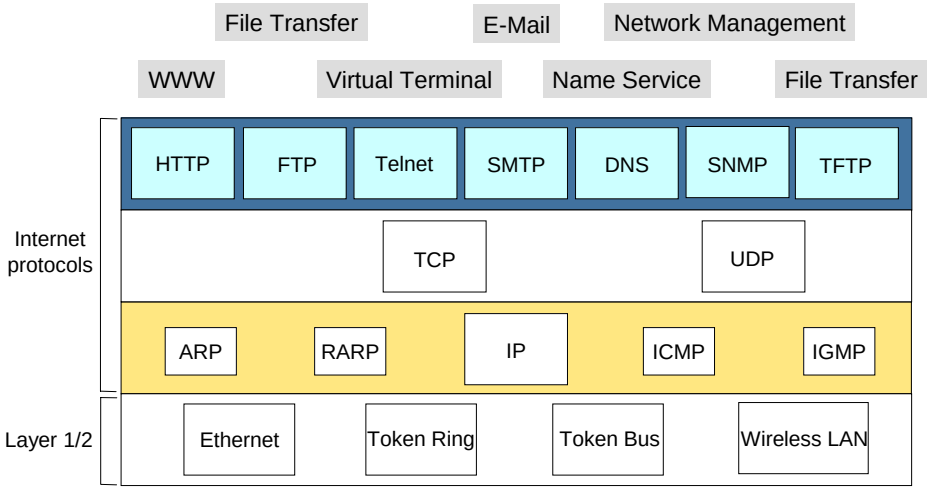
Application Protocols in the TCP/IP Reference Model



Application Protocols in the TCP/IP Reference Model

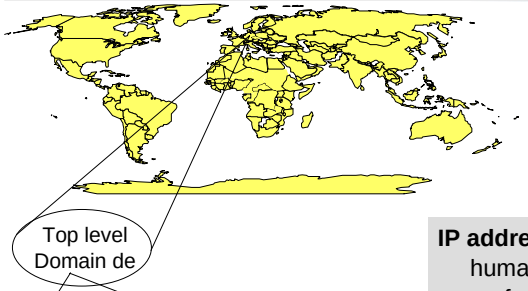
Protocols of the application layer are common communication services

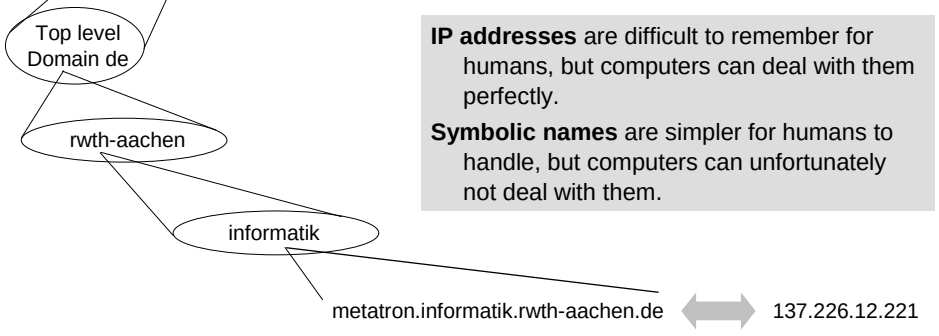
Protocols of the application layer are defined for special purposes and specify

- The *types* of the sent messages
- The *syntax* of the message types
- The *semantics* of the message types
- Rules for definition, *when* and *how* an application process sends a message resp. responses to it

Usually: Client/Server structure. Processes on the application layer are using *TCP(UDP)/IP-Sockets*

DNS - Domain Name System

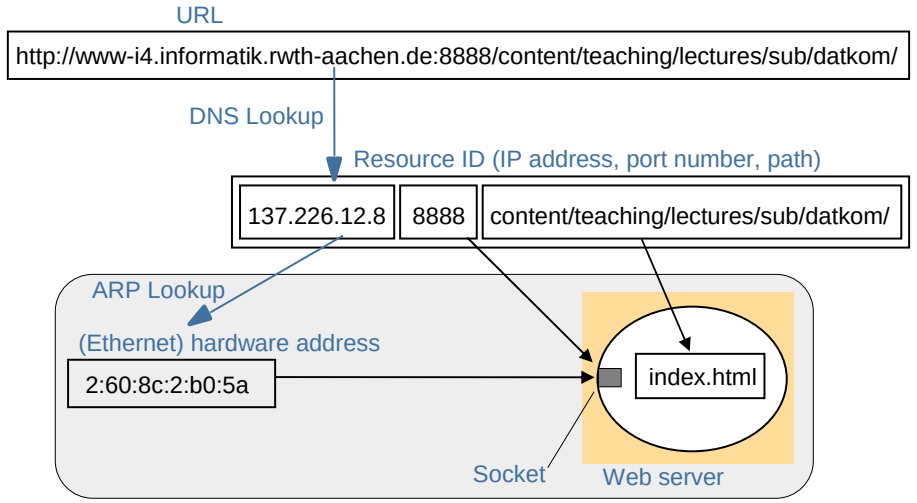




IP addresses are difficult to remember for humans, but computers can deal with them perfectly.

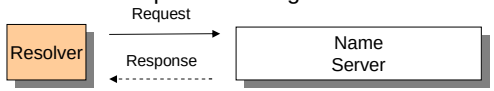
Symbolic names are simpler for humans to handle, but computers can unfortunately not deal with them.

Access to Remote Hosts

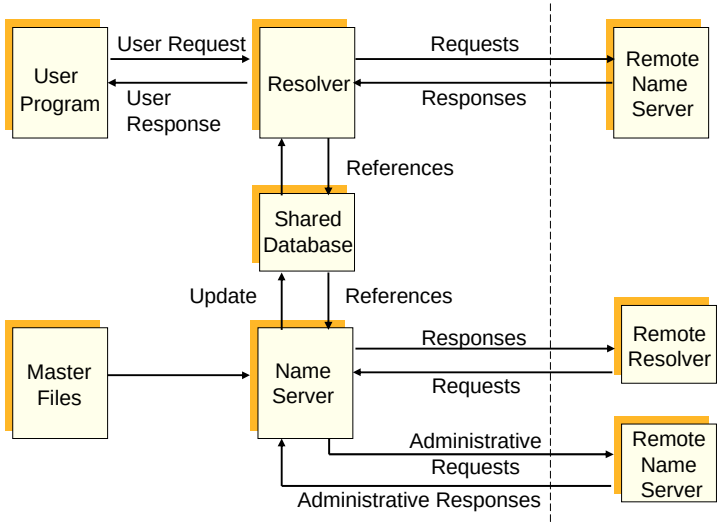


DNS - Concept

1. DNS manages the mapping of logical computer names to IP addresses (and further services)
2. DNS is a distributed database, i.e. the individual segments are subject to *local control*
3. The structure of the used name space of the database shows the administrative organization of the Internet
4. Data of each local area are available by means of a Client/Server architecture in the entire network
5. Robustness and speed of the system are being achieved by replication and caching of the naming data
6. Main components:
   - **Name Server:** Server which manages information about a part of the database
   - **Resolver:** Client which requests naming information from the server

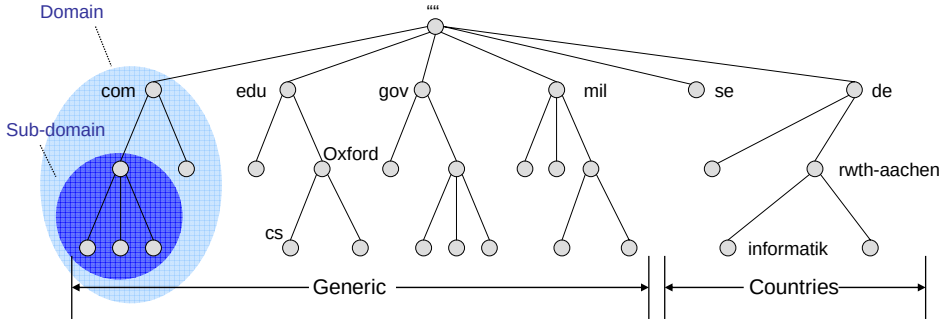


DNS - Architecture



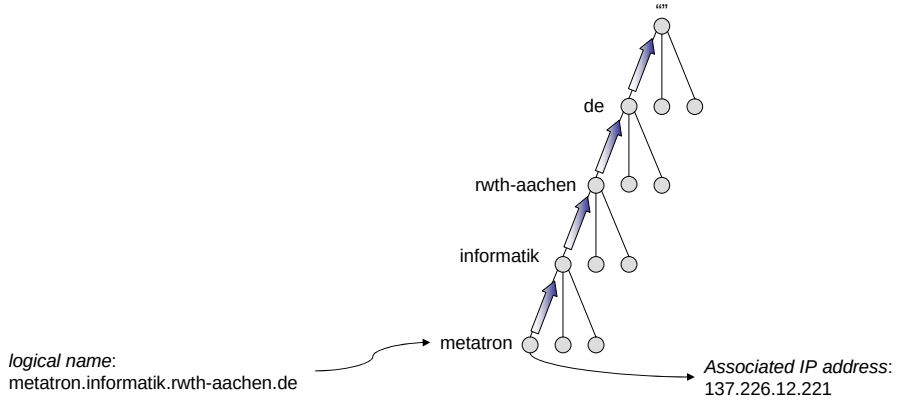
Structure of the Database

- For structuring of all information: the database can be represented as a tree
- Each node of the tree is marked with a label, which identifies it relatively to the father node
- Each (internal) node is root of a sub-tree
- Each of those sub-trees represents a **domain**
- Each domain can be divided into **sub-domains**



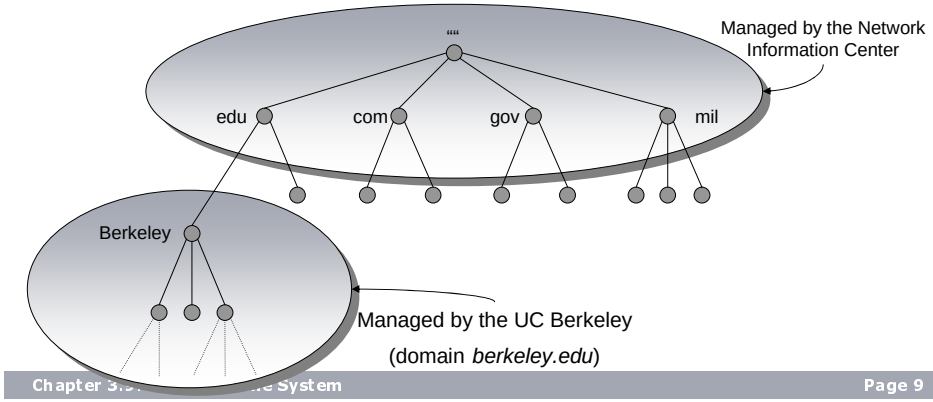
Domain Names

- The name of a domain consists of the sequence of labels (separated by “.”) beginning with the root of the domain and going up to the root of the whole tree
- In the leaf nodes the IP addresses associated with the names given by the label sequence are being stored



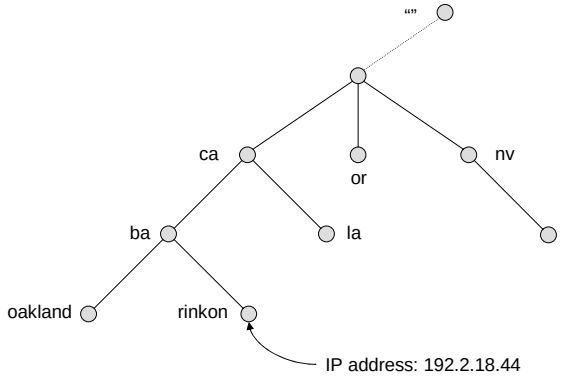
Administration of a Domain

- Each domain can be managed by another organization
- The responsible organization can split a domain into sub-domains and delegate the responsibility for them to other organizations
- The father domain manages pointers to the roots of the sub-domains to be able to forward requests to them
- The name of a domain corresponds to the domain name of the root node



Index of the Database

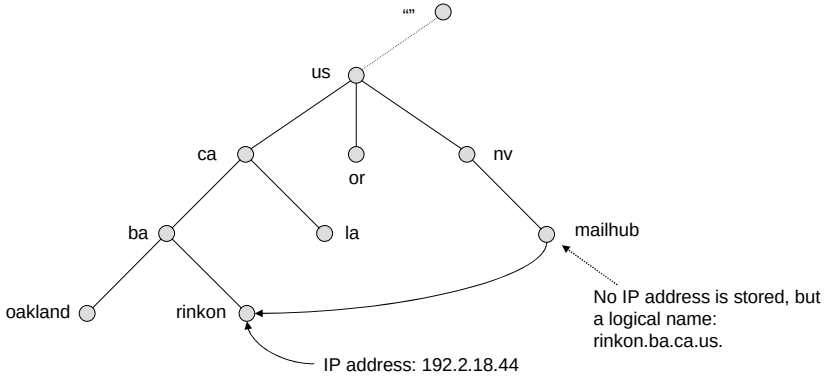
- The names of the domains serve as index for the database
- Each computer in the network has a domain name which refers to further information concerning the computer



The data associated with a domain name are stored in so-called **Resource Records** (RR)

Domain Name Aliases

- Computers can have one or more secondary names, so-called *Domain Name Aliases*
- Aliases are pointers of one domain name to another one (canonical domain name)



Name Space

The reverse tree represents the *Domain Name Space*

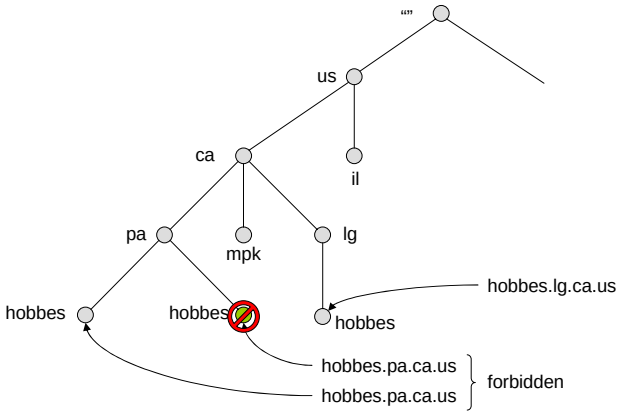
- The depth of the tree is limited to 127 levels
- Domain names can have up to 63 characters
- A label of the length 0 is reserved for the root node (“”)
- The **Fully Qualified Domain Name** (FQDN) is the **absolute domain name**, which is declared with reference to the root of the tree

  Example: informatik.rwth-aachen.de.

- Domain names which are declared not with reference to the root of the tree, but with reference to another domain, are called **relative domain names**

Name Collisions

- Nodes with the same father node must have different labels
- The hierarchical name space prevents the occurrence of collisions



Domains

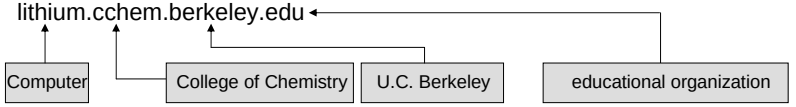
- A domain consists of all computers whose domain name is within the domain
- Leafs of the tree represent individual computers and refer to network addresses, hardware information and mail routing information
- Internal nodes of the tree can describe both a computer and a domain
- Domains are denoted often relatively or regarding their level:
  - **Top-Level Domain:** child of the root node
  - **First Level Domain:** child of the root node (top-level domain)
  - **Second Level Domain:** child of a first level of domain
  - etc.

Top Level Domains

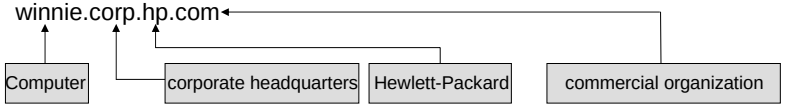
- Originally the name space was divided into seven top-level domains:
  1. **com**: commercial organizations
  2. **edu**: educational organizations
  3. **gov**: government organizations
  4. **mil**: military organizations
  5. **net**: network organizations
  6. **org**: non-commercial organizations
  7. **int**: international organizations
- Additionally, each country got its own top-level domain
- The name space was extended in the meantime by further top-level domains
- Within the individual top-level domains, different conventions for name structuring are given:
  - Australia: edu.au, com.au, etc.
  - UK: co.uk (for commercial organizations), ac.uk (for academic organizations), etc.
  - Germany: completely unstructured

Examples of Domain Names

- lithium.cchem.berkeley.edu

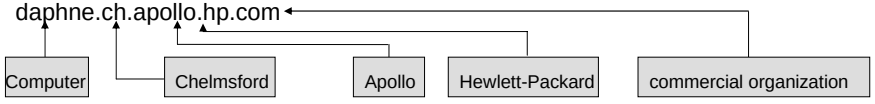


- winnie.corp.hp.com



- fernwood.mpk.ca.us

- daphne.ch.apollo.hp.com

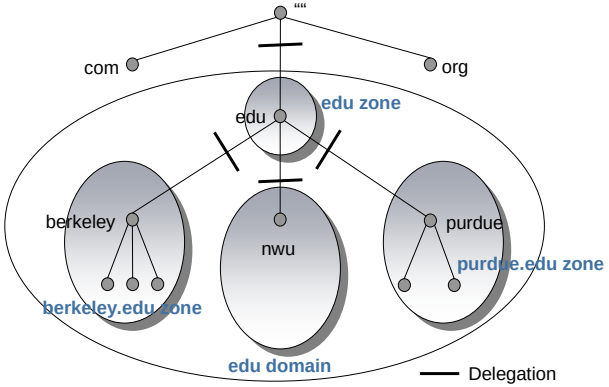


Name Servers and Zones

- Information about the name space are stored in name servers
- Name Servers manage the whole information for a certain part of the name space; this part is called **zone**
- The information about a zone is loaded either from a file or from another name server
- The name server has the authority for the zone
- A name server can be responsible for several zones

Domains and Zones

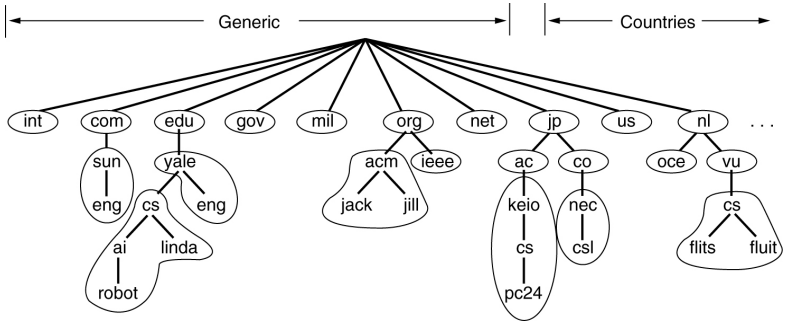
- Domain and zone are different concepts:



- Zones are (except within the lowest levels of the tree) smaller than domains, therefore servers have to manage less name information

Zones

There are no guidelines how domains are divided into zones. Each domain can select a dividing for itself.



Some zones (e.g. edu) do not manage IP addresses. As information they only store references to other zones

Zones and Delegation

- A zone contains the domain names, which the domain with the same domain name contains, apart from domain names in delegated sub-domains
- Example:
  - Top-level domain ca (Canada) has the sub-domains ab.ca (Alberta), on.ca (Ontario), qc.ca (Quebec)
  - Responsibility for the sub-domains ab.ca, on.ca and qc.ca is delegated to the name servers in the provinces
  - The **domain ca** covers all data in ca as well as all data in ab.ca, on.ca and qc.ca
  - The **zone ca** contains only the data in .ca, which mainly are pointers to the delegated sub-domains
- Name servers load zones instead of domains, since a domain contains more information than needed by the name server
- Example:
  - The root name server, which loads the root domain and with it the entire name space instead of the root zone

Types of Name Servers

- The **Primary Master** of a zone (also called **Master**) reads the data from a file configured by an administrator.
- A **Secondary Master** of a zone (also called **Slave**) receives the data from another name server, which is *authoritative* for the zone. In most cases this is the primary master. A secondary master can receive the data however also from another secondary master.
- When a secondary master is started, it contacts the master server and loads, if necessary, the zone data (**zone transfer**).
- Both, the primary master and the secondary masters are authoritative for the zone.
- The distinction between primary master and secondary master serves for a controlled replication of the data and thus increases both, the performance and the fault tolerance.

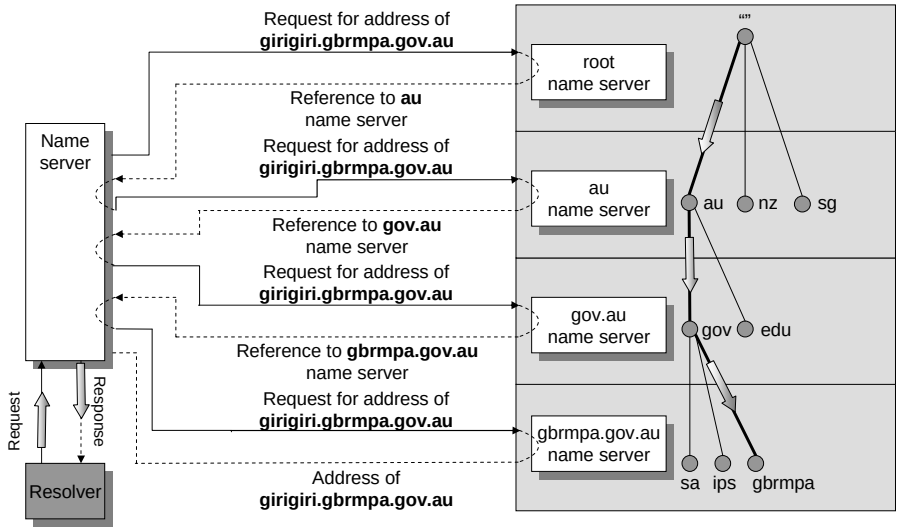
Data of a Zone

- The primary master reads the zone data from appropriate files (*Zone Data Files*)
- A secondary master can likewise read its zone data from these files
- A secondary master usually saves the data received from a primary master in appropriate files
- With a restart of a secondary master it first reads the saved data in the files to determine whether these are current
- The backup copies thus prevent unnecessary zone data transfer and do serve at the same time as additional source if the primary master is not available
- The files contain *Resource Records* which describe the zone's name information
- The resource records describe all computers in the zone as well as information concerning the delegation of sub-domains

Name Resolution

- Generally mapping of names to addresses
- The term **Name Resolution** also designates the process, in which a name server searches the name space for data, for which he is not responsible
- For the searching, a name server needs the domain name and the addresses of the **root name servers**
- A name server can ask a root name server for each name in the name space
- Root name servers know the responsible servers for each top-level domain
- On request, a root name server can return names and addresses of name servers responsible for the top-level domain of the searched name
- The top level name server again manages references to name servers which are responsible for the second level domain
- If additional information is missing, each search begins with the root name servers

Iterative Name Resolution

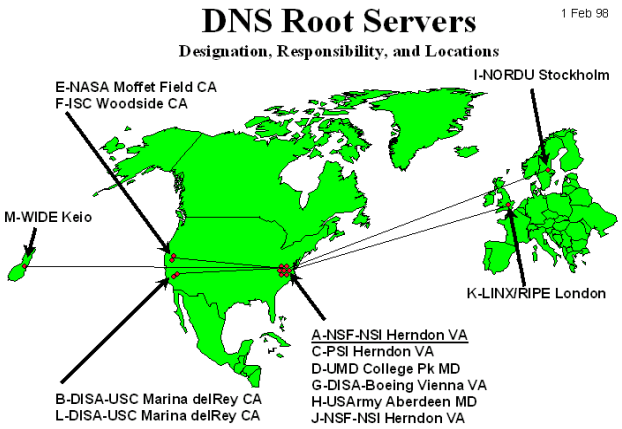


Recursive Resolution

- Distinction between **recursive** and **iterative** requests resp. **recursive** and **iterative name resolution**
- In case of recursive resolution, a resolver sends a recursive inquiry to a name server
- The name server must answer either with the searched information or an error message, i.e. the name server may not refer to another name server
- If the addressed name server is not responsible for the searched information, it must contact other name servers
- The name server can start a recursive or iterative inquiry; usually it will use an iterative inquiry
- With the inquiry, the name server tries to shorten the resolution process by directing the inquiry to the most suitable name server regarding the searched information (i.e. if known, a server on a lower level is contacted instead of the root name server)

Root Name Server

- Requests to which a name server cannot answer, are handed upward in the tree
- Name server on the upper levels are heavily loaded
- Inquiries, which go into another zone, often run over the root name server
- Thus, the root name server must always be available
- Therefore: *replication* - there are 13 instances of the root name server, more or less distributed over the whole world

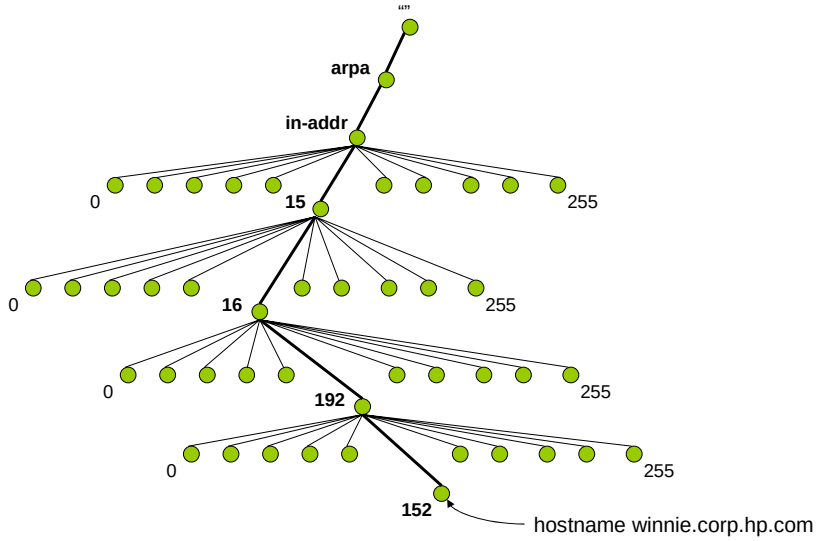


Problem: very central placement of the servers!

Mapping of Addresses to Names

- Information in the database is indicated by names
- Mapping of a name to an address is simple
- Mapping of an address onto a name is more difficult to realize (complete search of name space)
- Solution:
  - Place a special area in the name space, which uses addresses as label; the **in-addr.arpa domain**
  - Nodes in this domain are marked in accordance with the usual notation for IP addresses (four octets separated by points)
  - The in-addr.arpa domain has 256 sub-domains, each of which again having 256 sub-domains, …
  - On the fourth level, the appropriate resource records are assigned with the octet, which refers to the domain name of the computer or the network with the indicated address
  - The IP address appears backwards because it is read beginning with the leaf node (IP address: 15.16.192.152
    => sub-domain: 152.192.16.15.in-addr.arpa)

Mapping of Addresses to Names



Caching & Time to Live

- **Caching** is the process of buffering name information in a name server not responsible for those information. In further requests these information are present and the name resolution process can be speeded up
- Stored are not only information about the requested hosts, but additionally all information about other name servers used in the resolution process
- The **Time to Live** (TTL) indicates how long data are allowed to be buffered
- The TTL guarantees that no outdated information is used
  - Small TTL gives a high consistency
  - Large TTL gives a faster resolution of a name

Resource Record

- Entries in the zone data files the name server are **resource records**
- General structure: `(label, ttl, class, type, value)`

Type	Used in…	Description
SOA	Zone	Indicates the authority for the zone data
A	Host	Contains the IP address of a host; needed for name resolution
MX	Domain	Refers to the mail server of the domain
SRV	Domain	Refers to a server which offers a certain service in the domain
NS	Zone	Refers to a responsible name server for the zone
CNAME	Node	Canonical name, i.e. reference to the actual node
PTR	Host	Used for the mapping of an address to a name
HINFO	Host	Additional information to the host (CCU, operating system)
TXT	arbitrary	Other useful information

Example: Resource Records in a Zone File

```
ripe.net          7200  IN   SOA    ns.ripe.net.   olaf.ripe.net. (
                                        2001061501     ; Serial
                                        43200  ; Refresh 12 hours
                                        14400  ; Retry 4 hours
                                        345600 ; Expire 4 days
                                        7200 ; Clear cache 2 hours
                                        )
ripe.net          7200  IN   NS     ns.ripe.net.
ripe.net          7200  IN   NS     ns.eu.net.

pinkje.ripe.net   3600  IN   A      193.0.1.162
host25.ripe.net   2600  IN   A      193.0.3.25
```

Label — ttl — class — type — value

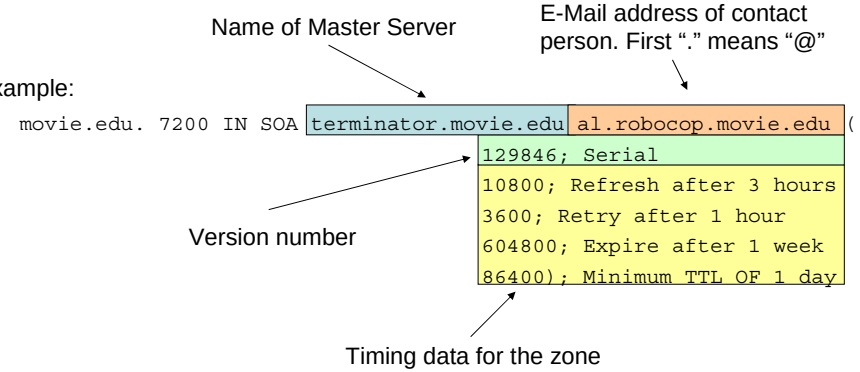
IN = Internet addresses

Example: Resource Records in a Zone File

```
; Authoritative data for cs.vu.nl
cs.vu.nl.        86400  IN  SOA     star boss (952771,7200,7200,2419200,86400)
cs.vu.nl.        86400  IN  TXT     "Divisie Wiskunde en Informatica."
cs.vu.nl.        86400  IN  TXT     "Vrije Universiteit Amsterdam."
cs.vu.nl.        86400  IN  MX      1 zephyr.cs.vu.nl.
cs.vu.nl.        86400  IN  MX      2 top.cs.vu.nl.

flits.cs.vu.nl.  86400  IN  HINFO   Sun Unix
flits.cs.vu.nl.  86400  IN  A       130.37.16.112
flits.cs.vu.nl.  86400  IN  A       192.31.231.165
flits.cs.vu.nl.  86400  IN  MX      1 flits.cs.vu.nl.
flits.cs.vu.nl.  86400  IN  MX      2 zephyr.cs.vu.nl.
flits.cs.vu.nl.  86400  IN  MX      3 top.cs.vu.nl.
www.cs.vu.nl.    86400  IN  CNAME   star.cs.vu.nl
ftp.cs.vu.nl.    86400  IN  CNAME   zephyr.cs.vu.nl

rowboat                 IN  A       130.37.56.201
                        IN  MX      1 rowboat
                        IN  MX      2 zephyr
                        IN  HINFO   Sun Unix

little-sister           IN  A       130.37.62.23
                        IN  HINFO   Mac MacOS

laserjet                IN  A       192.31.231.216
                        IN  HINFO   "HP Laserjet IIISi" Proprietary
```

SOA Record

- SOA = **Start of Authority**
- It indicates that the name server is authorative for the zone
- There can be only one SOA record in an appropriate file



SOA Record

- Attributes of the SOA record:
  - **Serial:** Serial number which serves the secondary master for the recognition of new versions of the zone data
  - **Refresh:** Time interval, at whose expiration the secondary master examines the topicality of its data
  - **Retry:** time interval; if the secondary master cannot contact the primary master at expiration of the refresh time, then it tries again after expiration of the retry time interval
  - **Expire:** if the secondary master cannot contact the primary master after the indicated length of time, it stops answering inquiries because it must assume its data is outdated
  - **TTL:** Refers to all resource records. This value is returned as part of the answer on a request to instruct other servers about the maximal time for caching the data.

NS Record

- NS = **Name Server**
- For each name server of a zone a NS record is created

- Example:

```
movie.edu. IN NS terminator.movie.edu
movie.edu. IN NS wormhole.movie.edu
```

- There are two name servers, which are in the regarded example installed on the computers terminator and wormhole

Address and Alias Records

- A = **ADDRESS**
- CNAME = **Canonical Name**
- At least one A record is needed for each host in the zone, CNAME records are optional
- Example:

```
; Host addresses
 localhost.movie.edu.     IN     A     127.0.0.1
 robocop.movie.edu.       IN     A     192.249.249.2
 terminator.movie.edu.    IN     A     192.249.249.3
 diehard.movie.edu.       IN     A     192.249.249.4
 misery.movie.edu.        IN     A     192.253.253.2
 shining.movie.edu.       IN     A     192.253.253.3
 carrie.movie.edu.        IN     A     192.253.253.4
;
; Multihomed host
;
 wormhole.movie.edu       IN     A     192.249.249.1
 wormhole.movie.edu       IN     A     192.253.253.1
```

Address and Alias Records

```
;
; Aliases
;
 bigt.movie.edu.     IN    CNAME    terminator.movie.edu.
 dh.movie.edu.       IN    CNAME    diehard.movie.edu.
 wh.movie.edu.       IN    CNAME    wormhole.movie.edu.
 wh249.movie.edu.    IN    A        192.249.249.1
 wh253.movie.edu.    IN    A        192.253.253.1

 A = ADDRESS
 CNAME = illustrates an alias on its canonical names
```

- For *multihomed* computers (connected with several networks), an own A record is needed for every secondary name if different aliases are to be stored for the addresses
- For a secondary name, which applies to both addresses, a CNAME record is created

PTR Record

- PTR = **Pointer**
- Provides information for the mapping of addresses to names

- Example:

```
1.249.249.192.in-addr.arpa.   IN   PTR   wormhole.movie.edu.
2.249.249.192.in-addr.arpa.   IN   PTR   robocop.movie.edu.
3.249.249.192.in-addr.arpa.   IN   PTR   terminator.movie.edu.
4.249.249.192.in-addr.arpa.   IN   PTR   diehard.movie.edu.
```

- Addresses should refer only one name, the original or canonical name

MX Record

- MX = **Mail Exchanger**
- MX record serves for the controlling of e-mail routing
- Specifies a mail server responsible for a domain name, which processes or passes on e-mail
- Additionally, a preference can be indicated if several mail servers are present

- Example:

```
peets.mpk.ca.us.   IN      MX     10      relay.hp.com.
```

  indicates that relay.hp.com is the mail server for peets.mpk.ca.us with the preference 10

- Only the relative preference value is important; the mail server with the smallest value is addressed first

nslookup

- Program for placing DNS inquiries
- Offers both an interactive and a non-interactive mode
- Interactive mode:

```
[aoxomoxoa:thissen] 42> nslookup
Default Server:  nets1.rz.RWTH-Aachen.DE
Address:  137.226.144.3
```

- Non-interactive mode:

```
[aoxomoxoa:thissen] 43> nslookup metatron
Server:  nets1.rz.RWTH-Aachen.DE
Address:  137.226.144.3
Name:    metatron.info-4.informatik.rwth-aachen.de
Address:  137.226.12.221
```

- the default name server for the zone is `nets1.rz.RWTH-Aachen.DE`, i.e. each inquiry is sent to this name server

DNS Protocol

DNS defines only one protocol format, which is used both for inquiries and for responses:

- *Identification*: 16 bits for the definite identification of an inquiry, to match requests and responses
- Flag: 4 Bit, marking of (1) request/response, (2) authorative/not authorative, (3) iterative/recursive, (4) recursion possible
- *„Number of… “*: Indication of the contained number of inquiries resp. data records
- *Questions*: Names to be resolved
- *Answers*: Resource records to the previous inquiry
- *Authority*: Identification of passed responsible name servers
- *Additional information*: further data to the inquiry. If the name searched is only an alias, the belonging resource record for the correct name is placed here

Identification	Flag
Number of Questions	Number of Answers RR
Number of Authority RR	Number of Additional RR
Questions (variable number of RR)	
Answers (variable number of RR)	
Authority (variable number of RR)	
Additional information (variable number of RR)	