Network - Domain Name Service

Michel FACERIAS

13 décembre 2023

Polytech Montpellier

Université de Montpellier

Table des matières

1 Once upon a time, A brief history of Domain Name Services

1.1 From ARPANet to Internet

1.1.1 Concept : ARPANet

 In the late 1960s, the U.S. **Department of Defense's Advanced Research Projects Agency** (ARPA, later **DARPA**), began funding an experimental **wide area computer network** that connected important research organisations in the US, **called the ARPAnet**.

1.1.2 Concept : TCP/IP for all

 The **Transmission Control Protocol/Internet Protocol** (TCP/IP) protocol suite was developed in the early 1980s, and quickly became the **standard host-networking protocol** on the ARPAnet.

The inclusion of the protocol suite in the University of California at Berkeley's popular UNIX operating system (**UNIX BSD**) was instrumental in **democratising internetworking** for more organisations than were previously attached to the ARPAnet.

The network grew from a handful of hosts to a network of tens of thousands of hosts. The original **ARPAnet became the backbone** of a confederation of local and regional networks based on TCP/IP, called **the Internet**.

1.2 From *hosts.txt* to Domain Name Service

1.2.1 Concept : `hosts.txt`, The manual way

 Through the 1970s, the ARPAnet was a **small and friendly community** of a few hundred hosts. A single file, `hosts.txt` (today `/etc/hosts.txt`), contained all the information you needed to know about those hosts : it held a **name-to-address mapping** for **every host connected** to the ARPAnet.

`hosts.txt` was maintained by Network Information Center ("the **NIC**") and distributed from a single host.

Administrators typically emailed their changes to the NIC, and periodically downloaded from the NIC the updated file, once or twice a week.

1.2.2 Concept : Yet, ARPANet grew

 As the ARPAnet grew, however, this scheme became unworkable :

— **Traffic and load** : The toll in terms of the network traffic and processor load involved in distributing the file, was becoming unbearable ;

— **Name collisions** : No two hosts in `hosts.txt` could have the same name. However, while the NIC could assign addresses in a way that guaranteed uniqueness, it had no authority over host names. There was nothing to prevent someone from adding a host with a conflicting name and breaking the whole scheme ;

— **Consistency** : Maintaining consistency of the file across an expanding network became harder and harder.

1.2.3 Concept : Finaly, the Domain Name System arrives

 The ARPAnet's governing bodies chartered an investigation into a successor for *hosts.txt*. Their goal was to create a system that solved the problems inherent in a unified host table system :

— The new system should allow **local administration of data**, yet make that data **globally available** ;
— The decentralisation of administration would **eliminate the single-host bottleneck** and relieve the traffic problem ;
— It should use a **hierarchical name space** to name hosts. This would ensure the **uniqueness of names**.

Paul Mockapetris was responsible for designing the architecture of the new system :

— In 1984, he released **RFC 882** and **RFC 883**, which describe the **Domain Name System** ;
— These RFCs were superseded by **RFC 1034** and **RFC 1035**, the current specifications of the Domain Name.

RFCs 1034 and 1035 have now been **augmented by many other RFCs**, which describe potential DNS security problems, implementation problems, administrative gotchas, mechanisms for dynamically updating name servers and for securing domain data, and more...

 See more : https://www.rfc-editor.org

2 The Domain Name System

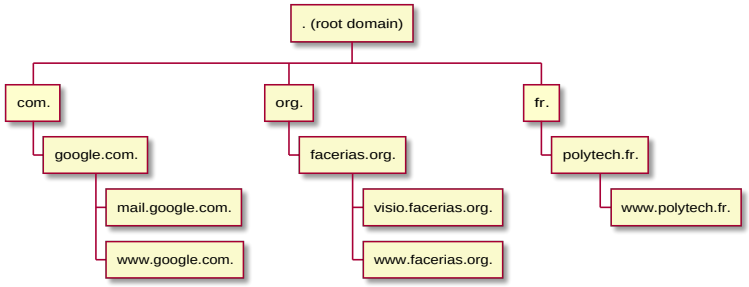
The **Domain Name System** (DNS) is a **distributed database**. This allows local control of the segments of the overall database, yet data in each segment are available across the entire network **through a client-server scheme**. Robustness and adequate performance are achieved through replication and caching.

2.1 How it works

2.1.1 Concept : Hierarchical structure

 The **structure of the DNS database** is very similar to the structure of the UNIX filesystem :

— The whole database is pictured as **an inverted tree**, with the root node at the top ;
— Each node in the tree has **a text label, which identifies the node** relative to its parent ;
— One label (the **null label**, or "") is reserved for the **root node**. In text, the root node is **written as a single dot** (".").



Programs called **name servers** constitute the **server half** of **DNS's client-server** mechanism. **Clients** are called **resolvers**.

2.1.2 Concept : Top-Level Domains

 It exists a few domain names that are **just under the root**. They are named **Top-Level Domains** (TLDs).

There is three classes of TLDs :

— Geographical 2 letters TLDs : eu, fr, sp, us ...

— Organisational 3 letters TLDs : com, org, net, gov, ...

— "Dark side" TLDs : info, online, life, store, tech, game, ...

It costs about 25k€ to record a TLD for the firts time and 25k€ of annual fee !

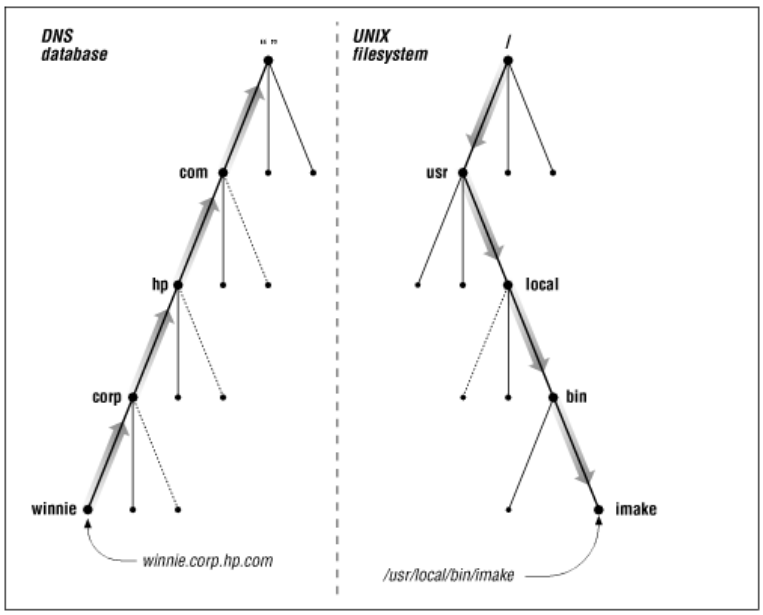
2.1.3 Concept : DNS namespace

 **Each node** is also the **root of a new subtree** of the overall tree. Each of these subtrees represents a partition of the overall database : **a domain** in the Domain Name System.

Each domain can be further **divided into** additional partitions, called **subdomains** in DNS. Subdomains are drawn as children of their parent domains.

Every domain has a unique name. A domain's domain name identifies its position in the database.

In DNS, the **domain name is the sequence of labels** from the node at the root of the domain to the root of the whole tree, **with "." separating the labels**.



⚠ **In DNS, names are read from bottom to top**, while in a filesystem, they are read from top to bottom

2.1.4 Concept : FQDN

 A **Fully Qualified Domain Name** is an unambiguous name, which **uses the entire namespace, up to the root**.

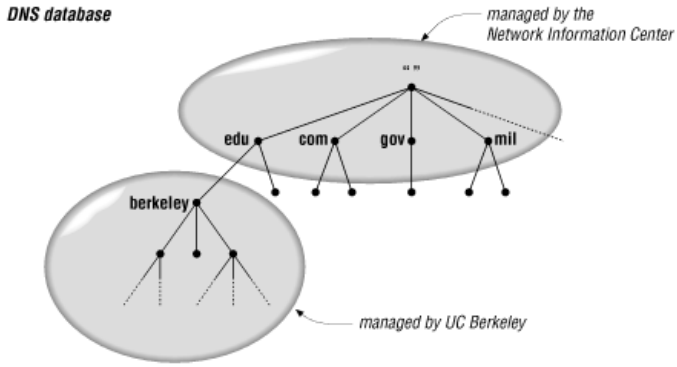
According to the last figure, *winnie.corp.hp.com* is *winnie*'s **FQDN**. It **should be written** *winnie.corp.hp.com.* (with the final dot) to mark the top of the namespace, but **this is rarely the case**.

2.1.5 Concept : Delegation of administration

In DNS, **each domain can be administered by a different organisation**.

Each organisation can then **break its domain into a number of subdomains** and dole out responsibility for those subdomains to other organisations.

For example, the InterNIC runs the *edu* domain, but assigns U.C. Berkeley authority over the *berkeley.edu* subdomain.



2.1.6 Concept : Domain vs Zones

The difference between a zone and a domain is important.

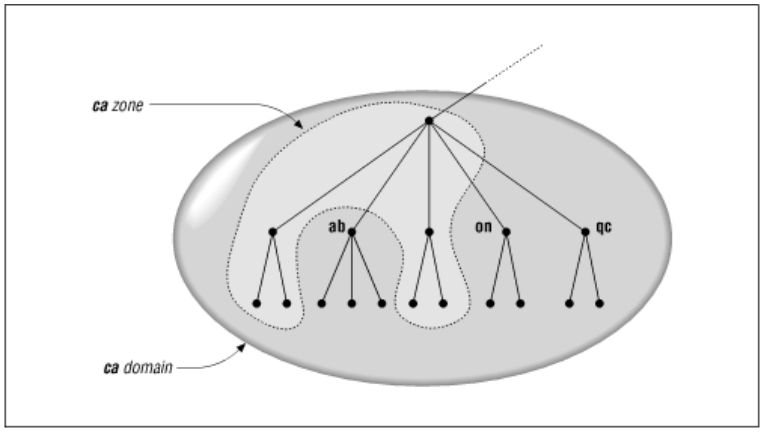
All top-level domains, and many domains at the second level and lower, like *berkeley.edu* and *hp.com*, **are broken into smaller**, more manageable units by delegation. **These units are called zones**.

For example, the top-level domain *ca* (for Canada) may have the subdomains *ab.ca*, *on.ca*, and *qc.ca*, for the provinces Alberta, Ontario, and Quebec.

Authority for the *ab.ca*, *on.ca*, and *qc.ca* domains **may be delegated** to people in each of the provinces.

The domain *ca* contains all the data in *ca* plus all the data in *ab.ca*, *on.ca*, and *qc.ca*.

But the zone *ca* contains only the data in *ca*, which is probably mostly pointers to the delegated subdomains.



Then :

— *www.ca* is managed in the *ca* zone ;

— *www.edu.ca* is managed in the *ca* zone too (there's no *edu.ca* delegation) ;

— *www.ab.ca* is not managed in the *ca* zone, but in a *ab.ca* zone.

— *www.qc.ca* is in the same situation...

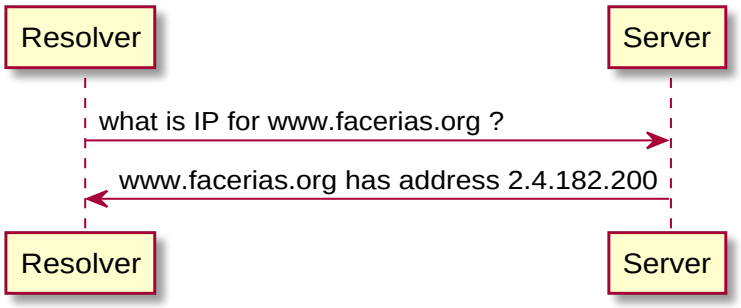
Each zone should be held by a different configuration file, on a same server or on different servers.

2.2 Clients

Clients are named **Resolvers**. It's often just library routines that **create queries** and send them across a network to a name server.

2.2.1 Concept : Recursive query

When a resolver make a **recursive query**, it wants a **full response**.



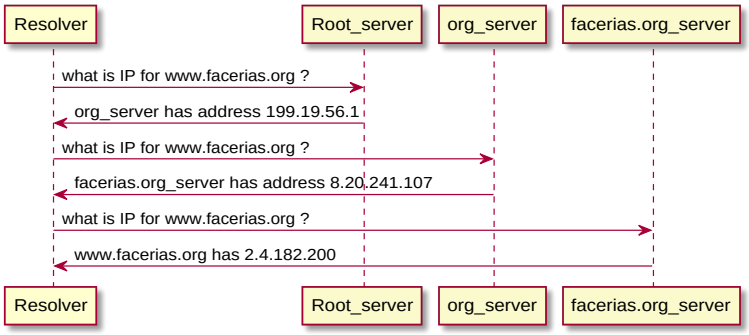
Yet, there are two possibilities :

— The server has the solution in his database, it answers directly ;

— The server hasn't the solution, it acts himself as an resolver to find the answer.

2.2.2 Concept : Iterative query

When a resolver make an **iterative query**, it accepts a **partial response**, mostly the address of an other server.



2.3 Servers

2.3.1 Concept : Authoritative Servers

Authoritative servers contain **information about some segment** of the database and make it available to **clients**.

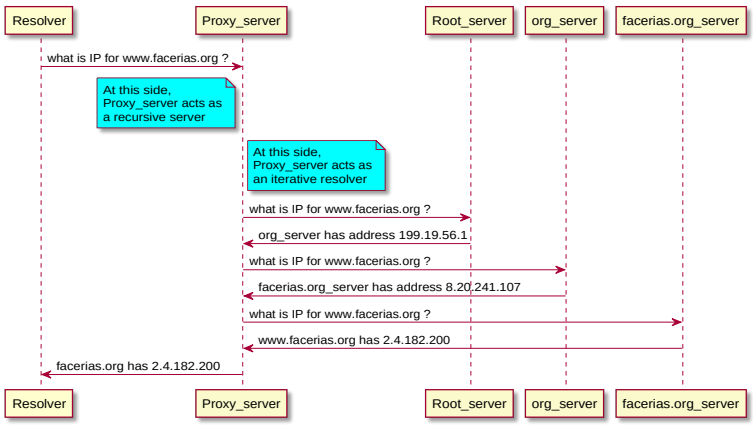
They are **able to answer recursive queries on their own zones**.
There is two type of authoritative servers :
— **primary** servers : on which database is manually updated ;
— **secondary** servers : on which database is updated from a primary server.

2.3.2 Concept : Forwarder servers

 **Forwarding** is a particular way to **deal with some domains**. It consist to send the query **to a server known to be authoritative** on this domain.
This hack is mostly used to find internal mail server (outgoing mail servers).

2.3.3 Concept : Proxy servers

 In a LAN, it should be a nice idea to have some **proxy services. DNS** is one of them.
The DNS proxy server should propose one or more role :
— a **DNS for local resources**, using a fake domain (not a child of TLD, like .home for internet boxes) or a real one (i.e. an internal view of your enterprise domain), with authoritative role ;
— a **recursive resolution** of all **other domains** ;
— a **forward** to your ISP domain ;
— a **global invisible redirection** to it, using a *DNAT iptables rule*, to avoid the use of an other DNS (aka 8.8.8.8, for **security or functional reason**).
The next figure shows how a proxy server acts for recursive queries.



Here, the proxy server acts as recursive for the left resolver. It acts as iterative resolver at its right, querying first a root server.

3 Tools to manage DNS

3.1 `host`, a CLI resolver

According to `man host`, it is a simple utility for **performing DNS lookups**.
Giving a server is an optional argument which can be either the name or IP address of the name server that host should query instead of the servers listed in `/etc/resolv.conf`.

3.1.1 To do : Read the doc !

Use a console, and have a look to :

— `man hosts`

— `man resolv.conf`

3.1.2 To do : Check your `/etc/resolv.conf`

Use a console to show the content of this file :

```
$ cat /etc/resolv.conf
search facerias.org
nameserver 192.168.100.200
```

3.1.3 Question : Name servers

What are the name servers used by my computer ?

3.1.4 Question : Final dot

Explain the results of these two querries :

```
$ host hal9000.facerias.org.
hal9000.facerias.org has address 192.168.254.200

$ host hal9000.facerias.org
hal9000.facerias.org has address 192.168.254.200
```

3.1.5 Question : Search order

In the same context, try to explain the result of these two other queries :

```
$ host hal9000.facerias.org
hal9000.facerias.org has address 192.168.254.200

$ host hal9000
hal9000.facerias.org has address 192.168.254.200
```

3.1.6 Question : Final dot, again

Allways in the same context, try to explain the result of these two other querries :

```
$ host hal9000
hal9000.facerias.org has address 192.168.254.200

$ host hal9000.
Host hal9000. not found: 3(NXDOMAIN)
```

3.1.7 To do : Try on your computer

 Try, using host, some queries, and modify your `/etc/resolv.conf` to search in *polytech.fr* and *dopolytech.fr* for incomplete names.

 Auto-configuration of your computer is going to fix `/etc/resolv.conf`. So you need to modify again !

3.1.8 Concept : Name service order

 There are **other name service providers**. At this time, you know :

— `/etc/hosts`
— DNS

The `host` **tool only queries the DNS**.

If you want to query `/etc/hosts`, you mus use a trick by hijacking an an other tool. `ping` perform queries, using `/etc/hosts`, then DNS in case of failure.

3.2 `bind`, a name server

The **first implementation** of the Domain Name System **was called *jeeves***, written by Paul Mockapetris himself.

A **later implementation was *bind***, written for Berkeley's BSD UNIX operating system by Kevin Dunlap.

bind is now maintained by the **Internet Software Consortium**.

 See more : http://www.isc.org/bind.html.

 *bind* is the name of the server and */etc/bind* is the configuration directory. But the daemon is called *named* !

3.2.1 To do : Install *bind*

 Create a virtual machine (or make a clone from a template).

Edit the network config file to staticaly address your VM.

According to sysadmin4 chapter, your sandbox network is 192.168.122.0/24, dynamics addresses offered to VM is 192.168.122.1 to 192.168.122.199 and the gateway is on 192.168.122.254, use these values :

```
$ cat /etc/network/interfaces
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

auto ens0 #this is my interface to the sandbox network
iface ens0 inet static
  address 192.168.122.200/24
  gateway 192.168.122.254
```

Change `/etc/hosts` and `/etc/hostname` to modify the name of your VM and use *myserver*. reboot your VM to verify that everithing is ok.

After this reboot, use `apt` :

```
# apt update \&\& apt upgrade
...

# apt install bind9
Lecture des listes de paquets... Fait
...
Les paquets supplementaires suivants seront installes
  bind9-utils dns-root-data python3-ply
Paquets suggrs
  bind-doc dnsutils resolvconf ufw python-ply-doc
Les NOUVEAUX paquets suivants seront installes
          bind9 bind9-utils dns-root-data python3-ply
...
```

Then, verify if `named` is running, and offering sockets.

```
# ps auxf
...
bind   13848  1.0  1.4 723224 28812 ?  Ssl  16:44   0:00 /usr/sbin/named -f -u bind

# # ss -lutpn
...
udp      UNCONN    0    0    192.168.122.200:53    0.0.0.0:*        ...
...
udp      UNCONN    0    0          127.0.0.1:53    0.0.0.0:*        ...
...
tcp      LISTEN    0   10    192.168.122.200:53    0.0.0.0:*        ...
...
tcp      LISTEN    0   10          127.0.0.1:53    0.0.0.0:*        ...
...
```

So, `named` seems to be running, waiting even on UDP 53 and TCP 53

3.2.2 To do : Make a first test

Use a console on you computer to querry the bind instance seated in your VM :

```
$ host hal9000.facerias.org 192.168.122.200
Using domain server:
Name: 192.168.122.200
Address: 192.168.122.200#53
Aliases:

hal9000.facerias.org has address 90.51.213.109
```

3.2.3 Question : Explain the query

In the query above, what are the second and the third argument ?

3.2.4 Question : Transport protocol

What is the transport protocol used by the host query ?
Use a `wirshark` cature to find the answer.

3.2.5 Question : Default configuration

 How does bind answer the query ?

Have a look in `/etc/bind`. Start reading `named.conf` file. Then search the way the server can answer queries out of the box.

3.2.6 Question : European root server

 Look at `/usr/share/dns/root.hints` and find how many root servers are in Europe ?

3.2.7 To do : create your own zone

 You are going to create a dummy zone named *dummy.fr*.

Create and edit a file named `/etc/bind/db.dummy.fr`.

Here is an exemple :

```
# cat /etc/bind/db.facerias.org
$ORIGIN .
$TTL 86400
facerias.org  IN      SOA     hal9000.facerias.org facerias.hal9000.facerias.org. (
                2022021401      ;serial : AAMMJJXX
                3600            ; Rafraichissement apres 1 heure
                600                     ; Nouvel essai apres 10 minutes
                3600000                 ; Obsolete apres 1000 heures (>41 jours)
                86400 )                 ; TTL minimum de 1 jour

                IN    NS        hal9000.facerias.org

$ORIGIN facerias.org.

;dhcp inconnus du domaine interne
intra1          IN      A       192.168.254.1
intra2          IN      A       192.168.254.2
intra3          IN      A       192.168.254.3
...
```

The third line is the **start of authority**, and should be understood like this :

— *facerias.org* is the domain ;

— *hal9000.facerias.org* is the authoritative server ;

— *facerias.hal9000.facerias.org* is a mail address, formerly *facerias@hal9000...*

The line *IN NS* declare the only server of the domain.

The lines *IN A* declare direct resolutions for *intra1.facerias.org* and so on !

 Don't forget to include `/etc/bind/db.dummy.fr` in the configuration chain !

3.2.8 To do : Final tests

 Fill in your zone file, and make some tests.