:

```
export jsmith label(Jimmy Smith Cert) dsn('jsmith.cert.p12') format(pkcs12der)
password(Jimmy Smith)
```

Example 12

The following RACDCERT command creates a key ring.

```
RACDCERT ID(WEBSRV) ADDRING(SERVA)
```

In eTrust CA-ACF2, the following command creates the same key ring:

```
Set profile(user) div(keyring)
Insert websrv ringname(SERVA)
```

Example 13

The following RACDCERT command connects a certificate to the key ring

```
RACDCERT ID(WEBSRV) CONNECT(ID(WEBSRV) LABEL('Certificate for serva') RING(SERVA)
DEFAULT)
```

In eTrust CA-ACF2, the following command connects the same certificate to the key ring:

```
Set profile(user) div(keyring)
Connect certdata(websrv) label(Certificate for serva) keyring(websrv)
ringname(SERVA) default
```

Alternatively you could enter:

```
Set profile(user) div(keyring)
Connect certdata(websrv.serva) keyring(websrv.serva) default
```

Example 14

The following RACDCERT command lists a key ring.

```
RACDCERT ID(WEBSRV) LISTRING(SERVA)
```

In eTrust CA-ACF2, either of the following commands will list the key ring:

```
Set profile(user) div(keyring)
list websrv ringname(SERVA)
list websrv.serva
```

Example 15

The following RACDCERT command removes a certificate from a key ring

```
RACDCERT ID(WEBSRV) REMOVE(ID(WEBSRV) LABEL('Certificate for serva') RING(SERVA))
```

In eTrust CA-ACF2, the following command removes the same certificate from the key ring:

```
Set profile(user) div(keyring)
remove certdata(websrv) label(Certificate for serva) keyring(websrv)
ringname(SERVA)
```

Example 16

The following RACDCERT command deletes a key ring.

```
RACDCERT ID(WEBSRV) DELRING(SERVA)
```

In eTrust CA-ACF2, the following command deletes the same key ring:

```
Set profile(user) div(keyring)
delete websrv ringname(SERVA)
```

Example 17

The following RACDCERT command adds a mapping profile that will associate the user ID WEBUSER with all digital certificates issued by VeriSign for Class 1 Individual Subscribers:

```
RACDCERT ID(WEBUSER) MAP IDNFILTER('OU=VeriSign Class 1 Individual
Subscriber.O=VeriSign, Inc..L=Internet') WITHLABEL('Savings Account')
```

In eTrust CA-ACF2, the following command creates the same profile:

```
Set control(gso) div(certmap)
Insert certmap.webuser idnfiltr(OU=VeriSign Class 1 Individual
Subscriber.O=VeriSign, Inc..L=Internet) label(Savings Account) USERID(webuser)
```

Using Certificates Signed by a Certificate Authority

When using certificates signed by a Certificate Authority (such as Verisign or Thawte) the Certificate Authority's root certificate must be obtained and inserted into CA-ACF2 as a CERTAUTH CERTDATA record. Typically this certificate can be obtained from the Certificate Authority's web site. The following steps can be used as a guideline for defining the Certificate Authority's certificate to CA-ACF2.

1. Determine the format (binary or text) of the Certificate Authority's certificate. Most Certificate Authority web sites provide the certificate in both formats.

   Text (PEM) format certificates begin with the line:

   ```
   ----BEGIN CERTIFICATE ----
   ```

   The DER encoded certificate data follows this line. Following the certificate data should be a line containing:

   ```
   ----END CERTIFICATE ----
   ```

   Binary format certificates are typically downloaded to your PC by clicking on a download icon from the Certificate Authority's web site. The file extension of the downloaded file on your PC will be "cer" for binary format certificates.

2. Transer the certificate to a mainframe dataset.

   If the certificate is in PEM format, cut and paste the entire certificate, (including the "- - - -BEGIN CERTIFICATE- - - -" and - - - -END CERTIFICATE - - - -" lines) into a mainframe dataset.

   If the certificate is in binary format, use FTP to transfer the certificate from your PC to a mainframe dataset.

3. Run CHKCERT against the mainframe data set:

   ```
   ACF
   CHKCERT DSN('TEST.CERTAUTH.PEM')
   ```

   Verify that the CHKCERT runs correctly and that the issuer, subject, and validity dates are correct.

4. Insert the CERTDATA record:

   ```
   ACF
   T PROFILE(USER) DIV(CERTDATA)
   INSERT CERTAUTH.recid DSN('TEST.CERTAUTH.PEM')
   ```

5. Change the CERTDATA record to TRUST status, if necessary.

   ```
   ACF
   T PROFILE(USER) DIV(CERTDATA)
   CHANGE CERTAUTH.recid TRUST
   ```

6. Connect the CERTDATA record to any keyrings that require it.

   ```
   ACF
   CONNECT CERTDATA(CERTAUTH.recid) KEYRING(userid.suffix) USAGE(CERTAUTH)
   ```

Appendix

A

Operator Identification Card Support

Operator identification (OID) card support provides users of IBM 3270 terminals containing OID card readers with an additional level of security beyond that provided by passwords. When a user logs on to TSO, he is prompted for his password, as usual. If he enters the correct password, he is prompted for an OID card. If he enters the correct OID card, he can log on.

An OID card is read by an operator identification card reader (OICR) or magnetic slot reader (MSR). The term OID card reader is used throughout this appendix to refer to these devices. These devices attach to a 3270-series terminal, and read a card the size of a credit card, with a magnetic strip on the back. Credit cards can be used as OID cards.

You can select the use of OID cards on a user-by-user basis. You can define multiple OID cards for a user. The user can enter any one of them or, for high security applications, he might be required to enter all of them.

A logonid that requires an OID card can log on only at a terminal that has an OID card reader or magnetic slot reader. If this were not the case, the OID requirement could be bypassed simply by going to a terminal that did not have an OID card reader.

Implementing OID Card Support

Operator identification card support is a standard feature of eTrust CA-ACF2. To use it, perform the following tasks:

- Select an attribute in the logonid record to indicate that a logonid requires an OID card
- Insert a GSO APPLDEF record to define the structure of the OID infostorage records
- Insert a GSO AUTHEXIT record to define the OID support exit to eTrust CA-ACF2
- eTrust CA-ACF2 supplies a support exit program in load module form, ACFEADID, for this purpose.

- Insert an identity infostorage record to define valid OID cards for each logonid that requires an OID card
- Set the appropriate attribute in the logonid record for each logonid that requires an OID card.

Choosing a Logonid Record Attribute

Choose an attribute in the logonid record to indicate that a user requires an OID card. eTrust CA-ACF2 provides eight attributes in the logonid record for use by OID card support and extended user authentication support: AUTHSUP1 through AUTHSUP8.

The examples in this appendix assume that you selected AUTHSUP1 and that the external @CFDE name for AUTHSUP1 in the ACFCFDE member of CAI.CAIMAC has been renamed to OID.

Inserting the GSO APPLDEF Record

The GSO APPLDEF record defines the structure of the identity infostorage records used by OID support. Use the ACF command to insert GSO APPLDEF records. See the "Maintaining Global System Options Records" chapter for details on how to issue the INSERT subcommand.

The following commands insert a sample GSO APPLDEF record to define infostorage records for OID support:

```
set control(gso)
 CONTROL
insert appldef.oid class(i|identity) -
                   type(aut|authsup) -
                   appldiv(oid) -
                   appldlen(8) -
                   dftdrtn(acf00dft) -
                   recid(acfoirsb|) -
                   recidlen(8) -
                   selauth(account)
```

APPLDEF.OID
The name of the record. This example appends .OID to differentiate this record from other APPLDEF records.

CLASS(i|identity)
The class of this infostorage record.

TYPE(aut|authsup)
The type code of this record.

APPLDIV(oid)
The division name of this record.

APPLDLEN(8)
The length of the division record.

DFTDRTN(acf00dft)
The default division routine if DIVISION is not specified when a user accesses this record.

RECID(acfoirsb)
The name of the record structure block (RSB) that defines the format of this structured infostorage record.

RECIDLEN(8)
The maximum number of characters for the record ID.

SELAUTH(account)
The logonid attributes a user must have to insert or modify these records. For example, users with the ACCOUNT privilege can modify this record.

Inserting the GSO AUTHEXIT Record

Use the GSO AUTHEXIT record to define the logonid attribute that indicates that a user must use an OID card to log on to the system, and the name of the exit that processes the OID cards. Use the ACF command to insert GSO AUTHEXIT records. The general syntax for doing this is described in the "Maintaining Global System Options Records" chapter.

The following commands insert a sample GSO AUTHEXIT record to define the logonid attribute and exit program name:

```
set control(gso)
 CONTROL
insert authexit.oid lidfield(oid) procpgm(acfeaoid)
```

AUTHEXIT.OID
The name chosen for this GSO AUTHEXIT sample record.

LIDFIELD(oid)
The logonid attribute chosen for this example (oid) indicates that an OID card is required. A user with the OID attribute in his logonid must use an OID reader to access the system.

PROCPGM(acfeaoid)
The name (acfeaoid) of the extended user authentication exit program chosen for this example.

Inserting the Identity Infostorage Records

You must create an IDENTITY(AUT) infostorage record that defines the valid OID cards for that logonid for each user who requires an OID card. If you do not define a record, the user cannot log on. Use the ACF command to insert IDENTITY(AUT) infostorage records. The general syntax for doing this is described in "Maintaining Global System Options Records."

The following commands insert a sample IDENTITY(AUT) record for a user:

```
set identity(aut) division(oid)
 IDENTITY
insert logonid oidcard(oid-data)
```

In the example, *logonid* is the logonid of the user whose record you are inserting. *OID-data* is the data that must be read from the card. You can enter the OID-data as shown in the following:

```
INSERT logonid OIDCARD(401322349999880)
```

If the OID card data contains blanks or special characters (characters other than alphabetic, numeric, or national characters), you must enclose the data in single quotes:

```
INSERT logonid OIDCARD('372811928499999 8806860799999')
```

If the OID card data contains single quotes, you must double them. For example, if the OID card data was 3332'3333'9999'8806, you should enter it as follows:

```
INSERT logonid OIDCARD('3332''3333''9999''8806')
```

Here is another way to create identity records, without typing in the OID card data:

```
INSERT logonid OIDCARD
```

The ACF command issues a prompt for the OID card data:

```
ACF61500: ENTER OIDCARD
```

At this point, run the OID card through the reader. The ACF command records the data from the card in the identity record.

Note: If the data on the OID card contains special characters (including spaces), you cannot use this method. You must enter the OID card data.

You can define multiple OID cards for a user. Define the first OID card as shown in the earlier examples. The second and subsequent ones are added as follows:

```
CHANGE logonid ADD OIDCARD(oid-data)
```

You can define up to 32 OID cards for a logonid.

To require a user to enter all of his OID cards when he logs on, specify OID-ALL on the identity record:

```
CHANGE logonid OID-ALL
```

Setting the Logonid Record Attribute

You must set an attribute in the logonid record for each logonid that requires an OID card. Use the ACF command to set the attribute. The general syntax for doing this is described in the chapter, "Maintaining Logonid Records."

The following commands set the attribute to require a logonid to log on with an OID card:

```
CHANGE logonid OID
```

The OID attribute indicates that an OID card is required. It must be the same attribute specified in the LIDFIELD field of the GSO AUTHEXIT record.

Appendix

B IBM-Supplied Resource Classes

z/OS SAF

APPCLU
Check the identity of logical units when establishing a VTAM session.

APPCSERV
Check a program run by a user to determine if it can act as a server for a specific APPC transaction program (TP).
APPL—Control access to applications.

CONSOLE
Control access to MCS consoles. Check the ability of commands issued from the MCS console to access other resources.

CSFKEYS
Control use of Integrated Cryptographics Service Facility (ICSF) cryptographic keys. See also GCSFKEYS.
CSFSERV—Control use of Integrated Cryptographics Service Facility (ICSF) cryptographic services.

DASDVOL
Check access to DASD volumes.

DATASET
Checks access to data sets.
DEVICES—Check a user's ability to allocate devices such as:

- Unit record devices that can be allocated only by PSF, JES2, or JES3
- Graphic devices that can be allocated only by VTAM
- Teleprocessing and communication devices that can be allocated only by VTAM.

DIRAUTH
Check information regarding label dominance.

DLFCLASS
Check accesses by Data Lookaside Facility.

DSNR
Check issued by DB2 security.

FACILITY
Check for various uses. For example, DFP issues REQUEST=AUTH,CLASS=FACILITY calls when a user attempts to perform catalog operations.

FIELD
Check field-level accesses.

GCSFKEYS
Check access to Integrated Cryptographics Service Facility (ICSF) cryptographic keys. This is the RACF resource group class for CSFKEYS class.

GDASDVOL
Check access to DASD volumes. This is the RACF resource group class for DASDVOL class.

GLOBAL
Check global access to a table entry.

GMBR
Check member class for GLOBAL class.

GSDSF
Check access by SDSF. This is the RACF resource group class for SDSF class.

GTERMINL
Check access by terminals. This is the RACF resource group class for TERMINAL class.

JESINPUT
Check access of commands or jobs entered through a JES input device.

JESJOBS
Check ability to submit and cancel jobs by job name.

JESSPOOL
Check access to job data sets in the JES spool (SYSIN and SYSOUT data sets).

NODES
Check the following:

- Can jobs enter the system from other nodes?
- Must jobs that enter the system from other nodes go through system entry validation (supply logonid and password)?

NODMBR
Check member class for NODES class.

NVASAPDT
Check access by NETVIEW/Access Services.

OPERCMDS
Check ability to issue operator commands (JES and z/OS commands).

PMBR
Check member class for PROGRAM class.

PROGRAM
Check access by programs or load modules.

PROPCNTL
Check ability of userids to be propagated to other subsystems.

PSFMPL
Check by PSF for page labeling.

RACFVARS
Check for ability to use RACF variables.

RVARSMBR
Check for member class of RACFVARS class.

SCDMBR
Check for member class for SECLABEL class.

SDSF
Check for ability to issue SDSF commands.

SECDATA
Check for security classifications of users and data.

SECLABEL
Check ability to use security labels.

SERVAUTH
Check access by servers.

SMESSAGE
Check ability to send TSO messages.

SURROGAT
Check if jobs can be submitted by surrogates and which users can act as surrogates.

TAPEVOL
Check access to data on tape volumes.

TEMPDSN
Check access to residual temporary data sets.

TERMINAL
Check access by terminals.

VTAMAPPL
Check ability to open ACBs from non-APF authorized programs.

WRITER
Check ability to use JES writers.

CICS SAF

TCICSTRN
Check access to CICS transactions.

GCICSTRN
Check access to CICS transactions. This is the RACF resource group class for TCICSTRN class.

PCICSPSB
Check access to CICS program specification blocks.

QCICSPSB
Check access to CICS program specification blocks. This is the RACF resource group class for PCICSPSB class.

FCICSFCT
Check access to CICS file control table.

HCICSFCT
Check access to CICS file control table. This is the RACF resource group class for HCICSFCT class.

JCICSJCT
Check access to CICS journal control table.

KCICSJCT
Check access to CICS journal control table. This is the RACF resource group class for HCICSFCT class.

DCICSDCT
Check access to CICS destination control table.

ECICSDCT
Check access to CICS destination control table. This is the RACF resource group class for DCICSDCT class.

SCICSTST
Check access to CICS temporary storage table.

UCICSTST
Check access to CICS temporary storage table. This is the RACF resource group class for SCICSTST class.

MCICSPPT
Check access to CICS processing program table.

NCICSPPT
Check access to CICS processing program table. This is the RACF resource group class for MCICSPPT class.

ACICSPCT
Check access to CICS program control table.

BCICSPCT
Check access to CICS program control table. This is the RACF resource group class for ACICSPCT class.

CCICSCMD
Check ability of a user to enter CICS systems programmer commands, such as INQUIRE, SET, PERFORM, and COLLECT.

VCICSCMD
Check ability of a user to enter CICS systems programmer commands, such as INQUIRE, SET, PERFORM, and COLLECT. This is the RACF resource group class for CCICSCMD class.

DFSMS SAF

MGMTCLAS
Check ability to use an SMS management class.

STORCLAS
Check ability to use an SMS storage class.

IMS SAF

AIMS
Check ability to use IMS/VS application group names.

CIMS
Check member class for DIMS.

DIMS
Check ability to use resource group class CIMS.

FIMS
Check member class for HIM.

GIMS
Check ability to use resource group class TIMS.

HIMS
Check ability to use resource group class WIMS.

OIMS
Check member class for WIMS.

PIMS
Check member class for UIMS.

QIMS
Check ability to use IMS support. Member class is PIMS.

SIMS
Check member class for UIMS.

TIMS
Check ability to issue IMS/VS transactions.

UIMS
Check ability to use IMS support. Member class is SIMS.

WIMS
Check ability to use resource group class for OIMS.

Information Management SAF

INFOMAN
Check member class for Information Management Version 5.

GINFOMAN
Check member class for Information Management Version 5. This is the resource group class for INFOMAN.

NetView SAF

RMTOPS
Check NetView remote operations.

TSO SAF

ACCTNUM
Check ability to use TSO account numbers.

TSOPROC
Check ability to use TSO logon procedures.

PERFGRP
Check ability to use TSO performance groups.

TSOAUTH
Check ability to use TSO user attributes, such as OPER and MOUNT.

SAF/SAF Product Return/Reason Codes

The following sequences of SAF/SAF Product return/reason codes can be expected for the following general error situations:

Logonid/Profile Not Found Errors

If you receive any of the following eTrust CA-ACF2 error messages:

ACF02010 RECORDS NOT FOUND
ACF02030 RECORDS NOT FOUND
ACF02089 NO SCOPE LIST RECORD FOUND - ACCESS DENIED
ACF02106 PROTOTYPE LID NOT FOUND
ACF0A005 RECORDS NOT FOUND
ACF0A089 NO SCOPE LIST RECORD FOUND - ACCESS DENIED

The following SAF Return/Reason codes can be expected:

SAF return code=4

SAF product return code=8

SAF product reason code=8 (SAF product reason: Segment not found)

Authority/Access Errors

If you receive any of the following eTrust CA-ACF2 error messages:

ACF00042 ACMCB/ACUCB NOT ALLOWED - INVALID AUTHORITY
ACF00103 NOT AUTHORIZED TO CHANGE FIELD fld
ACF00112 SUPER SECURITY PRIVILEGES NEEDED TO CHANGE FIELD fld
ACF00173 UNSCOPED SECURITY PRIVILEGE NEEDED TO CONNECT SITE OR CERTIFICATE
ACF02002 NOT AUTHORIZED FOR REQUEST
ACF02003 NOT AUTHORIZED FOR INSERT
ACF02004 NOT AUTHORIZED FOR DELETE
ACF02005 NOT AUTHORIZED FOR UNFORMATTED RETURN
ACF02007 NOT AUTHORIZED FOR MASK KEY PROCESSING
ACF02009 YOU ARE NOT AUTHORIZED TO ACCESS THIS LOGONID RECORD
ACF02012 NOT AUTHORIZED TO INSERT THIS LOGONID RECORD
ACF02013 NOT AUTHORIZED TO DELETE THIS LOGONID RECORD
ACF02014 NOT AUTHORIZED TO LIST THIS LOGONID RECORD
ACF02017 NOT AUTHORIZED TO ACCESS MODEL RECORD
ACF02020 DELETE/CHANGE/INSERT LID DENIED - LID HAS NO MUSUPDT AUTHORITY
ACF02028 DSC COMPONENT - NOT AUTHORIZED
ACF02031 NOT AUTHORIZED TO USE CONDITIONAL BYPASS PROCESSING
ACF02044 MODIFIED LIDREC HAS MORE AUTHORITY THAN YOU CURRENTLY HAVE
ACF0A017 NOT AUTHORIZED FOR REQUESTED FUNCTION

ACF0A019 THE CONSOLE OPERATOR HAS DENIED YOUR REQUEST
ACF0A020 DELETE/CHANGE/INSERT DENIED - ID HAS NO MUSASS UPDATE
ACF0A022 NOT AUTHORIZED TO CHANGE DSN FIELD
ACF0A028 DATABASE SYNCHRONIZATION COMPONENT - NOT AUTHORIZED
ACF0A031 NOT AUTHORIZED TO USE CONDITIONAL BYPASS PROCESSING
ACF0A205 NOT AUTHORIZED TO ACCESS APPLICATION, SYSID, OR RECORD

The following SAF Return/Reason codes can be expected:

SAF return code=4

SAF product return code=8

SAF product reason code=4 (SAF product reason: Field access check failed)

Recovery (ESTAE) Related Failures

If you receive the following eTrust CA-ACF2 error message:

ACF0A031 NOT AUTHORIZED TO USE CONDITIONAL BYPASS PROCESSING

The following SAF Return/Reason codes can be expected:

SAF return code=4

SAF product return code=4

SAF product reason code=0 (SAF product reason: ESTAE environment could not be established)

Security (eTrust CA-ACF2) Inactive Errors

If you receive any of the following eTrust CA-ACF2 error messages:

ACF00002 ACF2 SYSTEM NOT READY, TRY LATER
ACF02006 ACF2 NOT INITIALIZED, TRY LATER

The following SAF Return/Reason codes can be expected:

SAF return code=4

SAF product return code=12

SAF product reason code=0 (SAF product reason: Security inactive)

Exit-Related Failures

If you receive any of the following eTrust CA-ACF2 error messages:

ACF00402 ACF2 USER EXIT ENVIRONMENTAL ERROR
ACF02035 EXIT exit FAILED REQUEST
ACF02036 INVALID RETURN CODE FOR EXIT exit retc
ACF0A035 exit EXIT FAILED REQUEST
ACF0A036 INVALID RETURN CODE FOR EXIT exit retc

The following SAF Return/Reason codes can be expected:

SAF return code=4

SAF product return code=0

SAF product reason code=0 (SAF product reason: No successful exit processing can take place)

Parameter List-Related Failures

For parameter list-related failures, such as bad plist, or improper data, you might receive any of the following eTrust CA-ACF2 error messages:

ACF00003 INVALID FUNCTION REQUEST
ACF00004 APF SUPERCALL INVALID FOR SUBFUNCTION FUNC
ACF00005 APF SUPERCALL invalid for subfunction func with NAPF
ACF00006 APF SUPERCALL invalid for subfunction func with ACXEX
ACF00040 Control block error - ACUCB logonid does not match AC
ACF00041 ACMCB/ACUCB control block processing error
ACF00042 ACMCB/ACUCB not allowed - invalid authority
ACF00100 No alter request entries
ACF00101 ARES run past buffer end
ACF00109 Invalid operation requested for field fld
ACF00110 ADD/DEL/REPL required for operand list
ACF00111 Negative string length for field fld
ACF00121 Negative or zero length for field fld
ACF00122 STRING too long for field fld
ACF00131 Field length must be 1 for switch field fld
ACF00141 FIELD length must be 4 for binary field fld
ACF00142 Value must be non-negative for binary field fld
ACF00143 Binary 1 and 3 byte fields cannot have negatives
ACF00144 Maximum or minimum value exceeded for field fld
ACF00145 Counter field fld went negative, it was forced to zer
ACF00161 Field length must be 1 for field fld
ACF00163 Field $$ cannot be greater than field $$
ACF00164 Realm and kerbname cannot exceed 240 characters combi
ACF00165 Fields field1 and field2 are mutually exclusive
ACF00166 Flag flag required when field1 is used
ACF00167 Field field1 or field2 is required
ACF00168 value found in fld was not found in the certificate

ACF00169 DEFTKTLF, MAXTKTLF, MINTKTLF must be used together
ACF00171 key is invalid as a line delete keyword
ACF00172 key is invalid as a character delete keyword
ACF00179 Invalid userid specified in record key
ACF00180 Hexidecimal string too long for field fld
ACF00181 Invalid hexidecimal digits for field fld
ACF00182 Password is probably incorrect
ACF00191 For scope validation, field fld must be 8 characters
ACF00193 Invalid value for field fld - must be either PREVENT/LOG/ALLOW
ACF00194 fld attribute cannot be turned on
ACF00201 Entry count exceeds capacity of field fld
ACF00202 Field fld is not eligible for add or del
ACF00203 Entry not found in field fld
ACF00204 Duplicate data not allowed for field fld
ACF00205 Action leaves field fld incompletely specified
ACF00206 Value specified for field fld does not meet min. req'
ACF00207 Value specified for field fld is longer than max allo
ACF00208 Value required for field fld
ACF00209 Field fld not allowed for record
ACF00213 Values specified for field fld are mut exc
ACF00215 Invalid value specified for field fld
ACF00216 Negative values not allowed for fields fld
ACF00219 Invalid racroute value for fld: fld RSN: rsn
ACF00223 Invalid @CFDE length for field fld
ACF00237 Qualifier name, ACF2, is a reserved name
ACF00238 Invalid qualifier name specified
ACF00245 Offset value for fld is larger than the max full LID
ACF02001 Invalid subfunction request
ACF02008 Supplied mask does not contain any asterisks
ACF02016 Conflicting UID and LID masks
ACF02037 Keyword password is required
ACF0A001 Invalid Subfunction Request
ACF0A007 Record type is improper for requested function
ACF0A011 Model record does not match object of insert
ACF0A016 Data too large - update or insert rejected
ACF0A018 Invalid resource or entry name syntax
ACF0A021 Supplied mask does not contain any asterisks
ACF0A030 invalid pseudo-dsname syntax
ACF0A041 Certificate label is a duplicate of certificate record rrrr
ACF0A042 Key ring names cannot be changed in multiple record requests
ACF0A043 Certificate labels cannot be changed in multiple record requests
ACF0A201 Invalid request - undefined storage class
ACF0A202 Invalid request - undefined storage type
ACF0A203 Invalid request - undefined SYSID
ACF0A204 Invalid request - undefined record ID
ACF0A206 Improper component list supplied for validation
ACF0A207 SYSID or recid length error
ACF0A209 Masking not applied for this request
ACF0A210 Invalid mask supplied for SYSID or record ID

The following SAF Return/Reason codes can be expected:

SAF return code=8

SAF product return code=24

SAF product reason code=20 (SAF product reason: Plist version error)

Database-Full Error Situations

If you receive any of the following eTrust CA-ACF2 error messages:

ACF02105 No room in data base for alter request
ACF02107 No room in data base for insert request
ACF0A025 Insufficient space in data dase to complete request

The following SAF Return/Reason codes can be expected:

SAF return code=4

SAF product return code=16

SAF product reason code=20 (SAF product reason: insufficient room on data base)

Duplicate Record Error Situations

If you receive any of the following eTrust CA-ACF2 error messages:

ACF00176 Duplicate certificate detected - recordid
ACF02108 Logonid lid already exists
ACF0A040 Key ring name is a duplicate of existing key ring
ACF0A047 CERTMAP label is a duplicate of existing CERTMAP record:rrrr
ACF0A049 This is a certificate replacement for existing certificate record:rrrr

The following SAF Return/Reason codes can be expected:

SAF return code=4

SAF product return code=16

SAF product reason code=8 (SAF product reason: Duplicate record failure)

Other Error Situations

For those eTrust CA-ACF2 error messages not previously listed here, the following will be returned:

SAF return code=4

SAF product return code=8

SAF product reason code=0 (SAF product reason: No profile found)

Appendix

C

Protecting Operator Commands

You can protect operator commands using eTrust CA-ACF2. When an operator command is issued, SAF issues a CMDAUTH call to the CA SAF interface. This turns into a RACROUTE REQUEST=AUTH, CLASS=OPERCMDS call. eTrust CA-ACF2 does not process REQUEST=AUTH,CLASS=OPERCMDS calls. You must create a SAFDEF record to define the SAF call, a CLASMAP record to translate the resource class into a type code, and resource rules to validate the use of the operator commands.

Creating a SAFDEF Record

To create a SAFDEF record for the OPERCMDS class, issue these commands:

```
set control(gso)
 GSO
insert safdef.opr id(opercmds) jobname(-) mode(global)-
 racroute(request=auth,class=opercmds) userid(-) rep
```

Creating a CLASMAP Record

This step is optional. If you do not add this CLASMAP record, eTrust CA-ACF2 translates the class to the default, SAF. In the example, we selected OPR.

```
acf
 ACF
set control(gso)
 GSO
insert clasmap.opr resource(opercmds) rsrctype(opr)
```

Adding Resource Types to INFODIR Record

After you create the CLASMAP record, add the OPR type code to the GSO INFODIR record. To update the INFODIR record, enter these commands:

```
acf
 ACF
set control(gso)
 CONTROL
change infodir types(r-ropr)
```

Refreshing GSO Records

These changes do not take effect until you refresh the Infostorage database by issuing the following command:

```
F ACF2,REFRESH(infodir),SYSID(sysid),CLASS(c),TYPE(gso)
```

Writing Resource Rules

Now you need to decide which users can issue the SECTRACE commands. You might write a resource rule like the following:

```
acf
 ACF
set resource(opr)
 RESOURCE
compile * store
 .$key(trce) type(opr)
 . sectrace.- uid(oper-) service(read,update) allow
 .end
```

The sample rule uses the first-level qualifier of the entity TRCE.SECTRACE.- in the $KEY. The second-level qualifier is masked in the rule entry to indicate that we want the rule to apply to all SECTRACE resources. We granted access to UIDs that begin with OPER.

For some operator commands, you might have to specify a SERVICE, such as READ or UPDATE.

To write a rule, you need the name of the resource that you want to secure. In SAF, these names are referred to as entity names. In eTrust CA-ACF2 these names are referred to as resource names. SAF entity names are listed in the appropriate IBM guide. The resource names for eTrust CA-ACF2 operator commands are listed in Resource Names for CA-ACF2 Operator Commands later in this appendix.

Rebuilding Infostorage Directories

Anytime you create a new resource rule, you must rebuild that resource directory; otherwise, the pointers to the new rule are not built and eTrust CA-ACF2 does not recognize that the new rule exists.

To rebuild the infostorage directory, issue the following operator command:

```
F ACF2,REBUILD(opr),CLASS(r)
```

Resource Names for eTrust CA-ACF2 Operator Commands

The following tables list the eTrust CA-ACF2 operator command, the SERVICEs required in the resource rule, and the resource name.

eTrust CA-ACF2 z/OS Operator Commands

The eTrust CA-ACF2 operator commands, START, STOP, and MODIFY have the following resource names: (You cannot specify parameters, such as BACKUP or REFRESH.)

Command/Keyword	SERVICE	Resource Name
MODIFY ACF2	UPDATE	MVS.MODIFY.JOB.ACF2
START ACF2	UPDATE	MVS.START.JOB.ACF2
STOP ACF2	UPDATE	MVS.STOP.JOB.ACF2

CA MAC Operator Commands

The CA MAC operator commands, the required SERVICEs, and their source names are listed in the following.

Command/Keyword	SERVICE	Resource Name
DISPLAY MACS,DEVICE(addr)	READ	MACS.DISPLAY.DEVICE.addr
DISPLAY MACS,NODE(addr)	READ	MACS.DISPLAY.NODE.addr
DISPLAY MACS,OPTIONS	READ	MACS.DISPLAY.OPTIONS
MODIFY MACS,BACKUP(vol)	UPDATE	MACS.MODIFY.BACKUP.vol
MODIFY MACS,CLOSE(vol)	UPDATE	MACS.MODIFY.CLOSE.vol

Command/Keyword	SERVICE	Resource Name
MODIFY MACS,OPEN(vol)	UPDATE	MACS.MODIFY.OPEN.vol
MODIFY MACS,REBUILD(vol)	UPDATE	MACS.MODIFY.REBUILD.vol
MODIFY MACS,RESTORE(vol)	UPDATE	MACS.MODIFY.RESTORE.vol
STOP MACS	UPDATE	MACS.STOP
VARY MACS,DEVICE(addr)	UPDATE	MACS.VARY.DEVICE.addr
VARY MACS,NODE(addr)	UPDATE	MACS.VARY.NODE.addr

SECTRACE Operator Commands

The SECTRACE operator commands, the required SERVICEs, and their source names are listed in the following:

Command/Keyword	SERVICE	Resource Name
SECTRACE DELETE	UPDATE	TRCE.SECTRACE.DELETE
SECTRACE DISABLE	UPDATE	TRCE.SECTRACE.DISABLE
SECTRACE DISPLAY	READ	TRCE.SECTRACE.DISPLAY
SECTRACE LOGERR	UPDATE	TRCE.SECTRACE.LOGERR
SECTRACE MODIFY	UPDATE	TRCE.SECTRACE.MODIFIED
SECTRACE NOLOGERR	UPDATE	TRCE.SECTRACE.NOLOGERR
SECTRACE SET	UPDATE	TRCE.SECTRACE.SET

Translating SAF Levels to eTrust CA-ACF2 SERVICE Keywords

IBM documents the SAF access authority level for each of the MVS console commands. These authorities have their eTrust CA-ACF2 counterparts. You must use the SERVICE parameter and the appropriate keywords to properly define the validation that you want eTrust CA-ACF2 to perform.

The following table lists the SAF authority levels and the eTrust CA-ACF2 SERVICE keyword:

SAF Authority Level	eTrust CA-ACF2 SERVICE Keyword
ALTER	ADD
CONTROL	DELETE
READ	READ
UPDATE	UPDATE

Suppose you want to write a rule to protect the following command:

```
MVS CANCEL jobname
```

The authority level is UPDATE and the SAF resource name is MVS.CANCEL.JOB.jobname. For STCs, the resource name is MVS.CANCEL.STC.jobname. Here is how you might write a resource rule to validate the authority to issue these commands:

```
set resource(opr)
 RESOURCE
compile * store
 .$key(mvs) type(opr)
 . cancel.job.- uid(oper) service(update) allow
 . cancel.stc.- uid(oper) service(update) allow
 .
end
```

This rule lets all UIDs that begin with OPER issue the z/OS CANCEL jobname command.

If you enter the SERVICE as READ, eTrust CA-ACF2 prevents the OPER logonids from issuing the commands.

Appendix

D

Implementing Member-Level Protection

eTrust CA-ACF2 provides the ability to protect partitioned data sets (PDS) at the member level. In the past, access to PDS data could be allowed or denied only at the data set level; that is, a user with read access to a PDS could read all members of a PDS, and a user with write access could update all members of a PDS.

With eTrust CA-ACF2 member-level protection, the security administrator can restrict which members of a PDS a user can access. Therefore, a user with read access to a member-level protected PDS library is only allowed to see those members for which PDS resource rules have been defined. A user with write access to a member-level protected library is restricted to which members can be updated and also which members can be created and deleted.

The first step in implementing member-level protection is determining which PDS libraries require member-level protection. We do not recommend protecting individual user PDS libraries. Select the system and application libraries for which member-level protection is desired.

These libraries are identified to eTrust CA-ACF2 through the use of the GSO PDS record. The PDS record also identifies the resource rules against which member-level protection is performed through specification of the resource rule type code. These rules must be globally resident.

The resource rule key is the member accessed. There is also a special resource name <DIR>, which represents a direct access to the PDS directory. Most applications do not access the PDS directory directly. However, a PDS can be destroyed if a user inadvertently opens the PDS as a sequential data set. Use of a <DIR> resource rule for a member-level protected library can prevent that occurrence.

The resource rule SERVICE keyword can be used to identify the type of access being made to the PDS member as follows:

READ
The member is being read.

UPDATE
The member is being updated, created, or deleted.

Implementation Steps

The following steps explain how to implement PDS member-level protection:

1. Determine the library that requires member-level protection.
2. Determine the resource rule type code to use for this validation.
3. Write resource rules for the members in that library.
4. Ensure that the resource rules are made globally resident.
5. Define the library, volume, and resource type in the GSO PDS record.
6. REFRESH the PDS record to activate member-level protection for that library.

Note: eTrust CA-ACF2 PDS member level protection utilizes CA Common Services, which requires that the TNG390 CAILIB be in the LINKLST concatenation.

Examples

The following example shows protection of SYS2.PROCLIB so that users can update only those members that begin with a dollar sign($) but are allowed to read all members.

1. SYS2.PROCLIB is the library to be protected.
2. Resource type code PDS is used to write rules for this library.
3. Write the following rules:

```
SET R(PDS)

$KEY($*******) T(PDS)
 UID(-) ALLOW

$KEY(********) T(PDS)
 UID(sysprog) ALLOW
 UID(-) SERVICE(READ) ALLOW
```

4. Add the resource type code to INFODIR.

```
SET CONTROL(GSO)

CHANGE INFODIR TYPES(R-RPDS)
```

5. Perform a console REFRESH and REBUILD to make the rules resident.

```
F ACF2,REFRESH(INFODIR)
F ACF2,REBUILD(PDS)
```

6. Define the library and resource type in the GSO PDS record. Since VOLUME is not specified, SYS2.PROCLIB is protected regardless of where it resides.

```
SET CONTROL(GSO)

INSERT PDS.proc2 LIBRARY(sys2.proclib) RSRCTYPE(PDS)
```

7. Perform a console REFRESH to activate the protection.

```
F ACF2,REFRESH(PDS)
```

You can define libraries to use the same resource type code if their protection requirements are the same. In this example, the same resource rules protect both SYS1.PARMLIB and SYS2.PARMLIB while resource rules with type code PDJ protect the production JCL library:

```
INSERT PDS.parm1 LIB(sys1.parmlib) RSRC(PDS)
INSERT PDS.parm2 LIB(sys2.parmlib) RSRC(PDS)
INSERT PDS.jcl   LIB(prod.jcllib) RSRC(PDJ)
```

You can use the VOLUME keyword to restrict protection to a specific library. This example protects the production parmlib on SYSRES but does not protect any other SYS1.PARMLIB data sets that reside on other volumes:

```
INSERT PDS.parm1 LIB(sys1.parmlib) VOL(sysres) RSRC(PDS)
```

Use of the special resource name PDSALLOW lets you specify a library that is not to be protected. In this example, the parmlib on volume TSTRES is not to be protected, but all other SYS1.PARMLIB libraries are to be protected:

```
INSERT PDS.parm1 LIB(sys1.parmlib) VOL(tstres) RSRC(PDSALLOW)
INSERT PDS.parm2 LIB(sys1.parmlib) RSRC(PDS)
```

You might encounter situations in which you want to bypass member-level protection for a PDS that is normally protected. When a protected library is accessed, a data set validation call is performed with a data set name of CAPDSSEC.BYPASS.rsrctype for write access. The *rsrctype* value is from the GSO PDS record that describes the data set or data sets that are protected. If access is allowed to this data set resource, PDS member-level validation is bypassed.

The following example shows that users in production control do not need member-level protection rules for data sets controlled by the PDJ resource type. Also, system programmers can access any member-level protected data set when using the IEBCOPY program.

```
SET RULE

$KEY(CAPDSSEC)
 BYPASS.PDJ UID(prodcntl) W(A)
 BYPASS.- UID(sysprog) PGM(IEBCOPY) LIB('SYS1.LINKLIB') W(A)
```

Note that if your GSO OPTS MODE setting specifies RULE, and anything other than ABORT is specified for the norule condition, member-level protection is bypassed for all users. If using RULE mode, ensure that a CAPDSSEC access rule with $MODE(ABORT) is created.

The compression of a partitioned data set under member-level protection raises a special concern. Normally, the user performing the compress is required to have READ access to all members of the data set and UPDATE access to the special resource <DIR>.

A sample rule to secure the <DIR> resource is as follows:

```
SET RESOURCE(PDS)
$KEY(<DIR>) TYPE(PDS)
UID(sysprog) SERVICE(UPDATE) ALLOW
```

If this access cannot be given, and the user does not have any overriding privileges, such as NON-CNCL or READALL, use of the BYPASS rule, described above, can allow access. In addition to restricting the user to the IEBCOPY program, other controls, such as SHIFT, SOURCE, and UNTIL, can be used to control this type of access.

Notes and Restrictions

- PDS member-level protection requires that the CA-Common Services must be active and available for this support.
- PDS/E data sets are not under member-level controls.
- Program fetch for execution of load modules is not under member-level controls. This type of access is protected through the existing PROGRAM resource class.
- Execution of members of CLIST libraries that are enabled for VLF caching is not under member-level controls.
- Users still need proper data set access through access rules in addition to member-level access. That is, to read a PDS member, the user requires read access to the data set through data set access rules, and read access to the member through PDS resource rules.
- Users with ALLOC authority to a data set are able to bypass member-level protection controls because of their ability to rename the data set to a name outside of member-level protection.
- When an alias member name is accessed, the true member name is used in the member-level validation.
- The eTrust CA-ACF2 resource exits are not taken for member-level validations. The z/OS SAF router exit, ICHRTX00, can be used to modify the validation. The resource validation call is a RACROUTE FASTAUTH and can be recognized by the value *CAPDS* in the SUBSYS and REQSTOR fields.
- If you are planning on using PDS member-level security, you must have CA-Common Services installed into LNKLST or LPALST, or add to the eTrust CA-ACF2 procedure a STEPLIB statement that points to the library where CA-Common Services is installed.

- PDS member-level security validation process responds in one of two ways:

  1. User is allowed access.
  2. User is denied access and the task abends with S913.

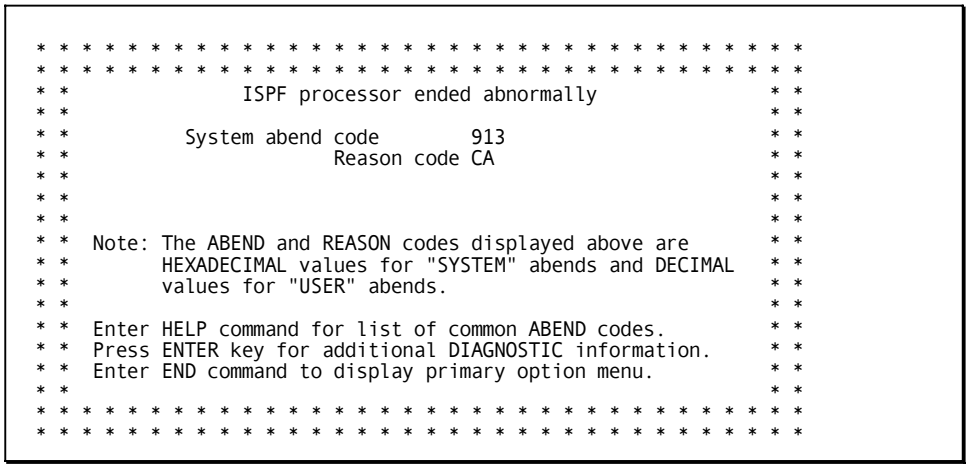
     eTrust CA-ACF2 can only report what is returned back from the IBM RACROUTE process. Therefore, S913 abend serves as a tool to describe why the user is denied and from what PDS library.

     The S913 abends does not mean the TSO ISPF user will be forced to logoff from their current ISPF session. Events are executed depending on what level of access(s) is given to the TSO ISPF user.

     When the TSO ISPF user only has READ access to the PDS member and attempts to update it, the following events take place:

     - Task abends with S913.
     - User receives a warning message informing them that they are denied.
     - User is returned back to the PDS member to allow them to cancel out.

     In another case, when the user does not have a READ access to the PDS member and attempts to read it, they are presented with the error recovery screen:



Troubleshooting

- The PDS member-level protection component is started automatically by eTrust CA-ACF2 when GSO PDS records are present. The initialization program name is CAS4INIT. A message at eTrust CA-ACF2 startup or refresh indicating that module CAS4INIT cannot be found means that the member-level protection component has not been properly installed.
- Utility programs CAS4STAT and CAS4TRCE are available for problem resolution. Technical Support might ask that you run these programs.

  CAS4STAT displays overall system status, location of key modules and other statistics. Run a job with the following EXEC statement:

  ```
  //PDS      EXEC PGM=CAS4STAT,REGION=6M
  ```

  CAS4TRCE enables tracing from the member-level protection component. Trace messages are issued to the system console using WTO. Run a job with the following EXEC statement:

  ```
  //PDS      EXEC PGM=CAS4TRCE,REGION=4M,PARM='opt'
  ```

  Where *opt* can be:

  ON—Enables general tracing.

  SEC—Enables tracing of security processing.

  CCW—Enables general tracing and also traces CCW I/O activity against a protected data set. This setting can potentially generate a lot of output and, depending on how many data sets are protected, impact system performance.

  MEM—Enables general tracing and also generates tracing showing PDS member information including disk address translation. This setting can generate an abundance of output.

  ALL—Enables all tracing.

  OFF—Disables all tracing.
- Utility program CAS4TERM can be run to disable the PDS member-level protection component. To accomplish this, run a job with the following EXEC statement:

  ```
  //PDS      EXEC PGM=CAS4TERM,REGION=4M
  ```

  To reenable the component, run a job with the following EXEC statement:

  ```
  //PDS      EXEC PGM=CAS4INIT,REGION=4M
  ```

- The authority to run the CAS4TERM, CAS4STAT and CAS4TRCE utility programs is controlled by the FACILITY security class. The default type code for FACILITY rules is FAC. The resource rule $KEY values that let a user execute these utility programs are as follows:

  CAPDSSEC.TERM

  CAPDSSEC.STATUS

  CAPDSSEC.TRACE

  For example, a rule to allow a system programmer to run the CAS4STAT and CAS4TRCE programs, but not the CAS4TERM program, follows:

```
SET R(FAC)

$KEY(CAPDSSEC) T(FAC)
 TERM UID(*) PREVENT
 STATUS UID(sysprog) ALLOW
 TRACE UID(sysprog) ALLOW
```

Appendix

E

National Language Support

eTrust CA-ACF2 provides national language support for many eTrust CA-ACF2 messages. When delivering a message to a terminal, console, or job, it determines the national language of the recipient of the message. If the message is available in the target language, the national language version of the eTrust CA-ACF2 message is delivered.

Establishing eTrust CA-ACF2 Global and User Languages

Global (system-wide) primary and secondary languages are specified in the eTrust CA-ACF2 GSO OPTS record, documented in the "Maintaining Global System Options Records" chapter. If specified, these languages are used as the languages for eTrust CA-ACF2 system and console messages and as the default for user messages if no user languages are assigned in a user profile record.

Primary and secondary languages are specified for an individual user in a USER PROFILE LANGUAGE record, documented in the "Maintaining Profile Records" chapter. If specified, these languages are used as the languages for messages delivered to a TSO session, batch job, or started task, and in returning messages to eTrust CA-ACF2 SVC requests. If no user languages were established for a user, the global languages specified in the GSO OPTS record are used as a default.

For eTrust CA-ACF2 national language support, the USER PROFILE directory and records must be globally resident. This is done by specifying them as resident in the GSO INFODIR record, documented in Infostorage Rule Directories (INFODIR) in the "Maintaining Global System Options Records" chapter.

Implementing National Language Support

eTrust CA-ACF2 national language support is implemented using MVS Message Services (MMS). Briefly, MMS is a z/OS system service address space that knows about a set of languages and has data sets called run-time message libraries. There is a run-time message library for each language supported. The data set contains all of the messages defined to MMS in that language. You can find more information about MVS Message Services and its implementation in the IBM MVS/ESA documentation set, specifically the *z/OS MVS Programming: Assembler Services Reference*, the *MVS/ESA Assembler Services Guide* (MVS/ESA 5.x and above) or the *MVS/ESA Assembler Programming Guide* (MVS/ESA 4.x), and the *MVS/ESA Initialization and Tuning Reference*.

The *Getting Started Guide* describes implementation of MVS Message Services and eTrust CA-ACF2 national language support.

Customizing eTrust CA-ACF2 Message Text

Since the eTrust CA-ACF2 message skeletons are distributed, customer sites can change the text of messages. If they do, they must adhere to the MMS message skeleton format described in the *IBM MVS/ESA Assembler Services Guide* and the limitations described in the comment section of the eTrust CA-ACF2 message skeleton members.

You can obtain the message skeletons for eTrust CA-ACF2 messages in other languages from the Computer Associates national support office in the respective countries.

Restrictions

National language support is available for most eTrust CA-ACF2 messages. It is not available for some eTrust CA-ACF2 messages and for interface specific messages issued by the eTrust CA-ACF2 product interfaces, such as eTrust CA-ACF2 CICS and eTrust CA-ACF2 IMS.

Appendix

F

RACF to eTrust CA-ACF2 Translation

Many applications and products describe the setup for external security in RACF terms. This appendix describes RACF terminology so that an eTrust CA-ACF2 administrator can create the necessary eTrust CA-ACF2 records.

For information about a particular product not covered here, contact eTrust CA-ACF2 Level 1 Technical Support.

RACF Segments and eTrust CA-ACF2 Profiles

User related information is stored by RACF in various database segments. ETrust CA-ACF2 accomplishes this using the LOGONID and PROFILE records. The following table lists the RACF segments and where the corresponding information is stored in eTrust CA-ACF2.

LOGONID indicates that the information is stored in the logonid record. PROFILE indicates that the information is in a separate profile record. PROFILE* indicates that the information is in a separate profile record, but that optional default data is stored in an Information Storage record pointed to by the SMSINFO field in the logonid record.

RACF Segment	eTrust CA-ACF2 Equivalent	Profile Name	Segment
CATEGORY	PROFILE	SECLABEL	CATEGORY
CERTDATA	PROFILE	USER	CERTDATA
DCE	PROFILE	USER	DCE
DFP	PROFILE*	DATASET	DFP
DLFDATA	PROFILE	DLFCLASS	DLFDATA
CICS	LOGONID or PROFILE	USER	CICS
KEYSMSTR	PROFILE	KEYSMSTR	SSIGNON
LANGUAGE	PROFILE	USER	LANGUAGE

RACF Segment	eTrust CA-ACF2 Equivalent	Profile Name	Segment
NETVIEW	PROFILE	USER	NETVIEW
OMVS	PROFILE	USER GROUP	OMVS OMVS
OPERPARM	PROFILE	USER	OPERPARM
SECLEVEL	PROFILE	SECLABEL	SECLEVEL
SECLABEL	PROFILE	USER or SECLABEL	SECLABEL
SESSION	PROFILE	APPCLU	SESSION
TSO	LOGONID		
BASE	LOGONID		
WORKATTR	PROFILE	USER	WORKATTR

BASE and TSO Segment Considerations

This section describes the BASE and TSO segments. The data for fields within these segments is within the eTrust CA-ACF2 logonid record.

The tables in the following sections identify fields within the RACF BASE and TSO segments that have identical eTrust CA-ACF2 logonid fields. The comments section identifies how the field can be used with the RACROUTE REQUEST=EXTRACT SAF call.

- By saying that a field is extractable, this means that it can be extracted via a RACROUTE REQUEST=EXTRACT,TYPE=EXTRACT (or EXTRACTN) SAF call.
- By saying that a field is replaceable, this means that it can be replaced by a RACROUTE REQUEST=EXTRACT,TYPE=REPLACE SAF call.

When using the SAF RACROUTE REQUEST=EXTRACT mechanism, you must see fields according to their name in the RACF database, **not** the corresponding eTrust CA-ACF2 name. This distinction becomes important when you use the SAF RACROUTE REQUEST=EXTRACT facility to obtain from and alter data within the BASE and TSO user profile segments.

Note: Not all RACF database fields within the BASE and TSO segments have equivalent fields within the eTrust CA-ACF2 logonid record. Some fields are RACF-specific and simply do not pertain to eTrust CA-ACF2. Other fields are implemented differently under CA-ACF2 and cannot be queried and/or administrated by the SAF RACROUTE REQUEST=EXTRACT facility. For example, the RACF MAGSTRIP field provides Operator ID (OID) support during signon processing. eTrust CA-ACF2 provides similar, but not identical functionality with Extended User Authentication (EUA). The eTrust CA-ACF2 EUA implementation is superior because it provides more functionality, and it provides greater administrative flexibility.

BASE Segment Fields

The following table identifies RACF BASE segment fields that have equivalent eTrust CA-ACF2 logonid fields.

RACF Field Name	eTrust CA-ACF2 Equivalent	Comments
PASSINT	MAXDAYS	eTrust CA-ACF2 supports both extract and replace.
PASSWORD	PASSWORD	eTrust CA-ACF2 supports only for extract.
PASSDATE	PSWD-TOD	eTrust CA-ACF2 supports only for extract.
PGMRNAME	NAME	eTrust CA-ACF2 supports both extract and replace.
DFLTGRP	GROUP	eTrust CA-ACF2 supports both extract and replace.
LJTIME	ACC-TIME	eTrust CA-ACF2 supports only for extract.
LJDATE	ACC-DATE	eTrust CA-ACF2 supports only for extract.
REVOKECT	PSWD-INV	eTrust CA-ACF2 supports only for extract.

Note: eTrust CA-ACF2 does not support any of the RACF combination fields within the BASE user segment.

TSO Segment Fields

The following table identifies RACF TSO segment fields having equivalent eTrust CA-ACF2 Logonid fields.

RACF Field Name	eTrust CA-ACF2 Equivalent	Comments
TACCNT	TSOACCT	eTrust CA-ACF2 supports both extract and replace.
TDEST	DFT-DEST	eTrust CA-ACF2 supports both extract and replace.
THCLASS	DFT-SUBH	eTrust CA-ACF2 supports both extract and replace.
TJCLASS	DFT-SUBC	eTrust CA-ACF2 supports both extract and replace.
TLPROC	TSOPROC	eTrust CA-ACF2 supports both extract and replace.
TLSIZE	TSORGN	eTrust CA-ACF2 supports both extract and replace.
TMCLASS	DFT-SUBM	eTrust CA-ACF2 supports both extract and replace.
TMSIZE	TSOSIZE	eTrust CA-ACF2 supports both extract and replace.
TOPTION	LIDTFLG2	eTrust CA-ACF2 supports only for extract.
TPERFORM	TSOPERF	eTrust CA-ACF2 supports both extract and replace.
TRBA	TSORBA	eTrust CA-ACF2 supports both extract and replace.
TSCLASS	DFT-SOUT	eTrust CA-ACF2 supports both extract and replace.
TUNIT	TSOUNIT	eTrust CA-ACF2 supports both extract and replace.
TUPT	Various fields	CA-ACF2 supports only for extract.

RACF Commands

ADDGROUP Command

RACF uses the ADDGROUP command to add a group definition. For example:

```
ADDGROUP OMVSGRP OMVS(GID(1))
```

In eTrust CA-ACF2, this definition is added to a profile record as follows:

```
SET PROFILE(GROUP) DIV(OMVS)
INSERT OMVSGRP GID(1)
```

Anytime you insert, change, or delete profile records, you must bring the changed profile records into storage as follows:

```
F ACF2,REBUILD(USR),CLASS(P)
```

In addition, if you changed the OMVS or OMVSGRP profile records, you must also rebuild the in-storage tables as follows:

```
F ACF2,OMVS
```

Note: See the section on supplemental groups in a previous chapter for more information.

ADDSD Command

RACF uses the ADDSD command to define profiles for data sets that are to be protected. This process is unnecessary in eTrust CA-ACF2 because all data sets are protected by default.

ADDUSER Command

RACF uses the ADDUSER command to define a new user to its database with the profile information necessary to let that user use the desired components of the system. The ADDUSER command is the same as the INSERT command in eTrust CA-ACF2. For example, the RACF command:

```
ADDUSER USER01 DFLTGRP(OMVSGRP) OMVS(UID(200) HOME(/) OMVSPROGRAM(/bin/sh))
PASSWORD(password)
```

Would be rendered in eTrust CA-ACF2 as follows:

```
INSERT USER01 GROUP(OMVSGRP) PASSWORD(password)
SET PROFILE(USER) DIV(OMVS)
INSERT USER01 UID(200) HOME(/) OMVSPGM(/bin/sh)
```

Anytime you insert, change, or delete profile records, you must bring the changed profile records into storage as follows:

```
F ACF2,REBUILD(USR),CLASS(P)
```

In addition, if you changed the OMVS USER or OMVS GROUP profile records, you must also rebuild the in-storage tables as follows:

```
F ACF2,OMVS
```

ALTDSD Command

In sitatuions where PROTECTALL is not active, RACF uses the ALTDSD command to add profiles for data sets that have been defined as protected. This process is unnecessary in eTrust CA-ACF2 because all data sets are automatically protected by default.

ALTGROUP Command

RACF uses the ALTGROUP command to change an existing group profile. For example:

```
ALTGROUP OMVSGRP OMVS(G10(2))
```

In eTrust CA-ACF2, the function of modifying an existing profile is performed with a CHANGE command. For example:

```
SET PROFILE(GROUP) DIV(OMVS)
CHANGE OMVSGRP GID(2)
```

Anytime you insert, change, or delete profile records, you must bring the changed profile records into storage as follows:

```
F ACF2,REBUILD(USR),CLASS(P)
```

In addition, if you changed the OMVS or OMVSGRP profile records, you must also rebuild the in-storage tables as follows:

```
F ACF2,OMVS
```

ALTUSER Command

RACF uses the ALTUSER command to change an existing user's profile. For example:

```
ALTUSER USER01 CICS(OPCLASS(10) OPIDENT(U01) TIMEOUT(30))
```

Or:

```
ALTUSER USER01 OMVS(HOME('/u/user01'))
```

eTrust CA-ACF2 uses the CHANGE command to accomplish the same thing by changing a logonid or profile record. For example:

```
SET LID
CHANGE USER01 CICSCL(10) CICSID(U01) IDLE(30)
```

Or:

```
SET PROFILE(USER) DIV(OMVS)
CHANGE USER01 HOME(/u/user01/)
```

Anytime you insert, change, or delete profile records, you must bring the changed profile records into storage as follows:

```
F ACF2,REBUILD(USR),CLASS(P)
```

In addition, if you changed the OMVS or OMVSGRP profile records, you must also rebuild the in-storage tables as follows:

```
F ACF2,OMVS
```

BLKUPD Command

RACF uses the BLKUPD command to let users repair the RACF database by directly changing its internal elements. This command is for RACF administration and would have no effect on an eTrust CA-ACF2 environment.

CONNECT Command

RACF uses the CONNECT command to connect or add an RACF userid to an existing group. This gives the userid the accesses associated with that group. For example:

```
CONNECT USERA GROUP(PAYGRP)
```

In eTrust CA-ACF2, grouping individuals together is handled with the UID string and the individual logonid fields that eTrust CA-ACF2 uses to construct the UID string. For example, if you want to add a user to the payroll department, you would issue the following command:

```
SET LID
CHANGE USERA DEPT(PAY)
```

DELDSD Command

In the event that PROTECTALL is not active, RACF uses the DELDSO command to delete a profile defining a data set for RACF protection. This process is unnecessary in eTrust CA-ACF2 because all data sets are automatically protected by default.

DELGROUP Command

RACF uses the DELGROUP command to delete a RACF group profile. For example:

```
DELGROUP PAYROLL
```

There is no direct corollary in eTrust CA-ACF2. eTrust CA-ACF2 uses the UID string to group people and only uses a group in selected environments such as UNIX System Services. A group is assigned to an individual logonid in the GROUP field and defined in the GROUP profile records. Deleting a group in this instance would be done as follows:

```
SET LID
CHANGE LOGONID GROUP()
SET PROFILE(GROUP) DIV(OMVS)
DELETE PAYROLL
```

DELUSER Command

RACF uses the DELUSER command to delete a user profile from RACF. For example:

```
DELUSER USERA
```

In eTrust CA-ACF2 you remove a user from the database by deleting the logonid as follows:

```
SET LID
DELETE USERA
```

LISTDSD, LISTGRP, and LISTUSER Commands

RACF uses the LISTDSD, LISTGRP, and LISTUSER commands to display the corresponding data set, group, and user profiles. For example:

```
LISTUSER USERA
```

eTrust CA-ACF2 provides a LIST command to let you list the individual or selected groups or database records. For instance, to list a logonid you would do the following:

```
SET LID
LIST USERA
```

PASSWORD Command

RACF uses the PASSWORD command to change a user's password or change interval. The same function is accomplished in eTrust CA-ACF2 using the CHANGE command on the logonid. For example:

```
SET LID
CHANGE USERA PASSWORD(NEWONE)
```

PERMIT Command

The RACF PERMIT command allows access to resources. For example:

```
PERMIT SYS1.PARMLIB CLASS(DATASET) ID(user) ACCESS(READ)
```

Or:

```
PERMIT USI161ME.HOGWA01.HOGWA01C.JOB03567.D0000002 CLASS(jesspoolclass) ID(user)
ACCESS(READ)
```

Based on the resource class, eTrust CA-ACF2 would use access rules or resource rules to perform the same control. If the resource class is DATASET, DASDVOL, or TAPEVOL, eTrust CA-ACF2 uses access rules to validate a request. Any other class is controlled with resource rules. The first RACF example uses a class of DATASET, so in eTrust CA-ACF2 this is an access rule as follows:

```
$KEY(SYS1)
 PARMLIB UID(user) R(A)
```

The second RACF example uses a class that is not one of the classes that uses access rules, so eTrust CA-ACF2 uses a generalized resource rule as follows:

```
$KEY(USI161ME) TYPE(SPL)
 HOGWA01.- UID(user) SERVICE(READ) ALLOW
```

RACDCERT Command

RACF uses the RACDCERT command to administer digital certificates. This command allows authorized users to add, list, modify, generate and delete certificates. It can also be used to generate certificate requests. See the RACF to eTrust CA-ACF2 Translation section of the Digital Certificate chapter for examples.

RALTER Command

The RACF RALTER command allows changes to an existing profile. For example:

```
RALTER PROGRAM * ADDMEM('CBC.SCLBDLL') UACC(READ)
```

In eTrust CA-ACF2, the equivalent function would be handled by making changes to an access rule or a resource rule and then recompiling it.

RDEFINE Command

Where a resource is not protected by default, RDEFINE is used in RACF to define resources. There is no counterpart to this in eTrust CA-ACF2. eTrust CA-ACF2 uses a default protection scheme, which assumes that the resource is protected. This default scheme requires that rules be written to allow access to a resource.

RDELETE Command

RACF uses the RDELETE command to remove a resource from RACF protection, and to remove other RACF administrative functions. There is no counterpart to this command in eTrust CA-ACF2. Removing resources from protection is the same as writing a generic resource rule to allow access to the resource class.

REMOVE Command

RACF uses the REMOVE command to take one or more users out of a specific group. Since eTrust CA-ACF2 has a grouping structure based on the UID string, moving a user from group to group is simply a change to the value in the appropriate field contained within the UID string.

RLIST Command

RACF uses the RLIST command to list the details of resource profiles. It is also used to perform refreshes of resource profiles. eTrust CA-ACF2 uses the LIST command to display the various records found in the eTrust CA-ACF2 databases. For refreshing resource rules that are globally resident, eTrust CA-ACF2 uses the F ACF2, REBUILD command. For locally resident rules, the SETNORUL command releases the old copies of rules in an address space forcing the address space to acquire new copies.

RVARY Command

RACF uses the RVARY command to deactivate or reactivate RACF functions or resources. There is no direct counterpart to this command in eTrust CA-ACF2.

SEARCH Command

RACF uses the SEARCH command to display information from its database based on specified search criteria. eTrust CA-ACF2 provides similar functionality in its LIST command and reports such as ACFRPTSL, ACRRPTXR, and ACFRPTRX.

SETROPTS Command

SETROPTS is used to set RACF options or turn them off. For example:

```
SETROPTS CLASSACT(JESSPOOL)
```

eTrust CA-ACF2 uses SAFDEF records as needed to activate various protection schemes. Since most SAF calls are protected by default, it is usually not necessary to add additional SAFDEF records. The eTrust CA-ACF2 SECTRACE and SHOW SAFDEF commands can be used to verify the SAF environment that an application is using.

For example, the SHOW SAFDEF command indicates that the class of JESSPOOL is ignored so it would need a SAFDEF record to activate as follows:

```
INSERT SAFDEF.JESSPOOL ID(JESSPOOL) MODE(GLOBAL)
-RACROUTE(REQUEST=AUTH,CLASS=JESSPOOL) REP
```

eTrust CA-ACF2 also uses GSO records to control options. Changes to the GSO records can be made via the F ACF2, REFRESH command.

If finer detail is required, the SECTRACE output could supply additional information that could be added to the SAFDEF record to make the request more specific.

CLASS

This defines a type of resource. eTrust CA-ACF2 divides resources into data set resources or general resources. If the class is DATASET, DASDVOL, or TAPEVOL, eTrust CA-ACF2 uses data set access rules to validate access. All other classes will use generalized resource rules. The currently defined classes and their eTrust CA-ACF2 type codes can be found by issuing an ACF SHOW CLASMAP command.

If the resource is not defined or you want to change the type codes used in the resource rules for a class, you can use a GSO CLASMAP record to accomplish this. For example, the JESSPOOL class comes predefined with a type code of SAF. To change this, you would issue:

```
SET CONTROL(GSO)
INSERT CLASMAP.JESSPOOL RESOURCE(JESSPOOL) RSRCTYPE(SPL) ENTITYLN(53)
```

RACF Attribute Translation

The following table shows how eTrust CA-ACF2 translates a RACF attribute.

RACF Attribute	eTrust CA-ACF2 Access Rule	eTrust CA-ACF2 Resource Rule
READ	READ	READ
UPDATE	WRITE	UPDATE
ALTER	ALLOC	ADD
CONTROL	WRITE	DELETE
EXECUTE	EXECUTE	EXECUTE

Note: In RACF, a higher attribute assumes access to the entire lower attribute. For example, giving update access to a resource would assume read access. eTrust CA-ACF2 does not make this assumption. Access to a resource must be specifically given.

Program Control (PADS)

RACF Program Control allows for the definition of an exact program and library environment. A similar function exists in eTrust CA-ACF2 called program pathing. This feature is used in data set access rules to specify the exact path (program and/or library) for accessing a particular data set or group of data sets.

RACF Attributes

The following table shows selected RACF attributes, their purpose, and the equivalent eTrust CA-ACF2 privilege that would be used to achieve the same result in an eTrust CA-ACF2 environment.

RACF Attribute	Purpose	eTrust CA-ACF2 Equivalent
PRIVILEGED	Lets a user bypass security checking without any logging.	NON-CNCL
TRUSTED	Lets a user bypass security checking but logging can be turned on for this user.	NON-CNCL
OPERATIONS	Lets a user bypass security checking for selected resource classes.	NON-CNCL
SPECIAL	Used in RACF administration.	SECURITY
AUDITOR	Lets a user review profile information.	AUDIT
REVOKE	Prevents a user from accessing the system.	SUSPEND or CANCEL

Index

?

@

_

+

2

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Z