6.3. Before DNS

Before **DNS**, there was **ARPANET**. The original solution to a name service was a flat text file called `HOSTS.TXT`. This file was hosted on a single machine, and, when you wanted to get a copy, you pulled it from this central server using **FTP** (**F**ile **T**ransfer **P**rotocol) or a similar protocol.

This did not scale well at all.

6.4. /etc/hosts

A descendant of the **HOSTS.TXT** file is the **hosts** file. On **Linux**, this file resides at **/etc/hosts**. It has a very simple syntax:

```
<IP ADDRESS> <HOSTNAME> [HOSTNAME or alias] ...
```

An example of an **/etc/hosts** file is the following:

```
127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4 ::1 localhost
localhost.localdomain localhost6 localhost6.localdomain6
10.100.42.4 laser4 laser4.example.com
10.100.42.1 mail mail.example.com
```

This **hosts** file usually takes precedence over other resolution methods.

6.5. DNS Basics

DNS is a distributed, hierarchical key-value database.

When a network node makes a **DNS** query, it most often makes that query against a recursive, caching server. That recursive, caching server will then make a recursive query through the **DNS** database, until it comes to an authoritative server. The authoritative server will then send the answer for the query.

The **DNS** database consists of a tree-like, key-value store. The database is broken into tree nodes called **Domains**. These **domains** are managed as part of a **zone**. **Zones** are the area of the namespace managed by authoritative server(s). The **DNS** delegation is done on zone boundaries.

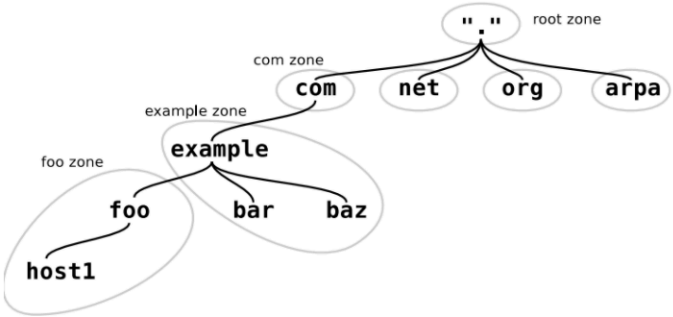

Figure 6.1: The DNS Tree

A theoretical recursive **DNS** query for **`host1.foo.example.com.`** would take the following steps:

1. The Client makes a recursive request to the caching nameserver it has configured.
2. The caching nameserver makes a query for **`host1.foo.example.com.`** to the root (".") zone nameserver.
3. The root (".") zone nameserver points to the nameserver for the **`com.`** zone.
4. The caching nameserver makes a query for **`host1.foo.example.com.`** to the nameserver for the **`com.`** zone.
5. The **`com.`** zone nameserver sends a response that points to the nameserver for the **`example.com.`** zone.
6. The caching nameserver makes a query for **`host1.foo.example.com.`** to the nameserver for the **`example.com.`** zone.
7. The **`example.com.`** nameserver sends a response that points to the nameserver for the **`foo.example.com.`** zone.
8. The caching nameserver makes a query for **`host1.foo.example.com.`** to the nameserver for the **`foo.example.com.`** zone.
9. The **`foo.example.com.`** nameserver responds authoritatively for the address **`host1.foo.example.com.`** to the caching nameserver.

caching nameserver.

10. The caching nameserver stores all of these queries and their responses in a cache and responds back to the client with the answer to the original query.

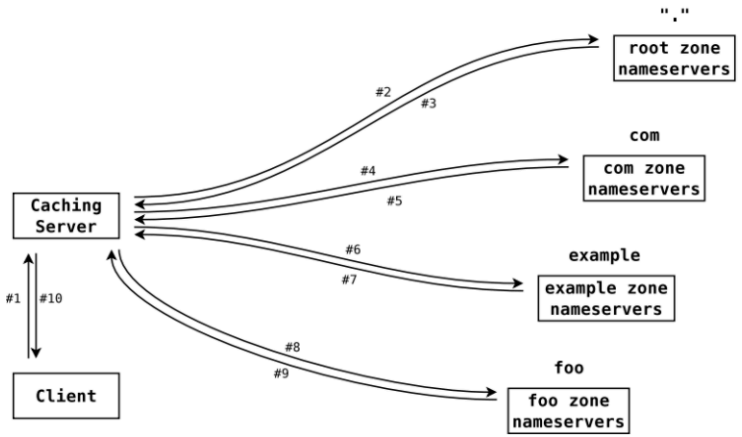

Figure 6.2: A DNS Query

6.9. Forward DNS Queries

Forward **DNS** queries use **`A`** or **`AAAA`** record types and are most often used to turn a **DNS** name into an **IP** address.

A **Fully Qualified Domain Name** (**FQDN**) is the full **DNS** address in the **DNS** database. In the **FQDN** below, the most significant part is the hostname – **`foo:`**

```
foo.example.com
```

- Comment marker - ; to end of line.

Below you can see an example of a zone file:

```
$TTL 60
@ IN SOA  ns1.example.com. admin.example.com. ( 2012012301 ; serial
3H        ; refresh
1H        ; retry
2H        ; expire
1M)       ; neg ttl  IN NS ns1.example.com.; IN A 192.168.23.43
IN AAAA 2001:500:88:200::10
;generate host1 thru host10
$GENERATE 1-10 host$    A 127.0.0.$
```

We would verify the above zone file by using the following command:

```
# named-checkzone example.com. /var/named/chroot/var/named/example.com.zone
```

The **@** character expands to whatever the zone name is.

This example is using the **CentOS** default-named **chroot** option for file location.

The **SOA** (**S**tart **o**f **A**uthority) is required for every zone. Special control fields include:

- Admin email.
- Primary name server.
- Serial number.
- Timers (Slave server **`refresh`** settings).
  - **`Refresh`**: How often to check for new serial from master.
  - **`Retry`**: How often to retry if no response from master.
  - **`Expire`**: How long to keep returning authoritative answers when we cannot reach the master server.
  - **`Negative TTL`**: How long to cache a **NX domain** answer.

The serial number is one of the more important things in the **SOA** record. It should increment every time you make a change to your zone file.

One of the ways to ensure your serial number is updated properly is to use the following format:

```
YYYYMMDDXX
```

where **`XX`** is a two-digit revision number.

The serial number is one of the more important things in the **SOA** record. It should increment every time you make a change to your zone file.

One of the ways to ensure your serial number is updated properly is to use the following format:

`YYYYMMDDXX`

where **XX** is a two-digit revision number.

The negative TTL (part of the **SOA** record) is the time your caching nameserver will hold onto a "this record does not exist" answer. To set the default TTL use the **$TTL** line at the beginning of the line.

A reverse **DNS** query is used to turn an **IP** address into a **DNS** name. It uses a `PTR` record type and an `arpa.` domain in a **DNS** database:

- `in-addr.arpa.` domain for **IPv4**.
- `ip6.arpa.` domain for **IPv6**.

In an **IP** address, the most significant part is on the right. Because an **FQDN** is left-to-right most significant, and an **IP** address is right-to-left, we have to translate an **IP** address to put it into the **DNS** database. For example:

`192.168.13.32`

becomes

`32.13.168.192.in-addr.arpa.`

and

`2001:500:88:200::10`

becomes

`0.1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.2.0.8.8.0.0.0.0.5.0.1.0.0.2.ip6.arpa.`

6.11. DNS Server Daemons

DNS server daemons respond to information requests from clients. These server daemons include:

- **djbdns.**
- **Microsoft DNS** server.
- **BIND** (**B**erkeley **I**nternet **N**ame **D**omain).
  - De facto standard **DNS** service.
  - Current version: **BIND9**.

BIND is a widely-used, **ISC** standard **DNS Internet** software. It is available for most **UNIX**-type systems.

BIND4 and **BIND8** are long-obsolete:

- Both have had many security problems.
- Their use is strongly discouraged.

BIND9 was a ground-up rewrite:

- Security issues were fixed.
- New features were enabled:
  - It can be run inside of a **chroot**.
  - **DNSSEC** - security extensions.
  - It allows for different views - different **DNS** answers, depending on the source **IP** address.
  - It has **IPv6** support.
  - It includes the **rndc** (**R**emote **N**ame **D**aemon **C**ontrol) utility.

6.13. BIND Configuration File

Settings for the **BIND** configuration file are stored in the `/etc/named.conf` file on **CentOS** and **OpenSUSE** and in the `/etc/bind/named.conf` file on **Ubuntu**. The syntax is in the **`named.conf(5)` man** page.

BIND uses three styles of comments:

- **C** style - `/*` comment here `*/` .
- **C++** style - `//` to end of line.
- **UNIX** style - `#` to end of line.

To use **BIND** as a caching nameserver, use the following steps:

1. Install the package.

- **On CentOS:**
  ```
  # yum install bind
  ```
- **On OpenSUSE**:
  ```
  # zypper install bind
  ```
- **On Ubuntu:**
  ```
  # apt-get install bind9
  ```

2. Change the configuration to **listen** on a public interface.

3. Start the **BIND** daemon **named**.

The configuration option to change the **listen** address is `listen-on` or `listen-on-v6`.

6.17. Zone Files

Authoritative zones are defined and must contain an **SOA** (**S**tart **o**f **A**uthority) record. Zone files should contain an **NS** (**n**ame**s**erver) record.

Some of the syntax considerations include:

- The record format is: "**{name} {ttl} {class} {type} {data}**".
- Always use trailing "." on all fully-qualified domain names.
- **@** special character.
- **GENERATE** syntax.
- Comment marker - ; to end of line.

Below you can see an example of a zone file:

```
$TTL 60
@ IN SOA  ns1.example.com. admin.example.com. ( 2012012301 ; serial
3H        ; refresh
1H        ; retry
2H        ; expire
1M)       ; neg ttl  IN NS ns1.example.com.; IN A 192.168.23.43
IN AAAA 2001:500:88:200::10
;generate host1 thru host10
```