Ubuntu Server DNS

Introduction

- Reverse DNS mappings
- How to configure a primary server for reverse mappings
- Secondary DNS servers and their configuration
- DNS logging

DNS Reverse Mapping

- DNS Reverse Mapping
  - Given an IP Address a DNS can determine the host details.
  - Although sometimes this is required for diagnostic purposes, more frequently these days it is used for security reasons to trace a hacker or spammer.
    - It can confirm that the specified IP address does represent the indicated host.
  - In order to perform reverse mapping DNS designers have defined a special (reserved) domain name called IN-ADDR.ARPA.

Reverse Lookup Approach

- An IPv4 address is written as follows:
  - **`192.168.254.17`**
- This IPv4 address defines a host address of 17 in the Class C address range **`192.168.254.x`**
- In this case, the most important part (the highest node) is on the left (**`192`**), not the right.
- This is the 'wrong way round' for DNS.

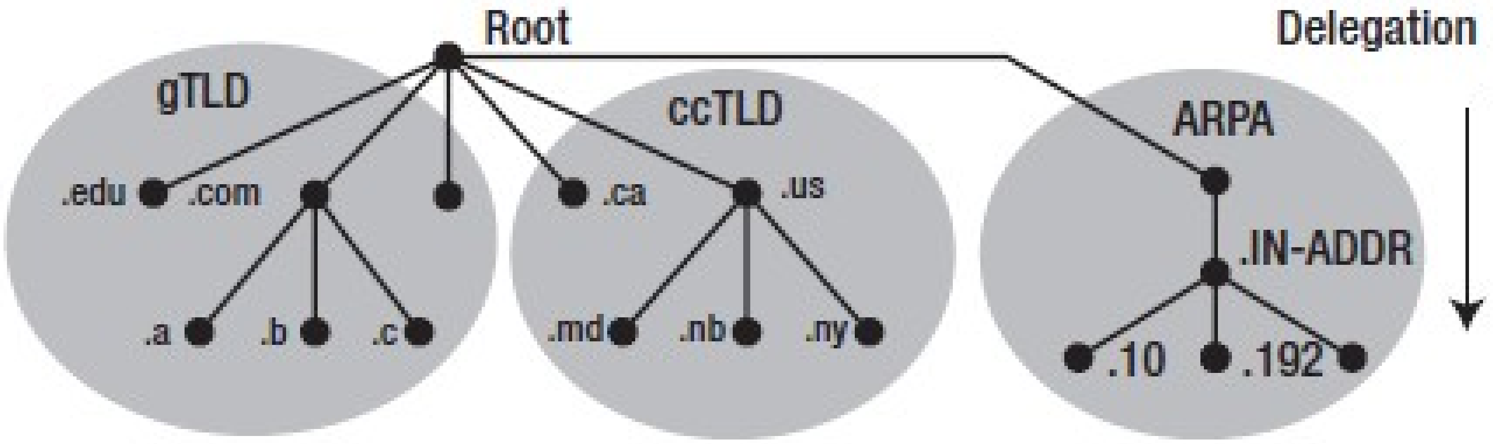
Reverse Lookup Approach

- The solution is elegantly simple:
  - create the domain name, reverse the order of the address and build the hierarchy under the special domain name
- IN-ADDR.ARPA
  - TLD is ARPA.
  - SLD is IN-ADDR

Reverse Lookup Approach

- IPv4 address (**17**) is the host address and defined inside a zone file.
- The result of the preceding manipulation is as follows:
  - IPv4 address =**192.168.254.17**
  - Class C base = **192.168.254**
  - Reversed Class C base = **254.168.192**
  - Added to IN-ADDR.ARPA domain =
    - **254.168.192.IN-ADDR.ARPA**

Reverse Lookup Approach



Reverse Zone File (101)

- To add a Reverse Zone you must modify the **`named.conf.local`** file so it points to the reverse lookup database (text file)

Reverse Zone File

- **named.conf.local**
  - **Add**

This is based on subnet (Class A,B,C)

```
zone “0.168.192.in-addr.arpa" {
    type master;
    file "/etc/bind/db.192.168.0";
};
```

Simple Reverse Zone File

```
;
; BIND reverse data file for 0.168.192.in-addr.arpa
;
$TTL    604800
@       IN      SOA     ns.student.co.uk. admin.student.co.uk. (
                              1         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      ns.student.co.uk.
55      IN      PTR     ns.student.co.uk.
```

Reverse Zone File

- The **Serial Number** in the Reverse Zone needs to be incremented on each change.
- For each **A** record you configure in **`/etc/bind/db.student.co.uk`** you need to create a **PTR** record in **`/etc/bind/db.192`**
- After creating the reverse zone file restart **`bind9`**

```
sudo service bind9 restart
```

Zone File Records

- **PTR**
  - The **PTR** RR is standardized in RFC 1035
  - Maps an IPv4 address to a particular host in the domain or zone.

Zone File Records

- **PTR** Syntax

```
name ttl class rr name
```

- **e.g**.

```
2       IN PTR ns1.example.com.
```

Zone File Records

Syntax	Example	Description
name	2	Although this looks like a number, it is in fact treated as a name. The name is unqualified, causing the **`$ORIGIN`** directive value to be substituted. You could have written this as **`2.254.168.192.INADDR.ARPA.`** (using the FQDN format).
ttl		There is no ttl value defined for the RR, so the zone default of 2d from the $TTL directive will be used.
class	IN	Internet Class
name	ns1.example.com.	Defines that a query for 192.168.254.2 will return ns1.example.com. This name *must* be written in the FQDN notation $ORIGIN would create the wrong name.

Configuring a Secondary DNS server

- A secondary DNS server provides redundancy and increases the availability of the service.
- A secondary DNS’s mode of operation is controlled via the master DNS’s SOA record which dictates how long it operates as a secondary DNS and also how often it refreshes etc.
- The master's serial number dictates if any updates occur at all!

Configuring Secondary DNS server

- Install **`bind9`** onto a basic server.
- Just like a primary, a secondary server can support multiple zones.
- In the case of a secondary server, the zones must have the same names as on the primary.
- You must specify both Forward and Reverse zones.
- All configuration is carried out in the **`/etc/bind/named.conf.local`** file
- Some configuration is required on the Master.

Configuring Secondary DNS server

The only configuration that is required on the secondary server is to identify the zones that must be mirrored and to give the location of the primary server for each zone.

- On the primary server, the zone needs to be configured to allow another DNS server (the secondary) to have authority to transfer the zone information.

Configuring Secondary DNS server

- The secondary server is configured to obtain its zone information from the primary server by editing the **`/etc/bind/named.conf.local`** configuration file.

```
zone “student.co.uk” {
    type slave;
    file “bak.student.co.uk”;
    masters { 192.168.1.55; };
};
```

NOTE: type is set to slave

NOTE: no directory specified for db file

You can have multiple masters

Configuring Primary server to allow transfers

- Transfer authorisation is enabled by adding the allow transfer option to the primary server's **/etc/bind/named.conf.local** configuration file.
- You can specify multiple addresses to allow more than one secondary server.

```
zone “student.co.uk” {
    type master;
    file “/etc/bind/db.student.co.uk”;
    allow-transfer {192.168.1.56; 192.168.1.57;};
};
```

Configuring primary server to allow transfers

- Don't forget that you should also arrange for the transfer of the reverse zone by making similar changes in the configuration files on both the primary and secondary servers.
- For all the changes to take effect and function correctly you must restart all the DNS servers.
  - Primary
  - Secondary
- All cached databases files are stored in:-

```
/var/cache/bind/
```

Configuring primary server to allow transfers

- These changes on the secondary server will result in entries in the system log showing that the operation has been carried out.

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

- You can view the last few entries on the **`syslog`** by entering the command below.

```
$sudo tail /var/log/syslog
```

Refreshing the Secondary DNS outside of the schedule

- Changes can be made on an ad hoc basis to the primary server's zones and then the primary can be restarted.
- Given time the secondary will retrieve any changes that are made.
- You may need to force a refresh of a secondary server to update the zone cache rather than wait (based on the SOA record)
- One of the many utilities you have to manage your **DNS** is the **`rndc`** Command
  - **`sudo rndc refresh <zone>`**

Logging DNS Activity

- A common practice in identifying problems with DNS configurations is to enable DNS logging.
- Logging is also used as a mechanism to determine the activity of a network's users.
  - What have the users been looking at (Big Brother!!!!)
  - This can be used to block unwanted sites in a firewall

Logging DNS Activity

- There are many options that can be used but the two primary logging directives that we are interested in are:-
  - Channel – where to put the log data
  - Category – what log data to collect
- Logging is configured in the **`bind9`** configuration file.

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

Logging DNS Activity

- Logging – directive to configure logging

```
logging {
    channel query.log {
    file "/var/log/named/query.log";
    severity debug 3;
    };
    category queries { query.log; };
};
```

Setting up the log file

- Since the *named daemon* runs as the *bind* user the **/var/log/named/query.log** file must be created and the ownership changed

```
sudo mkdir /var/log/named
sudo touch /var/log/named/query.log
sudo chown bind /var/log/named/query.log
```

Finally, restart the **bind9** server for your changes to take effect

```
sudo service bind9 restart
```

Conclusion

We have covered:

- the purpose of a Reverse Zone
- how to define a Reverse Zone in **bind**
- the **PTR** Resource Record type
- how to configure a secondary DNS server
- how to setup logging of DNS activities

Setting up the log file

- Before the **named** daemon can write to a new log file, the **AppArmor** profile must be updated.
  - **AppArmor** is a process manager that controls access to system resources.
- To configure the **AppArmor** service for **named** you need to edit one of the **AppArmor** config files.

```
/etc/apparmor.d/usr.sbin.named
/etc/apparmor.d/local/usr.sbin.named
```

- By default the **AppArmor** profile allows reading and writing by the **named** daemon of all files in the directory **/var/log/named** – so if you've chosen to put your DNS logs in this directory, AppArmor needs no further configuration.

Setting up the log file

- Otherwise, the **`log`** file needs to be given its permissions in one of the **AppArmor** config files, e.g

```
/var/log/query.log rw,
```

.

- Next, reload the profile for **`AppArmor`** changes:

```
cat /etc/apparmor.d/usr.sbin.named | sudo apparmor_parser -r
```

- Finally restart the **`bind9`** server for the logging to begin

```
sudo service bind9 restart
```