UNIT 2.
SERVICES INVOLVED IN WEB DEPLOYMENT

Web Applications Deployment
CFGS DAW

Author: Carlos Cacho López

Reviewed by: Lionel Tarazón Alcocer
lionel.tarazon@ceedcv.es

2019/2020

License

Nomenclature

During this unit we are going to use special symbols to distinct some important elements. This symbols are:

INDEX

U02. SERVICES INVOLVED IN WEB DEPLOYMENT

1. DNS

In Internet data packets travel using the Internet Protocol (IP). This protocol, either in version 4 or version 6, works using IP addresses, that is, four 8-bit numbers (from 0 to 255) that identify a device in the network. For example: 84.15.20.210

> The **Internet Protocol** is responsible for addressing hosts and for routing packets from a source host to a destination host across one or more IP networks. For this purpose, the Internet Protocol defines the format of packets and provides an addressing system that has two functions: Identifying hosts and providing a logical location service.

However, everybody is used to website names to identify hosts, such as *www.ceedcv.org*, *www.google.es*, etc.

> A **host** is a computer or other device connected to a network.

> You can access a website using its name or IP address. Open your internet browser and type *173.194.78.94 (it's* www.google.es *!!)*

Website names, easy to remember, have to be changed to IP addresses to know what the destination machine is. To do this, the **Domain Name System** (DNS) was created. So, a DNS server is a service to convert the name of any machine to its IP address and vice versa.

The **advantages** of using DNS are:

1. Name duplication problems disappear: one administrator controls the existing domains. There can be repeated names but in different domains.
2. Information consistency: distributed information is updated automatically.

It is important to know the **terminology** used:

- **hostname**: name of a host (computer) in a domain or network. For example: linuxserver, drive, mail, ceice, etc.
- **domain name**: collection of hostnames linked by dots. For example: ceedcv.es, google.com, gva.es
- **fully qualified domain name (FQDN)**: complete name of a host, composed by *hostname . domain name* For ex.: linuxserver.ceedcv.es, drive.google.com, mail.google.com, ceice.gva.es, etc.
- **top level domains (TLD)**: the highest level in the domain hierarchy. For example: com, es, org, etc.

1.1 Domain names

There are 2 types of domain names:

- **Flat names**: they do not have hierarchy. This system allows to classify only one name inside a category. For example, the teacher's name (*carlos*), but we cannot know his department.
- **Hierarchic names**: they use a hierarchy when building the name. For instance, when we want to create the name of a computer's teacher, we can add more information. In the previous example, we can add the department's teacher, the center and the country: *carlos.informatica.ceedcv.es*. This is used in Internet.

The domain names have some restrictions:

- They only can be made up of letters (English alphabet), numbers and dashes ("-").
- They can not start or end by a dash.
- Their longitude have to be minor than 63 characters (including the extension).

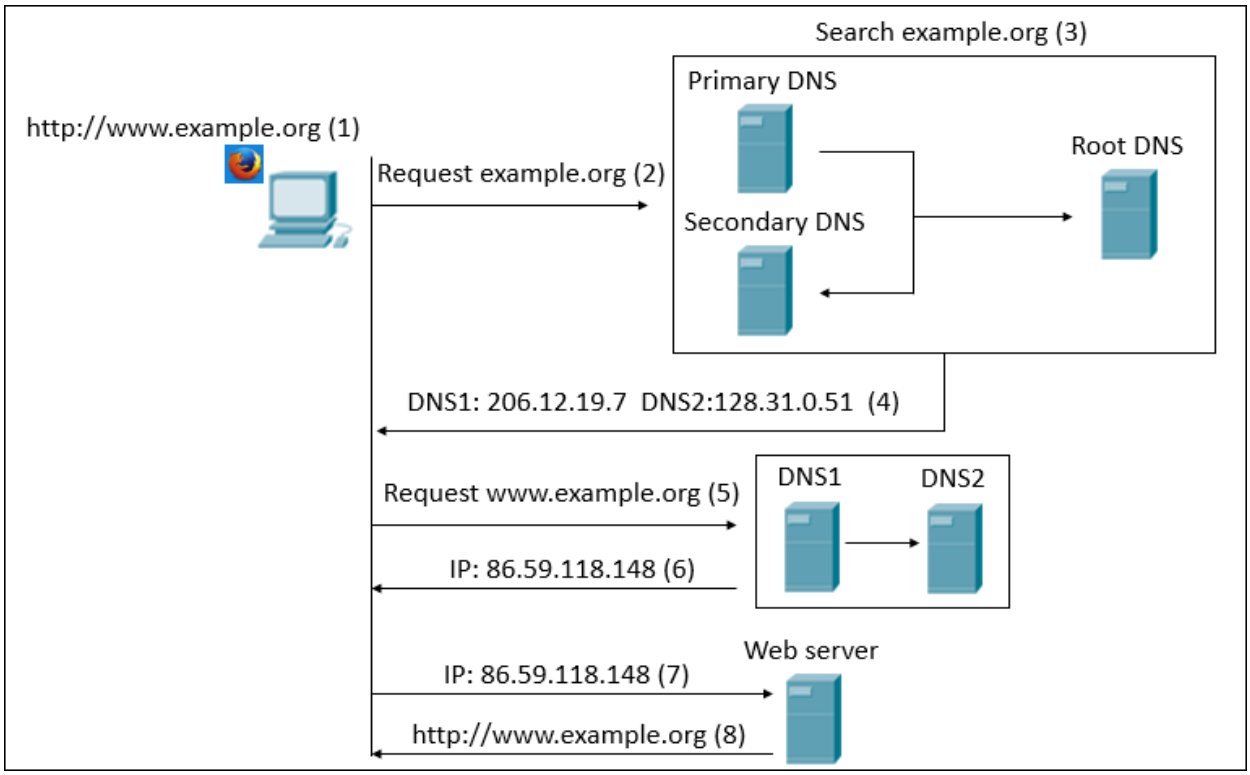
Nowadays, there are "multilingual domains" that allow us to use non-english symbols (ñ, ç...) and other alphabets (Arabic, Chinese, etc.).

1.2 DNS function

1.2.3 Forward Lookup

Let's see how this service works by using an example: *http://www.example.org/*

1. In your browser write the URL: *http://www.example.org/*
2. The browser looks for the DNS information of *example.org*
3. If you do not have the information in your computer, it will search in DNS servers you have in your network configuration. They are usually given by your **ISP** (Internet Service Provider such as Vodafone, SomConnexió, etc.). If the information neither is in that servers, it will be in the **DNS root**.
4. The information searched (IP addresses of the DNS servers of *example.org*) arrives to your computer (DNS1: 206.12.19.7 and DNS2:128.31.0.51). It is recommendable to have two DNS servers for each zone.
5. Now, your computer can connect with DNS1 server, or, if there is any connection problem with it, with DNS2 server. In this DNS servers are all the information to find the website.
6. Your computer receives the IP address of the web server where the web page is.
7. Your computer talks to the web server and searches the web page there.
8. Finally, the web server returns the information and the browser shows in the screen.



1.2.1 Reverse lookup

There is the inverse process, **Reverse lookup**, which uses a reverse DNS. It allows to obtain the domain names associated to an IP address. The requests follow a similar process than the one explained before, but they use a specific domain (**ip_addr.arpa** for IPv4 and **ip6.arpa** for IPv6).

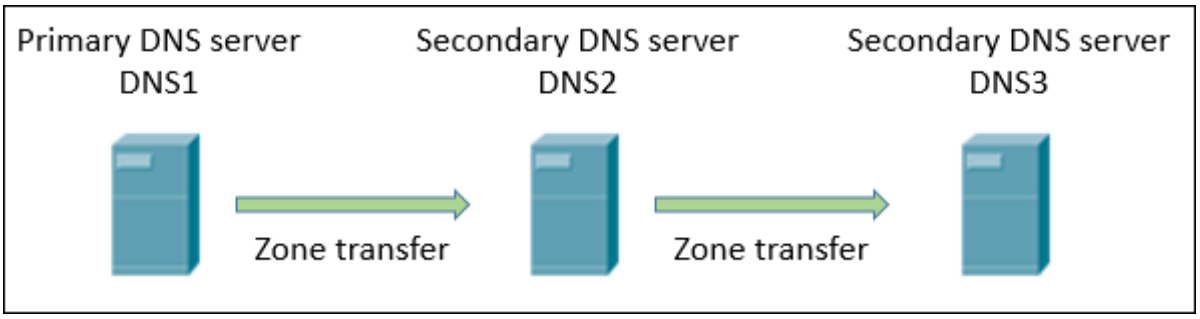
The IP addresses are represented in an inverted way, as if they were a domain. For instance, the IP *123.38.67.45* would be represented as *45.67.38.123.in-addr.arpa*. So, first it will ask for who is associated to 123, and then go on.

Inverse Zones are created for this process, including SOA, NS, PTR and CNAME records.

1.3 Types of DNS servers

The diferent types of DNS servers are:

- **Primary name server**: these servers store the information of their zone in a local database. They are responsible of keeping updated the information and any change has to be notified to them.
- **Secondary name servers**: these servers are also known as **slave** name servers, but at the same time they can be **masters** of other secondary name servers. They get the information of their zone from other authorized server in that zone. The process of copying information is known as "**zone transfer**".
- **Master name server**: the master server transfers zones to the secondary servers. When a secondary server starts it tries to find a master server and do the zone transfer. A master server can be at the same time primary or secondary for that zone. In this way, we can avoid that secondary servers overload primary servers with zone transfers. For example, in the picture we can see that the DNS3 server requests the zone to DNS2 server, not to DNS1.
- **Caching-only server**: this kind of servers do not have authority in any domain, they only contact with other servers to resolve the DNS client requests. This server has a **cache memory** with the last requests answered.



Every time a DNS client makes a request to these servers, they first check in their cache memory. If they find the IP address requested they send it to the client, otherwise, they look for it in other servers, copy the answer in their cache memory and send the answer to the client.

1.4 DNS architecture

The system is structured as a **tree**. Every node of the tree is composed of a group of servers who take care of solving a set of domains (**zone of authority**). A server can delegate in other one (or others) the authority on someone of his sub-fields (this is, some sub-domain of the zone on which it has authority). A **sub-domain** can turn as a specialization of a domain of previous level. For example, *mail.google.es* is a sub-domain of *google.es*, and *google.es* is a subdomain of *es*.

Zone Files contain information about a particular name space. There are systems in which each zone file is named similar to the domain. An example is *example.com.zone* where we can see the domain at the name of the zone file.

Each zone file contains directives and resource records. The first ones, directives, ask the DNS server to do tasks or to apply special configurations to the zone. The second one, resource records, defines the zone parameters and assigns identities to the individual hosts. The directives are optional, but not the resource records. They are needed to offer name services to the zone.

So, when we configure a domain in a DNS server, it has to belong to a zone. In the zone files we should include the IP of the web service (www), the IP of the mail service, etc.

There are two kinds of zone files:

- **Forward lookup zone**: the resolutions of this zone return the IP address of the requested resource.
- **Reverse lookup zone**: the resolutions of this zone looks for the machine name (hostname) of an IP address.

1.5 Types of DNS records

A DNS database is composed of one or more zone files used by the DNS server. As we have seen before, each zone has a set of resource record.

The DNS record or resource record describe the name of the domain and subdomains, the IP address and the services or resources that it has.

All the resource record have the same format:

Field	Description
Owner	Indicates the DNS domain name that has the resource record. It is an obligatory field.
Time to live (TTL)	Indicates the time of life of a reource record in cache memory. It is optional.
Class	Indicates the class of the resource record. For example, IN means that a resource record belongs to the class “Internet”. It is obligatory.
Type	Indicates the type of the resource record. For example, A means that a resource record stores host addresses information. It is obligatory.
Record specific data	Indicates a descripction of the resource. It is obligatory.

We can see the most used types of DNS records:

Record	Description	Syntax and example
A	[Address] Used to translate hostnames to IPv4 addresses.	owner class TTL A IP_version4 ex: host1.example.com IN A 127.0.0.1
AAAA	[Address] Used to translate hostnames to IPv6 addresses.	owner class TTL A IP_version4 ex: host1ipv6.example.com IN AAAA 1234:0:1:2:3:4:567:89ab
CNAME	[Canonical Name] Usedtocreateadditional hostnames (alias). The hostname have to be defined previously as A type.	owner ttl class CNAME Canonical_name ex: aliasname.example.com CNAME realname.example.com
NS	[Name Server] Used to show the names of the DNS server with total authority over a domain. Each domain can be associated to more than one DNS server.	owner TTL IN NS name_DNSS ex: example.com. IN NS Server1name.example.com
MX	[Mail eXchange] Associates a domain name with a list of servers of email interchanging for the domain	owner TTL class MX preferency hostInterchargerOfEmail ex: example.com. IN MX 10 mailserver1.example.com
PTR	[PoinTeR] Translates IP address to domain names (reverse record)	owner TTL class PTR DestinyDomainName ex: 1.0.0.10.in-addr.arpa. IN PTR host.example.com
SOA	[Start Of Authority] It gives information about the primary DNS server of the zone	owner class SOA NameServer ResponsiblePerson (SerialNumber UpdateInterval RetryInterval Expiration MinTTL) ex: @ IN SOA ServerName.example.com postmaster.example.com(1; serial number 3600; update [1h] 600; retry [10m] 86400; expiration [1d] 3600); minimum TTL [1h]
TXT	[TeXT] It allows the domain to identify themselves in an arbitrary way	owner TTL class TXT text_Chain ex: example.com TXT “Example of information of additional domain name”
SPF	[Sender Policy Framework] Used to avoid sending phishing emails. It specifies with hosts are allowed to send emails from the domain.	owner TTL class IN SPF text_Chain ex:example.comINSPF“v=spf1 a:mail.example.com –all”

1.6 Root DNS server

The Internet Corporation for Assigned Names and Numbers (**ICANN**) is the non-profit organization, created in 1998, responsible for IP address managing, the protocol identifiers, the domain system managing and the root servers managing.

ICANN is responsible for the coordination of the administration of the technical elements of the DNS. It guarantees a univocal resolution of the names, so the Internet user could find all the valid addresses. Therefoew, it is in charge of supervising the distribution of the specific technical identifiers used in the Internet operations, and of delegating the domain names of first level, like .com, .info, etc.

There are 13 addresses where we can find the **root servers** (they can be duplicated in many places). All these servers store a copy of the same file: the main index of the Internet address agenda. They have an address for each domain of the main level (.com, .es...)

The root servers are not consulted really often, because once the network machines know the address of a main domain level, they can keep it and they only check it sometimes. Anyway, the root servers are essential.

The names of the 13 servers are: A.ROOT-SERVERS.NET, B.ROOT-SERVERS.NET, ..., M.ROOT-SERVERS.NET. There are 9 organizations in charge of them (universities, enterprises, etc.)

1.7 DNS Tools

There are some software tools to test how our DNS works. Some of them are:

- **dig**: it resolves names and IP addresses very detailed. It offers information about the records and the wait time of the DNSS.
- **nslookup**: it resolves names and IP addresses not detailed.
- **ping**: it allows to test if DNS is working
- **whois**: it shows the information of the domain's owner.
- **Traceroute/tracert**: it shows how many jumps are needed to reach a domain

2. FTP PROTOCOL

The **File Transfer Protocol** (**FTP**) is designed to facilitate bi-directional transfer of files and records between hosts on a TCP/IP network.

The client establishes a **Control Connection** for the duration of an FTP session that typically spans multiple **data** transfers. FTP uses a separate TCP connection for data transfer.

Commands are issued and acknowledged over the **Control Connection**, a TCP connection to well-known port 21.

If the user issues a command that requires a response more elaborate than a one-line response code, a **Data Connection** is established between the client and the server. The response data—the contents of a file or a directory listing—is sent over that data connection. The port number for data connections, and the way in which data connections are initialized, vary depending upon whether the client requests the data in *active* or *passive* mode.

2.1 Active mode

Active mode is the original method used by the FTP protocol for transferring data to the client application. When an active mode data transfer is initiated by the FTP client, the server opens a connection from port 20 on the server to the IP address and a random, unprivileged port (greater than 1024) specified by the client. This arrangement means that the client machine must be allowed to accept connections over any port above 1024.

2.2 Passive mode

Passive mode, like active mode, is initiated by the FTP client application. When requesting data from the server, the FTP client indicates it wants to access the data in passive mode and the server provides the IP address and a random, unprivileged port (greater than 1024) on the server. The client then connects to that port on the server to download the requested information.

3. SSH PROTOCOL

SSH (or *S*ecure *SH*ell) is a protocol which facilitates secure communications between two systems using a client/server architecture and allows users to log into server host systems remotely.

Unlike other remote communication protocols, such as FTP or Telnet, SSH **encrypts** the login session, making it impossible for intruders to collect unencrypted passwords.

SSH is designed to replace older, less secure terminal applications used to log into remote hosts, such as telnet or rsh.

The SSH protocol provides the following safeguards:

- After an initial connection, the client can verify that it is connecting to the same server it had connected to previously.
- The client transmits its authentication information to the server using strong, 128-bit encryption.
- All data sent and received during a session is transferred using 128-bit encryption, making intercepted transmissions extremely difficult to decrypt and read.
- The client can forward X11 applications from the server. This technique, called ***X11 forwarding***, provides a secure means to use graphical applications over a network.

Because the SSH protocol encrypts everything it sends and receives, it can be used to secure otherwise insecure protocols.

4. BIBLIOGRAPHY

[1] Álvarez Villanueva, Cristina (2015): “Network services involved in the web application deployment. Domain name servers”, CEEDCV

[2] Cuesta, Sergio (2014): *Despliegue de aplicaciones web*

[3] https://httpd.apache.org

[4] *http://web.mit.edu*