Lab III: Domain Name System (DNS)

Objectives

- To understand the basic concept of DNS, relationship between hostname and IP address.
- To configure a simple DNS server.
- To understand DNS resolving protocol.

Readings

1. Tutorial on DNS, for example,
   1.1. http://www.freeos.com/articles/3956/
   1.2. http://www2.rad.com/networks/2002/dns/index.htm
   1.3. https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/5/html/Deployment_Guide/ch-bind.html
   1.4. https://www.apnic.net/publications/media-library/documents/resource-guidelines/reverse-zones

Equipment List

Equipment	Quantity
Linux host with a single Ethernet interface	4
Unmanaged switch	1

Introduction and Background

Domain name system (or DNS) is an Internet-wide distributed database for name lookup. The system composes of many cooperating DNS servers to perform the mapping between domain names and IP addresses. As we know, remembering a domain name, e.g., www.cpe.kmutt.ac.th, is easier than remembering 202.44.12.85. If we have no DNS or other conversion mechanisms, we have to remember IP address of each machine or network in order to reach it. The process of finding an IP address for a given domain name is called *name resolution* or *forward lookup*. We can also convert a given IP address to a domain name, which is the process known as *reverse lookup*.

An easy way to do name-address translation between domain names and IP addresses to store the mapping information in `/etc/hosts` file. This method is suitable for a storing a small set of static IP

addresses of machines in the same network since we have to maintain individual file for each machine. If we need to update the IP address of a server, we have to modify this file in all machines. In `/etc/hosts`, each line contains an entry containing IP address, fully qualified domain name, and optionally alias (or short host name). Multiple host names can be mapped into a single IP address.

Here is an example `/etc/hosts`

```
# Line beginning with '#' is a comment line
127.0.0.1       localhost
208.164.186.1   deep.network.lab          deep
208.164.186.2   mail.network.lab          mail student_mail
208.164.186.3   web.network.lab           web web2
```

In this example, the hostname 'deep' and its full domain name 'deep.network.lab' are mapped to IP address 208.164.186.1. You can think of 'network.lab' as the domain for the host 'deep'.

The configuration of DNS resolver in Linux involves 3 files:

- `/etc/nsswitch.conf`: This file contains configuration information for a program called the *name service switch*. It controls a variety of naming services on a Linux system. For DNS functions, the file needs to contain a line

  ```
  hosts: dns files
  ```

  With this line, name resolution will first invoke DNS and if fails, do a lookup in the file `/etc/hosts`.
- `/etc/resolv.conf`
- `/etc/hosts`

The structure of the DNS database is hierarchical and can be viewed as inverted tree. The top level is the root node. Each node in the tree has a text label and it is also the root of a new subtree. These subtrees represent segments or *domains* of overall databases. Each domain can be further divided into smaller partitions or *subdomains*.

DNS uses a client-server style of interaction. DNS servers or *nameservers* contain information about some segments of the database and make that information available to clients, called *resolvers*. Resolvers can be stand-alone program or library routines that create queries and send them across a network to a nameserver.

On Linux systems, DNS software is based on BIND (Berkeley Internet Name Domain) software distribution. The process name is `named`. To configure `named` in Ubuntu, we configure the configuration file

`/etc/bind/named.conf.local` and related zone data files. The zone data files contain the information for the forward lookup and the reverse lookup.[1]

For example, suppose we have a DNS server with IP address 192.168.2.7/24 and the hostname is `ns1`. It is configured to be authoritative to the domain (zone) `utopia.net` and the zone files are `utopia.net.db` and `utopia.net-reverse.db`. Then, the configuration file `/etc/bind/named.conf.local` would look like this:

```
# DB file for forward lookup zone
zone  "utopia.net" {
    type master;   # For slave, use 'type slave; masters {ip of master;};'
    file "/etc/bind/zones/utopia.net.db";
};
# DB file for reverse lookup zone
zone "2.168.192.in-addr.arpa" {
    type master;
    file "/etc/bind/zones/utopia.net-reverse.db";
};
```

The name of the zone files can be anything you like, and many zones can be defined in the same file. Make sure the directory permission of /etc/bind/zones/ is readable by bind process.

The zone file contains DNS resource records for forward lookup and reverse lookup. In this example, the content of the zone file `utopia.net.db` may look like this:

```
$TTL 86400       ; Default TTL for a DNS record cache in seconds (1 day)
@  IN SOA   ns1.utopia.net. root (
         1      ; SN must increase anytime you made change to this file.
         28800 ; refresh by slave (secondary) DNS server every 8 hours (if SN changes)
         14400 ; retry period if slave fails to contact
         3600000 ; How long slave server keeps its cache if it cannot contact primary server.
         86400  ; TTL Default time for slave to cache this zone file. Should be a day or less if
DNS records change regularly.
       )
@ IN NS ns1.utopia.net. ; Domain utopia.net. has the nameserver ns1.utopia.net.
                        ; Recommended as we can change ip address of ns1 without
                        ; affecting this record.
; @    IN NS 192.168.2.7.   ; Not recommended
ns1  IN A  192.168.2.7
www  IN CNAME ns1
```

The first line defines the directive $TTL, indicating how long the records may be cached.

[1] In some distribution, named.conf is read first. Its content will point to named.conf.local and others.

The following lines are DNS records, which are in the format

```
Name [TTL] Class Type Record_Data
```

Class is always `IN` (Internet), and `TTL` is set to the default value if not specified.

In the first record,

```
@  IN SOA   ns1.utopia.net. root ()
```

the type is SOA (Start of Authority), which tells that this server is a primary DNS server for this zone. There can be only one SOA record in a zone file. The "@" character expands the $ORIGIN directive that should be defined in the beginning of the zone file, e.g.,

```
$ORIGIN utopia.net
```

If $ORIGIN directive is not defined, bind9 will synchronize its value from the zone name in `/etc/bind/named.conf.local`, which in this case is '`utopia.net`'. Any unqualified name (those not ended with dot) will be appended by $ORIGIN.

An SOA record provides information about the domain for which this DNS server is responsible, which is '`utopia.net.`'. The label '`ns1.utopia.net.`' after the label SOA indicates the nameserver that will respond authoritatively for the domain (the one originating the zone file), which could be an external server. If the label is not ended with dot, it will be appended by the zone origin. The label "`root`" is the email address of the DNS server administrator. The fields in () are Serial Number, Refresh, Retry, Expiry, and Minimum. Their meaning are explained in the comments. A line comment starts after semicolon (;).

The 2nd record states that the domain `utopia.net.` has the nameserver `ns1.utopia.net.` Beware of a dot (.) for a resource record of type NS because its presence matters. For example, `utopia.net.` (with dot at the end). A "." at the end means that the name is a fully qualified domain name (FQDN). Without the ending dot, the DNS server will automatically add $ORIGIN and the name becomes `utopia.net.utopia.net`.

In the 3rd record (in the comment), type NS means that 192.168.2.7 is the name server for the domain `utopia.net`. like in the 2nd record. The '@' at the beginning indicates that this name server is for name 'utopia.net.'. The IP address in a type NS record must end with '.' to indicate prevent the DNS server to take it as another name and add the zone origin to it. However,

In the 4th record, type A means that the hostname `ns1` has an IP address 192.168.2.7. In the 5th record, the hostname "www" is the alias of the hostname "ns1".

For the reverse lookup, the content of `utopia.net-reverse.db` is as follows:

```
$TTL 86400      ; Default TTL for a DNS record cache in seconds (1 day)
@  IN SOA   ns1.utopia.net. root (  ; Same as the forward zone file
            2345     ; serial
            28800  ; refresh  8 hours
            14400   ; retry
            3600000 ; expire
            86400  ; TTL
            )
@   IN NS ns1.utopia.net.  ; @ = utopia.net.
7   IN PTR ns1.utopia.net. ; Pointer record (192.168.2.7 is mapped to
                           ; ns1.utopia.net)
8   IN PTR ns2.utopia.net. ; Pointer record (192.168.2.8 is mapped to
                           ; ns2.utopia.net)
```

The 3rd record in the file says that 192.168.2.7 has the domain name ns1.utopia.net. Note that '.' is needed to indicate that `ns1.utopia.net.` is a fullly qualified domain name. Otherwise, 192.168.2.7 will be reversed lookup to `ns1.utopia.net.2.168.192.in-addr.arpa`

Once all the configuration files and zone files have been created, start the named server with the command

```
sudo /etc/init.d/bind9 restart
```

The named process may fail to start if the content in the configuration files or zone files have wrong syntaxes.

To check the syntax in the configuration file, use

```
named-checkconf /etc/bind/named.conf.local
```

To check the syntax in the zone file, use

```
named-checkzone zone-name zone-file
```

To see error messages when the dns server fails to run, use

```
tail /var/log/syslog
```

The next step is to let the resolver knows where to search for the zone file and and the IP address of the DNS server. To do so, add the following lines in `/etc/resolv.conf`

```
search utopia.net
nameserver 192.168.2.7
```

This must be done for every host in the network that wants to resolve name-address by using this DNS server. To test if the DNS server works correctly, type the command `nslookup` at the command prompt. When prompted, enter `ns1` and you should the IP address 192.168.2.7 shows up on the screen. Another way is to use the command `host`. Try '`host ns1`', '`host www`', and '`host 192.168.2.7`' (without quotes) at the command prompt and see what happens.

The dns cache can be cleared by using

```
/etc/init.d/nscd restart
```

Lab Procedures

Part I: Resolving using /etc/hosts

1. Connect three PCs to a switch as shown in Figure 3.1. Configure the IP addresses of PC1, PC2, and PC3 with the following commands:

   PC1: `ifconfig eth0 10.0.1.1/24 up`

   PC2: `ifconfig eth0 10.0.1.2/24 up`

   PC3: `ifconfig eth0 10.0.1.3/24 up`

   Ping among the PCs to verify the connectivity.

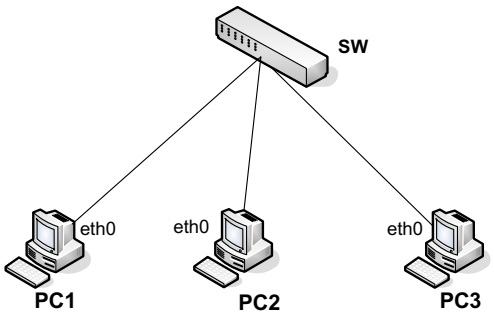

Figure 3.1

2. Clear the contents in `/etc/resolv.conf` and `/etc/hosts` on each PC. Also, check that no process with name `named` is running on any of these PCs with command

   ```
   ps aux | grep named
   ```

   If the named process is running, kill it with command

   ```
   /etc/init.d/bind9 stop
   ```

3. On PC1, edit the file `/etc/hosts` to associate hostnames with IP addresses for all the PCs.
   3.1. Assume that the domain is `cpenetwork.lab` and name the PCs as `pc1`, `pc2`, and `pc3`. Each PC must be reachable by its hostname and its domain name, e.g., `pc1` and `pc1.cpenetwork.lab`
   3.2. Pick another hostname `pc3alias` and assign it to PC3 so that PC3 has two different hostnames.

4. On PC2 and PC3, make sure that the file `/etc/hosts` is clean. That is, it contains only one entry looking like this:

```
127.0.0.1    localhost.localdomain    localhost
```

5. From PC1, ping PC3 by its IP address. Then ping PC3 by all its hostnames.
6. From PC2, ping PC1 by its IP address and then by its hostname.

Lab Question

1.1 Explain why the ping in Steps 5 and 6 succeeds or fails.

Part II: Setting up DNS server

1. Continuing from the network in Figure 3.1, remove entries from `/etc/hosts` in all PCs except the entry of 127.0.0.1.
2. Set up PC2 as the DNS server with `cpenetwork.lab` as the domain name by appropriately configurating the files .
   - Give the hostnames of PC1, PC2, PC3 as `pc1`, `pc2`, and `pc3` respectively.
   - Also give PC2 the alias hostnames `dns-server` and `www`

3. On all PCs, set resolver configuration in `/etc/resolv.conf` to point to the IP address of the DNS server so that they can lookup an IP address for the domain `cpenetwork.lab`. by entering the following content:

```
nameserver 10.0.1.2
search cpenetwork.lab
```

4. On PC2, start the DNS server by issuing command

   ```
   /etc/init.d/bind9 restart
   ```

   You can verify that named has been started by using process command `ps aux` and look for process name `named` with

   ```
   ps aux | grep named
   ```

Resolver using DNS server:

5. At PC2, start Wireshark capture on eth0. Disable name resolution feature in Wireshark to prevent interference from Wireshark by unchecking all name resolution options in Capture option dialog.

6. Clear DNS cache at PC1 and PC3 with the command

   ```
   /etc/init.d/nscd restart
   ```

7. Set the Wireshark display filter at PC2 to '`dns`'. Then, do the following steps:
   a) At PC1, ping a few packets to PC3 by hostnames `pc3` and `pc3.cpenetwork.lab`
   b) At PC3, ping a few packets to PC2 by its alias names `dns-server` and `www`.
   c) At PC1, ping a few packets to PC3 by hostname `pc3`

   After each step above, take a moment to observe what frames have been captured before doing the next one.
8. At each PC, clear the contents of `/etc/hosts`, `/etc/resolv.conf`, and all the DNS configuration and zone files you have created in this lab.

Lab Questions

Observe the traffic capture for each Ping step in Step 7 and answer the following questions:

2.1 What kinds of the DNS messages generated in each step? Explain their purposes and briefly indicate what kind of information is contained in those messages.

2.2. Do all the ping commands in Step 7 generate a DNS message? Why or why not?

Review Questions

1. What is the role of file `/etc/resolv.conf` in a Unix-based file system? Explain the meaning of the following content in `/etc/resolv.conf`:

   ```
   domain kmutt.ac.th
   nameserver 202.44.8.66
   nameserver 202.44.8.138
   nameserver 202.28.6.226
   ```

2. What is the role of file `/etc/hosts` in a Unix-based file system? Explain the meaning of the following content in `/etc/hosts`:

   ```
   10.1.130.213 staff.kmutt.ac.th  staff
   10.1.130.106 mysql.kmutt.ac.th
   10.1.130.103 netbackup-cc
   ```

3. In `named` DNS server program, there are many configuration parameters. What do `time-to-live`, `expire`, `refresh`, and `retry` mean?
4. Describe the concept of root servers. Why do we need them?
5. What are the primary, secondary DNS servers of kmutt.ac.th domain (both names & IP addresses)? How can you find out about these servers?
6. DNS can store information of many types. The resolver can specify the desired type. What do types NS, A, MX mean?
7. What is the standard port/protocol when a resolver contacts a DNS server?