Directory Services

PRINCIPLES – NIS – LDAP – DNS
Labs and deadlines

- AMD -> Intel
- Just nu uppgraderar vi kernel
  - Hoppas det löser interupts -> slött nätverk
- Deadlines
- Om jag glömmer (vilket jag gör) säg till mig om att lägga ut föreläsningsslides
What is a directory?

**Fundamental properties**
- Maps keys to values
- Relatively frequent lookups
- Relatively infrequent updates

**Examples**
- Phone book
- Office directory
- User database
- List of contacts
Directories in Linux

User database
- /etc/passwd, /etc/shadow

Group database
- /etc/group

Host names
- /etc/hosts

Network names
- /etc/network

Protocol names
- /etc/protocols

Service names
- /etc/services

RPC program numbers
- /etc/rpc

Known ethernet addresses
- /etc/ethers

Automount maps
- /etc/auto.master

Standard implementation: local files
The scalability problem

Example
- 13000 users and 5000 hosts
- Passwords valid for 30 days
- 50% of changes made at 8-10
  - One change every 28.8 seconds
  - Propagation time: 0.00567s

Problems
- Performance issues
- Hosts that are down
- Other propagation failures
- Simultaneous updates
What is a directory service

A specialized database
- Attribute-value type information
- More reads than updates
- Consistency problems are sometimes OK
- No transactions or rollback
- Support for distribution and replication
- Clear patterns to searches
Directory services

Components
- A data model
- A protocol for searching
- A protocol for reading
- A protocol for updating
- Methods for replication
- Methods for distribution

Common directory services
- DNS
- X.500 Directory Service
- Network Information Service
- NIS+
- Active Directory (Windows NT)
- NDS (Novell Directory Service)
- LDAP (Lightweight X.500)
### Directory services

#### Global directory service
- **Context:** entire network or entire internet
- **Namespace:** uniform
- **Distribution:** usually
- **Examples:** DNS, X.500, NIS+, LDAP

#### Local directory service
- **Context:** intranet or smaller
- **Namespace:** non-uniform
- **Examples:** NIS, local files
Directory services in Linux

Alias: name services
- /etc/nsswitch.conf selects service
- Several services per directory
- Modular design/implementation

Examples from /etc/nsswitch.conf
users  files,nis
users  nis[notfound=return],files
hosts  dns,files
NIS, NIS+, LDAP
Network Information Service

Domain (NIS domain)
- Systems administered with NIS
- No connection to DNS domain

NIS server
- Server that has information accessible through NIS
- Serves one or more domains

NIS client
- Host that uses NIS as a directory service for something
NIS

Protocol
- RPC based
- No security
- No updates
- Replication support

Data model
- Directories known as maps
- Simple key-value mapping
- Values have no structure

Replication
- Master/slave servers

Distribution
- No distribution support!

<table>
<thead>
<tr>
<th>passwd.byname</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>donkn</td>
<td>1002:*:203</td>
</tr>
<tr>
<td>johne</td>
<td>1003:trzQw</td>
</tr>
<tr>
<td>alatu</td>
<td>2031:kprrT</td>
</tr>
<tr>
<td>johmc</td>
<td>2032:bRelZ</td>
</tr>
<tr>
<td>edwyo</td>
<td>2033:*:204</td>
</tr>
<tr>
<td>ricst</td>
<td>2034:vv1dk</td>
</tr>
<tr>
<td>petde</td>
<td>2232:*:204</td>
</tr>
<tr>
<td>larwa</td>
<td>3021:*:204</td>
</tr>
</tbody>
</table>
NIS

Master server
- Maps built from text files
- Maps in `/var/yp`
- Maps built with `make`
- Maps stored in binary form
- Replication to slaves with `yppush`

Slave servers
- Receive data from master
- Load balancing and failover

Processes/commands
- `ypserv` Server process
- `ypbind` Client process
- `ypcat` To view maps
- `ypmatch` To search maps
- `ypwhich` Show status
- `yppasswdd` Change password
NIS

NIS client
- Knows its NIS domain
- Binds to a NIS server

Two options
- Broadcast
- Hard coded NIS-server

- ybind
NIS

Scalability problems
- Flat namespace
- No distribution

Security problems
- No access control
- Broadcast for binding
- Patched as an afterthought

Primitive protocol
- No updates
  - Hack for password change
- Search only on key
- Primitive data model

Solution: NIS+
NIS+

**Scalability**
- Hierarchical namespace
- Distributed administration

**Security**
- Authentication of server, client and user
- Access control on per-cell level

**New protocol**
- Updates through NIS+
- General searches
- Data model with real tables

So why is NIS+ not used?
LDAP

Protocol
- TCP-based
- Fine-grained access control
- Support for updates
- Flexible search protocol

Replication
- Replication is possible

Data model
- Based on X.500
- Object-oriented
- Objects can be extended freely
- Attribute-based data model
- Hierarchical namespace

Distribution
- Distributed management is possible
Example of user

NIS+ table "passwd.org_dir.example.com"

<table>
<thead>
<tr>
<th>name</th>
<th>passwd</th>
<th>uid</th>
<th>gid</th>
<th>gecos</th>
<th>home</th>
<th>shell</th>
</tr>
</thead>
<tbody>
<tr>
<td>davby</td>
<td><em>LK</em></td>
<td>1211</td>
<td>1200</td>
<td>David</td>
<td>/home/davby</td>
<td>/bin/sh</td>
</tr>
<tr>
<td>fsmith</td>
<td>3x1231v76T89N</td>
<td>1329</td>
<td>1200</td>
<td>Fran</td>
<td>/home/fsmith</td>
<td>/bin/sh</td>
</tr>
</tbody>
</table>

NIS table passwd.byname (user name as key):

<table>
<thead>
<tr>
<th>name</th>
<th>password</th>
</tr>
</thead>
<tbody>
<tr>
<td>davby</td>
<td>davby:*:1211:1200:David:/home/davby:/bin/sh</td>
</tr>
<tr>
<td>fsmith</td>
<td>fsmith:*:1329:1200:Fran:/home/fsmith:/bin/sh</td>
</tr>
</tbody>
</table>
Example of user

```plaintext
dn: uid=fsmith,ou=employees,dc=example,dc=com
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
uid: fsmith
givenname: Fran
sn: Smith
cn: Fran Smith
cn: Frances Smith
telephoneNumber: 510-555-1234
roomNumber: 122G
o: Example Corporation International
mailRoutingAddress: fsmith@example.com
mailHost: mail.example.com
userPassword: {crypt}3x1231v76T89N
uidNumber: 1329
gidNumber: 1200
homeDirectory: /home/fsmith
loginShell: /bin/sh
```
The future

LDAP is taking over

- NIS is too insecure, doesn’t scale and is inflexible
- NIS+ is hard to implement and doesn’t exist on many OSes
- X.500 is too complex and has a bad reputation
- Other options have similar problems
DNS
DNS: Data model

- Functional:  \( \text{NAME} \to \{ \text{TYPE} \to \text{RDATA} \} \)
- Relational:  \((\text{NAME, TYPE, RDATA})\)

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>RDATA</th>
</tr>
</thead>
<tbody>
<tr>
<td>ida.liu.se</td>
<td>A</td>
<td>130.236.177.25</td>
</tr>
<tr>
<td>ida.liu.se</td>
<td>MX</td>
<td>0 ida.liu.se</td>
</tr>
<tr>
<td><strong>ida.liu.se</strong></td>
<td>NS</td>
<td><strong>ns.ida.liu.se</strong></td>
</tr>
<tr>
<td>ida.liu.se</td>
<td>NS</td>
<td>ns1.liu.se</td>
</tr>
<tr>
<td>ida.liu.se</td>
<td>NS</td>
<td>ns2.liu.se</td>
</tr>
<tr>
<td>ida.liu.se</td>
<td>NS</td>
<td>nsauth.isy.liu.se</td>
</tr>
</tbody>
</table>
DNS: TYPE & RDATA

**TYPE**
- SOA – Start of authority
- NS – Name server
- MX – Mail exchanger
- A – Address
- AAAA – IPv6 address
- PTR – Domain name pointer
- CNAME – Canonical name
- TXT – Text

**RDATA**
- Binary data, hardcoded format
- TYPE determines format

... and many more
DNS: Namespace

Names
- Dot-separated parts
  - one.part.after.another

FQDN
- Fully Qualified Domain Name
- Complete name
- Always ends in a dot

Partial name
- Suffix of name implicit
- Does not end in a dot

Namespace
- Global and hierarchical

Diagram:
```
<root>
  
  com
  
  net
  
  org
  
  se

  google

  ibm

  www

  ibm

  liu

  www

  www

  www

  www

  www

  www
```

www
DNS: Replication

Secondary/slave nameserver

- Indicated by NS RR
- Data transfer with AXFR/IXFR

Questions

- How does a slave NS know when there is new information?
- How often should a slave NS attempt to update?
- How long is replicated data valid?

Example

sysi-00:~# host -t ns ida.liu.se
ida.liu.se  NS  nsauth.isy.liu.se
ida.liu.se  NS  ns.ida.liu.se
ida.liu.se  NS  ns1.liu.se

Rule of thumb

- Every zone needs at least two nameservers
DNS: Distribution

**Delegation**
- A NS can delegate responsibility for a subtree to another NS
- Only **entire** subtrees can be delegated

**Zone**
- The part of the namespace that a NS is authoritative for
- Defined by SOA and NS

**Domain**
- A subtree of the namespace
DNS: Delegation

Delegating NS
- **NS record** for delegated zone
- A record (glue) for NS when needed

Delegated-to NS
- **SOA record** for the zone

Example

```
a.example.com    NS    ns2.x.com

b.x.com         NS    ns.b.x.com
ns.b.x.com      A     10.1.2.3

b.x.com         SOA   (         
                 ns.b.x.com
                 dns.x.com
                 20040909001
                 24H 2H 1W 2D       )
```
# DNS: Delegation

## Format of SOA

- **MNAME**: Master NS
- **RNAME**: Responsible (email)
- **SERIAL**: Serial number
- **REFRESH**: Refresh interval
- **RETRY**: Retry interval
- **MINIMUM**: TTL for negative reply

## SERIAL

- Increase for every update
- Date format common
  - 2004090901

## REFRESH/RETRY

- How often secondary NS updates the zone

## MINIMUM

- How long to cache NXDOMAIN
DNS: Cacheing

Cacheing creates scalability
- Cacheing reduces tree traversal
- Cacheing of A and PTR reduce duplicate DNS queries

Choosing good cache parameters is vital

Cache parameters
- TTL – Set per RR
- Negative TTL – Set in SOA

Example
$TTL 4H SOA ( MNAME RNAME
SERIAL REFRESH
RETRY 1H )
24H NS ns
ns 24H A 10.1.2.3
DNS: The server

Recursive/iterative
- Does the server offer recursion?
- To which clients is it offered?

Authoritative/nonauthoritative
  ...
- Authoritative: first-hand information
- Otherwise: cached information

Review
- **Recursive**: the nameserver gives a definite answer, but may ask other nameservers in order to generate it
- **Iterative**: the nameserver gives a definite answer only for locally known information; otherwise it generates a referral
DNS: The client

Client requirements
- Use a recursive NS (resolver)
- Use partially qualified names

Partially qualified names
- Add suffix if there are fewer than \( n \) dots in the name (\textsf{ndots})

Name server (resolver)
- Specified in /etc/resolv.conf

Example: /etc/resolv.conf
\texttt{search ida.liu.se}
\texttt{nameserver ns.ida.liu.se}
\texttt{ndots 2}
DNS: Root Name Server

Handles the root zone
- Data generated by ICANN
- Data distributed by Verisign
- Distribution from hidden master

Why no more than 13?

Thirteen services
- Some are anycast
- Over 60 servers
<table>
<thead>
<tr>
<th>Operator</th>
<th>Locations</th>
</tr>
</thead>
<tbody>
<tr>
<td>A VeriSign</td>
<td>Dulles VA</td>
</tr>
<tr>
<td>B ISI</td>
<td>Marina Del Rey CA</td>
</tr>
<tr>
<td>C Cogent Communications</td>
<td>Herndon VA; Los Angeles; New York City; Chicago</td>
</tr>
<tr>
<td>D University of Maryland</td>
<td>College Park MD</td>
</tr>
<tr>
<td>E NASA Ames</td>
<td>Mountain View CA</td>
</tr>
<tr>
<td>F Internet Systems Consortium, Inc.</td>
<td>Ottawa; Palo Alto; San Jose, CA; New York City; San Francisco; Madrid; Hong Kong; Los Angeles; Rome; Auckland; Sao Paulo; Beijing; Seoul; Moscow; Taipei; Dubai; Paris; Singapore; Brisbane; Toronto; Monterrey; Lisbon; Johannesburg; Tel Aviv; Jakarta; Munich;</td>
</tr>
<tr>
<td>G U.S. DOD NIC</td>
<td>Vienna VA</td>
</tr>
<tr>
<td>H U.S. Army Research Lab</td>
<td>Aberdeen MD</td>
</tr>
<tr>
<td>I Autonomica/NORDUnet</td>
<td>Stockholm; Helsinki; Milan; London; Geneva; Amsterdam; Oslo; Bangkok; Hong Kong; Brussels; Frankfurt</td>
</tr>
<tr>
<td>J VeriSign Global Registry Services</td>
<td>Dulles VA (2 locations); Mountain View CA; Seattle WA; Amsterdam; Atlanta GA; Los Angeles CA; Miami; Stockholm; London; Tokyo; Seoul; Singapore; Sterling VA (2 locations, standby)</td>
</tr>
<tr>
<td>K RIPE NCC</td>
<td>London; Amsterdam; Frankfurt; Athens; Doha (Quatar)</td>
</tr>
<tr>
<td>L ICANN</td>
<td>Los Angeles</td>
</tr>
<tr>
<td>M WIDE Project</td>
<td>Tokyo; Seoul (KR); Paris (FR)</td>
</tr>
</tbody>
</table>
DNS: CNAME

Canonical name
- Pointer within namespace
- Johansson: See Johnson

CNAME Whoopsie 1
www CNAME informatix
www A 130.236.177.12

CNAME Whoopsie 2
ida.liu.se. NS ns.ida.liu.se.
ns CNAME vitalstatistix
vitalstatistix A 130.236.177.12

www.ida.liu.se  CNAME  informatix.ida.liu.se
DNS: PTR

Address-to-name mapping
- Same RR type for IPv4 och IPv6
- ”A big reverse zone in the sky”

IPv4: in-addr.arpa.
- Reverse address and add in-addr.arpa.
- Same as any other name in DNS!
  - Same lookup, cache etc.
  - CNAME works too

15.189.236.130.in-addr.arpa. PTR sysi-05.sysinst.ida.liu.se.
DNS: Delegation in in-addr.arpa.

Delegation
- Delegation of entire subtrees
- Subtrees at each dot
- In in-addr.arpa a dot after each octet of the address

Q: How to delegate partial subtrees corresponding to small subnets, e.g. 10.17.1.0/26?
A: Use CNAME to create a new zone that can be delegated!
A: Delegate each address as a separate zone

$\text{GENERATE 1-63} \ \$ \ \text{CNAME} \ \$.rv4.sysinst.ida.liu.se.$
DNS: The protocol

TCP or UDP
- Normally UDP port 53
- TCP if the reply is too large

DNS packet
- Header section Flags etc.
- Query section Queries to the server
- Answer section Replies to the queries
- Authority section Referrals to other NS
- Additional section Extra data that may be useful (e.g. glue)
DNS: The protocol

Header section: flags
- QR Query or response
- OPCODE Type of quer
- AA Authoritative Answer
- TC TrunCation
- RD Recursion Desired
- RA Recursion Available
- Z Reserved
- RCODE Result code

Flags
- Set RD for recursive quer
- If AA is not set, reply is from cache
- If TC it set, the reply is too large for UDP

RCODE
- SERVFAIL Problem with NS
- NXDOMAIN No such name
- REFUSED Refuse to reply
DNS: The protocol

**Question section**
- Contains questions
- Also included in reply

**Answer section**
- Contains requested RRs
- Empty in referral replies

**Authority section**
- Indicates authoritative NS
- Never empty in referrals

**Additional section**
- RR related to response, but not part of response
- E.g. A for NS in authority section
sysi-00:~# dig www.ida.liu.se @a.ns.se
; <<< DiG 9.2.4rc5 <<< www.ida.liu.se @a.ns.se
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 7059
;; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 4, ADDITIONAL: 4

;; QUESTION SECTION:
;www.ida.liu.se. IN A

;; AUTHORITY SECTION:
liu.se. 86400 IN NS ns2.liu.se.
liu.se. 86400 IN NS sunic.sunet.se.
liu.se. 86400 IN NS nsauth.isy.liu.se.
liu.se. 86400 IN NS ns1.liu.se.

;; ADDITIONAL SECTION:
ns1.liu.se. 86400 IN A 130.236.6.251
ns2.liu.se. 86400 IN A 130.236.6.243
sunic.sunet.se. 86400 IN A 192.36.125.2
nsauth.isy.liu.se. 86400 IN A 130.236.48.9
sysi-00:~# dig www.ida.liu.se @nsauth.isy.liu.se
; <<>> DiG 9.2.4rc5 <<>> www.ida.liu.se @nsauth.isy.liu.se
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 49836
;; flags: qr aa rd; QUERY: 1, ANSWER: 2, AUTHORITY: 4, ADDITIONAL: 4

;; QUESTION SECTION:
www.ida.liu.se. IN A

;; ANSWER SECTION:
www.ida.liu.se. 259200 IN CNAME informatix.ida.liu.se.
informatix.ida.liu.se. 259200 IN A 130.236.177.26

;; AUTHORITY SECTION:
ida.liu.se. 259200 IN NS ns1.liu.se.
ida.liu.se. 259200 IN NS ns2.liu.se.
ida.liu.se. 259200 IN NS nsauth.isy.liu.se.
ida.liu.se. 259200 IN NS ns.ida.liu.se.

;; ADDITIONAL SECTION:
ns.ida.liu.se. 259200 IN A 130.236.177.25
ns1.liu.se. 43200 IN A 130.236.6.251
ns2.liu.se. 43200 IN A 130.236.6.243
nsauth.isy.liu.se. 21600 IN A 130.236.48.9
sysi-00:~# dig www.ibm.com @ns.ida.liu.se
; <<>> DiG 9.2.4rc5 <<>> www.ibm.com @ns.ida.liu.se
;; global options:  printcmd
;; Got answer:
;; ---HEADER---
;; opcode: QUERY, status: NOERROR, id: 38042
;; flags: qr rd ra
;; QUERY: 1, ANSWER: 6, AUTHORITY: 4, ADDITIONAL: 4

;; QUESTION SECTION:

;; ANSWER SECTION:
www.ibm.com. 1800 IN A 129.42.16.99
www.ibm.com. 1800 IN A 129.42.17.99
www.ibm.com. 1800 IN A 129.42.18.99

;; AUTHORITY SECTION:

;; ADDITIONAL SECTION:
ns.austin.ibm.com. 70372 IN A 192.35.232.34
ns.watson.ibm.com. 92202 IN A 129.34.20.80
ns.almaden.ibm.com. 70372 IN A 198.4.83.35
DNS: Commands

- **nslookup**
  - Look up names

- **host**
  - Look up data in DNS

- **dig**
  - Look up data in DNS
  - Full access to protocol

- **whois**
  - Information about who has registered a domain
  - Many versions – jwhois is nice

---

Don’t troubleshoot DNS using nslookup. It will only cause grief.
DNS: Server types

Master
- Source of DNS data
- Authoritative for zone

Secondary
- Authoritative for zone

Forwarder
- Cache only
- Forwards queries

Recursive-only
- Performs recursive queries
DNS: Server architecture

- Master
- Forwarder
- Recursive
- Slaves
- Administrator
- Clients
- Firewall
Zone configuration in BIND

Files
- named.conf
- Zone files

In Debian: /etc/bind
- named.conf
- named.conf.local
- named.conf.options
- Zones.rfc1812
- db.0
- db.127
- db.empty
- db.local
- db.root
named.conf

Zone definition (master)

zone "sysinst.ida.liu.se" {
    type master;
    file "/etc/bind/sysinst.zone";
}

Options
- Who can query the server
- Who can update the server
- Which ports to use
- Which address to use

… and so on

Other stuff
- Options
- Access control
$TTL 3600

@ IN SOA (sysinst-gw.ida.liu.se. davby.ida.liu.se. 2006083100 ; Serial 3600 ; Refresh 1h 1800 ; Retry 30min 604800 ; Expire 3600 ; TTL )

IN NS sysinst-gw.ida.liu.se.
IN NS ns.ida.liu.se.

IN MX 10 ida-gw.sysinst.ida.liu.se.

ida-gw IN A 130.236.189.1
debian IN CNAME ida-gw
heretix IN A 130.236.189.62

$GENERATE 0-16 sysi-${0,2,d} A 130.236.189.${10,,d}
$GENERATE 1-8 a$-gw A 130.236.189.${29,,d}
$GENERATE 1-8 b$-gw A 130.236.189.${37,,d}
$GENERATE 1-8 c$-gw A 130.236.189.${45,,d}
More stuff in BIND

- Views
- Dynamic update
- DNSSEC
Directory Service Summary

Properties
- Search-optimized database
- Attribute-based data
- Distributed management for scalability
- Replication for performance and reliability
- Search protocol
- Update protocol

Common directory services
- DNS – Host names etc.
- NIS/NIS+ – Replace local files
- LDAP – General directory service