

- 66 - DNS - Domain Name Service

- **What is DNS : Domain Name Service**

- Resolution of:

HostName	---	>IP Number	Forward resolving
IP	-----	>HostName	Reverse Resolving

- Distributed Database of

- hostnames
- Ip Addresses
- Hosts hardware information
- Sub-Domain DNS addresses

- **What are Domains**

- Host Name System in a similar hierarchy as a UNIX file system.

Difference is that expression of names are in the reverse order as with a path in a file system.

eg. UNIX File system: /usr/local/httpd/htdocs/gif/
Domain names: phd.math.berkeley.edu.

- **Access hierarchy of DNS servers in internet**

- 1.- nodes: eg. berkeley.edu (see O'Reilly Page 24)
- 2.- domain: eg. berkeley.edu and all its sub sub-nodes and sub-domains
- 3.- zones: Part of domain served authoritatively from a name server

- **Domains hierarchy:**

- root domain '.'
- SubDomains or Domains can contain hosts and (sub)domains
- Sometimes name is terminated by '.', it is said to be absolute name (used by some software)
- **Fully Qualified Domain Name (FQDN)**
- Root Name Servers (13 of them) are responsible for Top Level Domains (.com .de .org .net .fr .uk etc)
 - 1 in PSINet (commercial Internet Backbone)
 - 1 in NASA (National American Space Agency)
 - 2 in Europe
 - 1 in Japan

- **Name -> IP Address resolving Sequence in Linux system:**

1. If destination host is seen as a name instead of an IP then the resolver of the local system is asked to resolve the name.
2. The `/etc/host.conf` is read to know the sequence of search:
Note: `/etc/nsswitch` has a similar function but has more flexibility to define the order of search for each specific service. `host.conf` is fading out slowly. In SuSE this file is only available in packet `ypserv`(NIS services)

most used entries in /etc/host.conf

`order hosts, bind, nis` Order of name -> IP search
 (Berkeley Internet Name Domain software)
`multi on` Allows to return multiple host addresses for one hostname if they are present in `/etc/hosts`

less used entries in /etc/host.conf

`nospoof on` After name-->IP, do IP--> name then compare if names don't match, NS query fails
`spoofalert on` If above `nospoof` is on and names don't match then error message is logged through syslog.
 (see `man host.conf` for more information about entries)

3. If `order hosts` is in `/etc/host.conf` then `/etc/hosts` is searched through for name resolution.
4. Entries in `/etc/nsswitch.conf` (Example)

```
# An example Name Service Switch config file. This file should be
# sorted with the most-used services at the beginning.
#
# The entry '[NOTFOUND=return]' means that the search for an
# entry should stop if the search in the previous entry turned
# up nothing. Note that if the search failed due to some other reason
# (like no NIS server responding) then the search continues with the
# next entry.
#
# Legal entries are:
#
#      compat          Use compatibility setup
#      nisplus         Use NIS+ (NIS version 3)
#      nis              Use NIS (NIS version 2), also called
YP
#      dns             Use DNS (Domain Name Service)
#      files           Use the local files
#      db              Use the /var/db databases
#      [NOTFOUND=return] Stop searching if not found so far
#
# For more information, please read the nsswitch.conf.5 manual page.

passwd:      compat
group:       compat
hosts:     files lwres dns
networks: files dns
services:   files
protocols:  files
rpc:        files
ethers:     files
```

```

netmasks:      files
netgroup:      files
publickey:     files

bootparams:    files
automount:     files nis

```

5. If name is not found in `/etc/hosts`

and order `hosts`, bind is in `/etc/host.conf`

then `/etc/resolv.conf` file is read to get first name server IP.

```
nameserver 123.456.789.123
```

```
nameserver 123.456.789.124
```

```
nameserver 123.456.789.125
```

```
#domain linux.local (almost same as search. Deprecated)
```

```
search linux.local linux.home
```

(Domains added to names without '.')

6. This DNS server is then contacted

- if DNS is alive then answer is expected from it.

- if DNS is NOT alive then `/etc/resolv.conf` is read for another nameserver IP address. (up to 3 nameserver possible)

(See next section for Full DNS Queries sequence)

7. If resolver comes back from `/etc/hosts` or DNS with an IP Addr.

then client program makes its service request to this IP. Address.

- **Full DNS Queries sequence:**(see O'reilly page 28)

- Program -----`joe.foo.de`-----> Local Host Resolver

- Local Host Resolver <-----NOT FOUND in `/etc/hosts`

- Local Host Resolver---recursive -`joe.foo.de`----> DNS-1

- DNS-1 -----iterative--`joe.foo.de` ---> Root DNS

- DNS-1 <-----DNS-name and addr. of `.de`-----Root DNS

- DNS-1 -----iterative--`joe.foo.de` ---> DNS-of `.de`

- DNS-1 <---DNS-name and addr. of `foo.de`---DNS-of `.de`

- DNS-1 ---iterative--`joe.foo.de` -----> DNS-of `foo.de`

- DNS-1 <-----ANSWER of Host addr-----DNS-of `foo.de`

- Local Host Resolver<---`joe.foo.de=IP.ADDR`----> DNS-1

- Program <-----`joe.foo.de=IP.Addr`---Local Host Resolver

- **Recursive and Iterative (non-recursive) queries**

Recursive: Ask the DNS to resolve the name and return ONLY with a final answer.

Normally made by:

Resolvers to DNS

or by DNS to forwarders DNS.

Iterative: Ask the DNS to either return with a final answer or a list of DNS-addresses who might know the answer.
Normally made by: DNS to DNS.

What is Source Of Authority(SOA) in DNS

- When a **DNS** is responsible for keeping the main database of a domain, it is said that he is **Source Of Authority** for this domain.
If he gives the responsibility to keep the database of one of his sub-domains to another DNS server that this second DNS will be **Source Of Authority** for this sub-domain. In other words he has Authority to that Name Resolution system for this sub-domain.
- Slaves for the main server can also have authority on the same domain since they normally always have an intact copy of the main database.

• DNS Clients setup

- Enter the order in `/etc/host.conf`

```
order hosts, bind
multi on
```

- Enter the DNS addresses in the `/etc/resolv.conf`
eg.

```
search linux.local
nameserver 192.168.100.133
nameserver 194.25.0.52
nameserver 194.25.0.60
```

Liste of Deutsche DNS found at:

www.fli41.de/german/extern/docu/stable/doc/deutsch/html/skap.html

• DNS Server Setup

- Install package `bind9` and `bind9-utils`
- Files involved:

-`/etc/named.conf`

-Directories:

```
    /var/named/           (Old SuSE)
    or /var/lib/named    (SuSE)
    or /var/cache/bind   (Debian)
    localhost.zone
    root.hint
    slave/      Directory
```

-Editing `/etc/named.conf` File:

- Server options section
- Localhost and Root servers zones section
 - '.' root servers zone
 - localhost forward zone
 - localhost reverse zone
- Domains Zones sections

- Forward master zones
- Reverse master zones
- Forward slave zones
- Reverse slave zones

Cache only DNS (Bind 8/9)

/etc/named.conf:

```
options {
    directory "/var/named";
};

zone "." in {
    type hint;
    file "root.hint";
};

zone "0.0.127.in-addr.arpa" in {
    type master;
    file "127.0.0.zone";
};
```

- **Cache only DNS (Bind 4)**

/etc/named.root:

```
directory                /var/lib/named

cache                    .                named.ca
primary                  localhost        localhost.zone
primary                  0.0.127.in-addr.arpa  127.0.0.zone

forwarders 172.16.2.1 172.16.2.2
```

Example of master /etc/named.conf

```
# overall options of the server section
#
options {
    directory "/var/named";
    # the default is to fail,
    # if the master file is not correct
    check-names master warn;

    # checks in the master zone
    # if the hostname is valid in the client's context.
    pid-file "/var/run/named.pid";
    datasize default;
    stacksize default;
    coresize default;
    files unlimited;
    recursion yes;
    multiple-cnames no;
    #The forwarders are used if answer not found locally
    forwarders { 212.185.252.9; 194.25.0.125; };
};

# defining an additional ACL:
acl can_download { localnets; };

# Root servers zone
zone "." in {
    type hint;
    file "root.hint";
};

# Localhost forward zone
zone "localhost" in {
    type master;
    file "localhost.zone";
    check-names fail;
    allow-update { none; };
};

# Localhost reverse zone
zone "0.0.127.in-addr.arpa" in {
    type master;
    file "127.0.0.zone";
    check-names fail;
    allow-update { none; };
};

# Forward master zone (includes 2 subnets in this example)
zone "stars.priv" in {
    type master;
    file "stars.priv.zone";
    notify yes;
};

# Reverse master zone
zone "10.168.192.in-addr.arpa" in {
    type master;
    file "192.168.10.zone";
    notify yes;
};
```

Syntax of named database files

(see page 77 of POSTFIX manual)

1. Comments are starting with ';'.
2. Each empty line should be closed with ';'.
No real empty lines allowed
3. Except for the **StartOfAuthority** record each record is on one line.
4. Records types:

[*TTL*] is optional and is *in seconds*. [*class*] is normally always IN (internet)

Start Of Authority Record (SOA)

domainname [*TTL*] [*class*] SOA *origin*. *hostmaster* (*extras*)

domainname is the domain that this DNS is Authoritative for. @ for local
origin is the FQDN-hostname where the master zone Database is located including the trailing dot '.'

hostmaster is the email address of the hostmaster and replacing the @ with a dot.

eg. *michel.linux.local* instead of *michel@linux.local*

extras is the list of values like serial number, etc. all inside parentheses set .

Name server(NS)

domainname [*TTL*] [*class*] NS *ServerName*

If *domainname* is blank, then it refers to the SOA-record zone

Slaves NS declarations are needed to allow slaves updates on master restart.

Address record(A)

[*FQDN-]host* [*TTL*] [*class*] A *IP-address*

If [*FQDN-]host* is blank, then it refers to the name in previous record.

Canonical Name record (CNAME)

nickname [*TTL*] [*class*] CNAME [*FQDN-]host*

Host Information (HINFO)

[*FQDN-]host* [*TTL*] [*class*] HINFO *Hardware Software*

Hardware and *Software* are text describing them.

Mail Exchanger Record (MX)

DestinationHost [*TTL*] [*class*] MX *Pref.* *MailServerHost*

DestinationHost is the mail destination host name. Blank is localhost

Pref. is the preference number for *MailServerhost* to use

Smaller numbers have priority

MailServerhost is the Mail Exchange Server to use for this destination.

It MUST contain the FQDN including the trailing dot '.'

Pointer Record(PTR)

rev-IP.in-addr.arpa [*TTL*] [*class*] PTR *HQDN-hostname.*

rev-IP.in-addr.arpa is the IP in reverse order with .in-addr.arpa

HQDN-hostname. is the FQDN of the host including the last dot '.'

It MUST contain the FQDN including the trailing dot '.'

- `/var/named/stars.priv.zone` **Database file**

```

;file /var/named/stars.priv.zone
; SOA Record          SOA=Start of Authority

$TTL 2D
@           IN SOA      SIRIUS root.localhost (
                2000121501    ;serial = Date YYYYMMDDxx
                24H           ;refresh= how often the slave has
                                ; to check that its data are up to date
                2H           ;retry=if the slave fails to reach his master,
                                ; tries to reconnect every retry seconds
                30D          ;expire=if the slave fails to contact his master
                                ; for expire sec, it stops to give out his data.
                4D )         ;minimum TTL (time to live)=the
                                ; maximum time other servers are allowed to
                                ; cache these data

; NS Records
                IN NS SIRIUS    ; Inet Address of name server
                IN NS MOON      ; Inet Address of slave name server
                IN NS SUN       ; Inet Address of slave name server
                                ; Slaves NS are NEEDED so that the
                                ; server updates slaves on reload/restart

; Address and Alias Records
;
MOON         IN A          192.168.2.1
MOON         IN A          192.168.10.1
SIRIUS       IN A          192.168.10.10
SUN          IN A          192.168.10.14
;
ISDN         IN CNAME SIRIUS
NEWS         IN CNAME SIRIUS
GLOBEALL     IN CNAME SIRIUS
VIVATION     IN CNAME SIRIUS
EARTH        IN A          192.168.2.12

```

- `/var/named/192.168.10.zone`:

```

; file /var/named/192.168.10.zone
$TTL 2D          ; TTL 2 Days
@           IN SOA      SIRIUS.stars.priv.  root.localhost. (
                2000071501    ; serial
                24H           ; refresh      = 86400 sec
                2H           ; retry        = 7200 sec
                30D          ; expiry       = 2592999 sec
                4D )         ; minimum TTL = 345600 sec

                IN NS      SIRIUS.stars.priv.
                IN NS      MOON.stars.priv.
                IN NS      SUN.stars.priv.

; PTR Record
;
1           IN PTR      MOON.stars.priv.
10          IN PTR      SIRIUS.stars.priv.
11          IN PTR      SSV046.stars.priv.
12          IN PTR      EARTH.stars.priv.
14          IN PTR      SUN.stars.priv.
15          IN PTR      BIDON.stars.priv.
16          IN PTR      BIDON2.stars.priv.

```

Slave configuration files

- `/etc/named.conf`

```
#
# a slave zone (das ist nur ein Beispiel!)
#
zone "stars.priv" in {
    type slave;
    file "slave/stars.priv.zone";
    masters { 192.168.10.10; };
};

zone "10.168.192.in-addr.arpa" in {
    type slave;
    file "slave/192.168.10.zone";
    masters { 192.168.10.10; };
};

zone "2.168.192.in-addr.arpa" in {
    type slave;
    file "slave/192.168.2.zone";
    masters { 192.168.10.10; };
};
```

- **Mail Exchanger MX Record:**

Syntax:

```
Domainname IN MX Priority-Number hostname
```

Examples:

```
stars.priv. IN MX 10 SIRIUS.stars.priv.
stars.priv. IN MX 20 MOON.stars.priv.
stars.priv. IN MX 30 SUN.stars.priv.
```

The smallest Number has the highest Priority !!!

- **Server stat/stop**

```
rcnamed start
"" "" stop
"" "" restart
"" "" status
```

- **Testing DNS with nslookup, host and dig**

- **nslookup**

```
>Hostname or IP-Addr.
>ls -d Domainname
>server ServerName
>help
>exit or <Str>d
```

- **host** `[-v] hostname` Resolve the hostname.

`-v` = verbose, the output is like with dig.

```
host hostname DNS-Server
```

Use the *DNS-Server* to resolve the *hostname*

- ```

host IP-Address Reverse resolve the IP-Address.
host -l Domain Show all hosts addresses of Domain.
host -t mx Domain Show Mail-Exchange-Servers of Domain
host -a Hostname Shows all DNS info about the Host

```
- **dig** [*@server*] *name* [*type*] like **host** but with more functions (type = any, a, mx, ns etc.)

```

dig sun.linux.local Resolve sun.linux.local
dig @dozlinux sun Resolve sun vom DNS-Server dozlinux
dig linux.local any Show all of Domain linux.local
dig -x IP-Adresse Reverse resolve an IP-Address.

```

- **Dynamic Updating of DNS**

### nsupdate

This utility allows to update records in the DNS dynamically.

- Conditions:
- There should be No fixed records for new dynamic requests.
  - The dynamically entered hosts must be from the same domain.

The examples below show how nsupdate could be used to insert and delete resource records from the `example.com` zone. Notice that the input in each example contains a trailing blank line so that a group of commands are sent as one dynamic update request to the master name server for `example.com`.

```

nsupdate
> update delete oldhost.example.com A
> update add newhost.example.com 86400 A 172.16.1.1
>

```

Any A records for `oldhost.example.com` are deleted.

And an A record for `newhost.example.com` it IP address `172.16.1.1` is added. The newly-added record has a 1 day TTL (86400 seconds)

```

nsupdate
> prereq nxdomain nickname.example.com
> update add nickname.example.com 86400 CNAME \
 somehost.example.com

```

**Note:** If bind is in the order line in `/etc/host.conf` in a server then each new connection might be attempted to reverse resolve the IP of incoming client before continuing.

- **Updating root servers hint file** `root.hint`:

```

cd /var/named

dig @e.root-servers.net . ns > root.hint.new
or
dig @a.root-servers.net . ns > root.hint.new
mv root.hint root.hint.old
mv root.hint.new root.hint
rndc restart

```

A master for one zone can also be a slave of another zone

```
#
#example of slave zone (the master is at 192.168.71.37)
#
Forward slave zone
zone "elop.temp.berlin" in {
 type slave;
 file "slave/elop.temp.berlin.zone";
 masters { 192.168.71.37; };
};

Reverse slave zone
zone "70.168.192.in-addr.arpa" in {
 type slave;
 file "slave/192.168.70.zone";
 masters { 192.168.71.37; };
};

zone "71.168.192.in-addr.arpa" in {
 type slave;
 file "slave/192.168.71.zone";
 masters { 192.168.71.37; };
};
```

**IMPORTANT NOTE:**

- Make sure that the master has the IPs/Addr of all the slaves as NS records in its databases, otherwise the master won't know which slaves should be notified when a change in master occurs.

eg.

```
IN NS slave1.linux.site.
IN NS slave2.linux.site.
```

- Make sure that the master database Serial Number is increased each time you do a change to the master databases.

- **bind4 configuration files (Only as information)**

```
/etc/named.boot
 /var/named/localhost.rev
 /var/named/domainname.zone
 /var/named/domainname.rev (.rev = reverse)
```

- **Troubleshooting**

/var/log/messages shows all messages of bind

## Supporting sub-domains in the same DNS

- Usefull to create multiple Apache VirtualHosts in `laptop.linux.local` web server without any extra DNS entries needed for new virtual hosts.
- Each sub-domain node (sub-domain host) should have its own database file in the DNS and the resolving of its name and all of the sub-domain names will be done by this file and not the the file of the main domain.
- Only if a normal host in the same domain will be resolved by the main domain database file.

**Example:** A main domain (`linux.local`) has 5 hosts of which 1 is a sub-domain node (`laptop.linux.local`).

### part of /etc/named.conf

```

#-- Forward resolution for domain linux.local
zone "linux.local" IN {
 type master;
 file "linux.local.zone";
 notify yes;
};
#-- Forward resolution for subdomain *.laptop.linux.local
zone "laptop.linux.local" IN {
 type master;
 file "laptop.linux.local.zone";
 notify yes;
};
#-- Reverse resolution for 192.168 subnet
#-- Here all the domain hosts are part of the same subnet
#(including the sub-domain node)
zone "168.192.in-addr.arpa" IN {
 type master;
 file "192.168.zone";
 notify yes;
};

; --- forward zone file for domain linux.local -----
; /var/named/linux.local.zone
$TTL 2D
@ IN SOA dozlinux root.localhost (
 2002091201 ; serial
 3H ; refresh
 15M ; retry
 1W ; expiry
 1D) ; minimum
;
laptop.linux.local IN NS dozlinux
laptop.linux.local IN NS dozlinux
irmgardm IN A 192.168.100.166
lapwin2000 IN A 192.168.100.61
idefix IN A 192.168.100.200
proxix IN A 192.168.100.233

```

- Here the address of `laptop.linux.local` is not listed (as `laptop`). It only resolved by its own sub-domain database file shown below.

```

• ; /var/named/laptop.linux.local.zone
;
$TTL 2D
@ IN SOA dozlinux.linux.local root.localhost (
 2002091201 ; serial
 3H ; refresh
 15M ; retry
 1W ; expiry
 1D) ; minimum
;
 IN NS dozlinux.linux.local.
extrapc IN A 192.168.100.120
* IN CNAME @
@ IN A 192.168.100.60

```

-----  
`dozlinux.linux.local.` is the main DNS server where this file resides.

\* means all the names belonging to sub-domain `laptop.linux.local`  
 eg. `my.virtual.host.laptop.linux.local`

- Here (because of the all names called *anything.laptop.linux.local* will be resolved to the IP no. of the sub-domain node(host) `laptop.linux.local`).