Chapter 6
BIND
Naming Your Servers

6.0 BIND -- The Name-Server

So far we have been dealing with IP addresses. Whenever connecting from the client to the server we used the IP of the server rather than its hostname. From now on, after we configure the DNS (Domain Name System) server, we can use hostnames instead of IP addresses. The package BIND, that we are going to set up, binds hostnames with IP addresses. The acronym BIND stands from Berkley Internet Name Domain. A DNS server listens on port 53.
Before setting it up, there are certain things that you should know about DNS. If you already know these, you can skip this section.

6.0.1 Hostnames and Domainnames

To identify a host in the Internet exclusively, it is not enough to simply know the hostname, since there can be so many hosts with the same hostname. So you need to have its 'full name' called the fully qualified domainname (FQDN). FQDN composes of the hostname, subdomainnames, and the root domainname all of them separated by dots. For example in `compaq.hazarika.org`, `compaq` is the hostname, `hazarika` is a subdomain with org as the root domain. Generally when people say domainname, they refer to the sub and the root domainname together. So in our example hazarika.org is the domainname.

6.0.2 The Domain Name System

DNS is a hierarchical naming system just like in Linux filesystem, files are placed in a hierarchical order. At the top level are the root domains for example .com, .edu, .org etc.

Any hostname in the Internet will always end in one of these root domain. Under these root domains are its subdomains. Say for example a hypothetical `hazarika.org`. Now any hosts under the domain `hazarika.org` will always end in `hazarika.org`. Now again, these subdomains can have further subdomains. In our example, it might be `hazarika.org, home.hazarika.org` or `friends.hazarika.org`. This can go on even further.

Now, suppose you want to own a domainname say `letslearnlinux.edu`, you need the permission of the central authority for your region. Actually ICANN (Internet Corporation for Assigned Names and Numbers) is responsible for all Internet domainnames and IPs, but it has delegated different authorities and companies to take care of their region. But if you want to have a domainname under `hazarika.org`, you need the permission of `hazarika.org`, since it has full authority to create subdomains under its own domain (all domains ending in `hazarika.org`). This applies to all sub levels. For example, `hazarika.org` has delegated different subdomains to different departments and say you want your host - `chiropriyo`, to be a part of `friends.hazarika.org`, you need the permission of authority of `friends.hazarika.org` and not the central authority of `hazarika.org`. Thus your FQDN (Fully Qualified Domainname) will be `chiropriyo.friends.hazarika.org`.

6.0.3 DNS Servers

To hold the information of the root domains (com, edu, org), there are 13 root name servers (`a.root-servers.net`, `b.root-servers.net` to `m.root-servers.net`). Now, these root servers should hold the information of the nameservers of its immediate subdomain. That is, in our example, it needs to have the information (IP to host mapping etc) of the nameserver for `hazarika.org` say `ns.hazarika.org`, but it does not bother about what other hosts are within `hazarika.org`. Now the subdomain nameservers will have the information of its immediate subdomain nameservers and also the hosts under its own domain. This applies to all name servers of all subdomains. To illustrate this, lets look at our example. The nameserver of the domain `hazarika.org`, will have the information about the nameservers of its subdomains -- `friends.hazarika.org` and `home.hazarika.org` as well as it needs to have information about hosts directly under its domain, say `compaq.hazarika.org`. But it will not store any information about `chiropriyo.friends.hazarika.org` since `friends.hazarika.org` has its own nameserver, say `ns.friends.hazarika.org` for that purpose.

6.0.3.1 Zones

All the hosts in a particular domain whose information is maintained by a particular nameserver are said to be in the same zone. To understand zones, I am taking the above example. The nameserver `ns.friends.hazarika.org` contains all the information about hosts within the domain `friends.hazarika.org`. All these hosts are said to be in the same zone. Similarly all the hosts in `home.hazarika.org` will be in another zone maintained by another nameserver. However it is not necessary that the nameserver should fall in the same zone. It can be in a different zone. And the same nameserver can work for multiple zones. Each zone must have at least one nameserver maintaining the database. There is no strict rule that a subdomain must belong to a different zone. Say `home.hazarika.org` does not want to run a separate nameserver and use the central nameserver. Then all the hosts within the domain `home.hazarika.org` will fall under the zone of `hazarika.org`. The hostnames and domainnames for the zone are kept in files called forward zone files.
Note: In this section, by nameservers I meant primary nameservers. See the following section for primary nameservers.

6.0.3.2 Primary, Secondary, Authoritative and Caching Nameservers

The primary name server is that server which has the original database for its zone and does not consult any other servers for that. A zone can have secondary nameservers which gets its information from the master server which can be a primary nameserver or another secondary nameserver. Even secondary servers stores the zone files. Every time a secondary nameserver starts, it checks whether its copy of the zone files are consistent with its master server. If it has the latest copy, it is happy with it, else it will load the latest copy from the master server. The concept of primary and secondary nameservers apply to zones. A primary server for a particular zone can be a secondary server for a different zone. Primary and secondary name servers are called authoritative nameservers for their zone.

A non authoritative name server is called a caching nameserver. It stores temporary data called cache. When a host queries a caching nameserver, it will look up its cache first. If it cannot find the required information, it will query other nameservers in turn and apart from returning the result to the querying host, it will cache the data it just collected. The time for which it will cache the data is called TTL (Time To Live) and it is decided by the zone administrator. A caching nameserver is also capable of negative caching. Say `nohost.hazarika.org` does not exist and turned up the result - host does not exist. The caching nameserver can cache such negative responses.

6.0.4 Reverse DNS lookup

As you probably know or might have guessed reverse DNS lookup means looking up the hostname given an IP. But the question is, how is it done? If you think about it, it needs an exhaustive search of the whole database which will make it a very slow and painful process. The solution to this problem is simple and elegant. IP address is a bunch of numbers separated by dots that can pinpoint a host in the Internet. A FQDN is a bunch of names separated by dots that can pinpoint a host in the Internet (though not exactly). So why not have domainname records that uses numbers instead of names and they will point to hosts (FQDN). Since in hostname specification, the hostname is written first, then the domainnames, in IP address records the host address will be written first and then the network address, thus reversing the IP address. The root domain for these subdomains is in-addr.arpa. So for example a host with IP 15.16.134.12 will have the FQDN `12.134.16.16.in-addr.arpa`. The files maintaining this kind of database are called reverse zone files.

Now enough of talking. Lets get to the real thing.

6.1 Installing BIND

In Fedora Core 5 you can find the BIND package in the DVD. The package is named `bind-9.3.2-4.1.i386.rpm`. Installation is described in section 1.1.

6.2 Configuring BIND

You might be wondering "We have installed four servers, but none of them felt like configuring a '*real*' server. Then why do people fuss about configuring servers?" Well the answer is, because in all the servers we have configured, the default configuration worked! It worked too well!! It worked so well that we did not bother to change any settings in the configuration file. The good news is that, the default configuration will work even this time. But only if there had been something called default domainname and default zone, we wouldn't have to bother about configuring nameservers at all. Our real task at hand now is to make the zone files -- having information about all the hosts in the zone, and we have to do it by our hand (for the primary nameserver). For the Microsoft Guy -- there is a graphical utility for configuring bind you can get by typing `system-config-bind` in the terminal.

For simplicities sake, in this chapter, we will configure only a primary nameserver.

6.2.1 `named.conf`

As you have already figured out, all the servers have a main configuration file -- the heart and soul of that server. This time the file is `/etc/named.conf`. Open it up and see.

Hopefully you noticed, C++ and C style comments can be used apart from our good old shell style comments. As you remember, C, C++ comments either start with // or are enclosed within /* and */

Another thing you might have noticed is that, all the statements end with a semicolon, again C, C++ style! Some of these statements are within braces and there is a semicolon after the closing bracket, like in structures and classes in C and C++!

The first statement is `options`. Right now the only option you have to know about is the `directory` option. Just like you specified Apache's directory for keeping the website, here you can specify which directory contains the zone files. If you do not use this option, you will have to give the full path of the zone files.

The next statement is `controls`. Leave it as it is. You don't have to bother about it now.

The remaining section contains the zone statements. We will have to add one statement for each zone file. We can come to configure this section later, once we are done with making our zone files. By default, BIND places them in the directory `/var/named/`.

6.2.2 The Forward Zone Files

The records present in the zone files are explained below. While I am explaining, it will be helpful for you to understand if you open up a zone file and see it for yourself. I am providing a sample zone file `hazarika.zone`. Open it up and see.

Before describing the records, there are certain things you should know. Hostnames and domainnames are not case-sensitive and zone file do not discriminate between them, but case is preserved (i.e. the case remains as it is). FQDN must be specified with a trailing dot (.). If you do not specify the dot in the end, the nameserver will treat it as hostname and add the domainname automatically. Comments in zone files start with a semicolon -- ;

6.2.2.1 TTL Value

It specifies time to live to be used by caching servers.

```
$TTL    <time>
```

Here time can be in hours, days or weeks with trailing H, D or W respectively. If you do not specify anything, it will interpreted as seconds. For a not-so-changing zone entry one day will be safe enough.

```
$TTL 1D
```

6.2.2.2 SOA Record

This states the start of authority (SOA) for the zone. It can appear only once in the zone file. Lets take a look at our zone file for the domain hazarika.org.

```
hazarika.org. IN SOA notebook.hazarika.org. manna.hazarika.org. (
                         1             ;  serial
                         1D            ;  refresh
                         1H            ; retry
                         1W            ; expiry
                         1D)           ; negative caching TTL
```

`hazarika.org.` (note the trailing dot) must start from the first column which shows that our nameserver is authoritative for the domain `hazarika.org`.

`IN` specifies that the zone file is for the Internet. Other classes are not used so much, so you don't have to bother about that. This field is optional and can be omitted.

The entry after `SOA`, `notebook.hazarika.org.` is the FQDN of the nameserver for the domain `hazarika.org`.

`manna.hazarika.org` is the email address of the person in charge of the zone. The first dot stands for @, in this case `manna@hazarika.org`.

The other fields are within bracket which allows to write the values in more than one lines. It is written this way for better human readability and you can write everything in a single line without the brackets without any problem.

The first value is the serial no. which the secondary name server uses to determine whether it has the up-to-date zone file.

The second value is the refresh rate of the secondary DNS server for that particular zone.

The third value is the time after which the secondary server will retry connecting to its master after a connection failure.

If you fail at first, try, try again, but for how long? You might say we should never give up and keep trying till eternity's end, but the secondary nameserver will give up eventually if it is unable to connect to the master nameserver for too long(which is quite reasonable, if you can see). This time is specified in the fourth value. After this time period is over, it will contact the root servers.

The fifth and final value is the TTL for negative caching.

6.2.2.3 NS Records

This record specifies the nameservers for the zone. We are using two authoritative nameservers one primary and one secondary. So the entries are --

```
hazarika.org.  IN  NS    notebook.hazarika.org.
hazarika.org.  IN  NS    compaq.hazarika.org.
```

6.2.2.4 MX Records

This is used to specify mail servers for the domain. The number 10 is the relative preference-order with higher preference to lower values. Here since there is only one mail server, it does not matter.

```
hazarika.org.  IN  MX  10   compaq.hazarika.org.
```

6.2.2.5 Hostname Records and Aliases

Name-to-address mapping are written in the A record (A-Address). Here we are using private IP addresses since we are doing it for experimental purpose and we are configuring a hypothetical domainname. In real life, when you will configure “real” DNS servers in the Internet, they will have public IP address. The numbers of entries in this field will depend on number of hosts in the network.

```
compaq          IN    A       192.168.1.1
notebook        IN    A       192.168.1.2
lenovo          IN    A       192.168.1.3
thinkpad        IN    A       192.168.1.4
desktop         IN    A       192.168.1.5
```

Now, you can put all your aliases using CNAME which stands for canonical name. Since `compaq` is the HTTP server as well as the mail server, it is aliased as `www` and `mail`. Here, we also use alias ns which stands for nameserver, `ns1` and `ns2` signifies primary and secondary namserver.

```
www               IN   CNAME   compaq
mail              IN   CNAME   compaq
ns2               IN   CNAME   compaq
ns1               IN   CNAME   notebook
```

That's the end of the zone file. Hearing that you give a sigh of relief. But wait. What about your reverse zone file?

6.2.3 The Reverse Zone Files

For this we create a file called `192.168.1.rev`. I have provided this sample file too. If you have read section 6.0.4 about reverse dns, and understood the forward zone file properly, you should have no problem interpreting this file at all. I won't explain the entries in this file because they have already been explained.

The only thing that did not appear in the forward zone file is the `PTR` records. These are the address-to-name mappings.

```
1.1.168.192       IN PTR   compaq.hazarika.org.
2.1.168.192       IN PTR   notebook.hazarika.org.
```

Either make new zone files, or make necessary changes in the zone files I have provided with this manual and copy them under the directory `/var/named`.

6.2.4 The Other Zone Files

We have created our custom zone files. Apart from that, there are some other zone files that the installation of BIND provides. These should be there for proper working of our nameserver. These are --

6.2.4.1 The `localhost` Zone Files

Since a host refers to itself as the localhost and has an IP address for referring to its own – the loopback address, there should be a forward and a reverse zone file corresponding to them. These files are created when you install BIND with the names `localhost.zone` and `named.local` respectively.

6.2.4.1 The Root Hints file

Your nameserver needs to know where the root nameservers are. These are the last resort, if a host queries about a domain that it does not know about, and none of its neighbors know. For this a file exists called `named.ca`. If it didn't exist, you would have to retrieve it from `ftp.rs.internic.net` (192.41.0.6) using anonymous FTP. Though it is provided, it is your responsibility to keep it up to date.

6.2.5 `named.conf` Revisited

Now that we have completed writing our zone files, we are ready to make the corresponding entry in our configuration file. Open up the configuration file `/etc/named.conf`. You can see that the zone statements for localhost and loopback address as well as the hint for root file is already included. Here is the zone statement for the `hazarika.org` that you should include at the end of the default zone statements.

```
zone "hazarika.org" in {
          type master;
          file "hazarika.zone"
};
```

It says that this nameserver is authoritative for the domain `hazarika.org`, `in` means the Internet class. Recall that I said IN was optional in the zone file. If you wouldn't have specified that, your server would have taken that value from the '`in`' entry in the `named.conf` file. '`type master`' means this is the primary nameserver for this zone. Additionally, you can add many more options like `allow-update`, `allow-query` etc. but right now you don't have to bother. You will get to know some of these features towards the end of this chapter.

Similarly, put an entry for the reverse zone --

```
zone "1.168.192.in-addr.arpa" in {
       type master;
       file "192.168.1.rev"
};
```

The last thing you need to do is to make your server aware that it itself is the nameserver. You can do that by changing the file `/etc/resolv.conf` so that it looks like the following (make the necessary changes in the domainname).

```
domain hazarika.org
nameserver 127.0.0.1
```

That's everything we need to change. We are all set. All we have to do is start the server and see it in action.

6.3 Starting and Stopping your Nameserver

Before you start your server, you can check whether your zone-files and configuration files are free from syntactical errors. Zone-files can be checked for errors by the command

```
named-checkzone <zonename> <filename>
```

So we check our hazarika.zone file --

```
named-checkzone hazarika.org hazarika.zone
```

We get the following output which tells us everything is alright.

```
zone hazarika.org/IN: loaded serial 1
OK
```

Additionally we can check that our configuration file `named.conf` is alright by the command

```
named-checkconf
```

If it returns you back the shell prompt silently, it means everything is alright. Else, it will show you the error message.

After checking and correcting the errors, if any, start your server just like the way described in section 1.3. The name of the service is `named` which you can find in `/etc/init.d/`.

6.4 The Client

There is one single change in the client computer needed to be done -- changing your primary DNS server to the one you just configured. Open the `/etc/resolv.conf` and change the line nameserver to the IP of the nameserver you just configured. Additionally, change your domain to your domainname. In our case, `resolv.conf` looks like this --

```
domain hazarika.org
nameserver 192.168.1.2
```

Now you can check whether the DNS server is working properly. Try pinging the hosts in your zone as well as other zones outside your domain.

```
ping www.hazarika.org
```

```
ping www.google.com
```

nslookup is a nifty tool for DNS queries. Try

```
nslookup www.hazarika.org
```

Now test whether reverse dnslookup is working properly.

```
nslookup 192.168.1.1
```

Apart from that, `host` can be used to check IP address of a hostname and vice versa. It has the same syntax as nslookup.

```
host www.hazarika.org
host 192.168.1.1
```

6.5 DNS Security

Configuring a DNS server is a meticulous job. First of all, it should give the clients the necessary information, otherwise the whole purpose of having a DNS server will fail. Again if it leaks too much information, it will attract attackers like a flower attracting honey-bees to collect its nectar. So you will have to be very careful on deciding who gets how much information. There are a few basic steps you can take just to keep this in check which are discussed below.

6.5.1 Stopping Zone Updates

`allow-update` is an option that allows other hosts to update the zones of your nameserver. This is needed only for dynamic updates and by default it is disabled. Do not use `allow-update` option if you don not understand the risk involved. If you cannot see any `allow-update` option in your named.conf, then it means your nameserver is safe, you need not worry since nobody is allowed to update your zones.

6.5.2 Restricting Queries

Sometimes you might not want everyone to query your nameserver except a few selected hosts. You can do this by the allow-query option. It has the following syntax

```
allow-query { <address-match-list> };
```

It can exists inside the option statement --

```
options {
       allow-query { 192.168.1/24; 192.168.0.1; };
};
```

or a zone statement --

```
zone "hazarika.org" in {
          type master;
          file "hazarika.zone"
          allow-query { 192.168.1/24; };
};
```

6.5.3 Preventing Unauthorized Zone Transfers

By default BIND allows everyone to transfer zones. If you do not understand the gravity of this situation, consider this -- you created a secret compartment in your house (may be to keep your valuable belongings) and you kept the map of your whole house (including the secret compartment) in a public place for everyone to view, so that the thieves can make a proper plan how to steal your belongings. The zone files give lots of valuable information for an attacker to make a proper plan to break into your system or systems in your network. You should keep your zone files safe and should not allow anyone except your slave nameservers to transfer zones from your nameserver. The syntax is same as `allow-query`.

Since we have not configured a secondary DNS server, we should allow noone to transfer zones from our server.

```
zone "hazarika.org" in {
          type master;
          file "hazarika.zone"
          allow-transfer { none; };
};
```

6.5.4 Creating A Chroot Jail for BIND

Now, consider the worst situation that an attacker gets unauthorized access to the server through a BIND exploit. He can disrupt DNS services no doubt, but there is a way to prevent access to the rest of the system such that other parts of the system are not harmed. This can be done by running named in a chroot jail such that named will treat `/var/named/chroot/` to be the root directory `/` and hence it cannot get out of it. There is a procedure to do this, but Fedora makes it even simpler. All you have to do is install the package `bind-chroot-9.3.2-4.1.i386.rpm`. After you install, you will find that all the files have been moved relative to `/var/named/chroot/` directory. `named.conf` will move from `/etc/` to `/var/named/chroot/etc/` and all the zone-files will move from `/var/named/` to `/var/named/chroot/var/named/`. Only a soft-link will exist in place of the original file. Thus, you have successfully created a chroot jail for your DNS server.

6.6 Conclusion

DNS server is one of the most important servers in the Internet. Some servers will not work at all or will not work properly without a DNS server. For others, it makes life a lot easier. Imagine you have to type `http://69.147.112.160/` instead of `http://mail.yahoo.com/`. Like it is useful, if it is not configured properly, can even cause havoc. It might lead to leaking sensitive information, security break-ins, and DNS spoofing by which Internet users can be directed to wrong and often malicious websites. Hence DNS servers should be configured with great care.