UNIT 3 NETWORK CONFIGURATION AND SETTING

Structure **Page Nos.**

3.0 INTRODUCTION

A computer network is a telecommunications network that allows computers to exchange data. The physical connection between networked computing devices is established using either cable media or wireless media. The best-known computer network is the Internet.

Network configuration is an activity to properly configure any network infrastructure through which various network applications/services can be run and accessed. Administrators must be able to configure IP addresses as well as other configuration files w.r.t different network services such as Dynamic Host Configuration Protocol (DHCP), Domain Name System (DNS), Email, Web Servers and other such related network services.

In this unit, you will learn configuration settings of various network services such as Dynamic Host Protocol (DHCP), Domain Name System (DNS) Network File System (NFS) and Web Server

3.1 OBJECTIVES

After going through this unit, you will be able to:

- know how to install various network services;
- configure a Dynamic Host Control Protocol;
- understand and configure a Domain Name System; and
- know on how to configure a Samba server.

3.2 CONFIGURING NETWORKS

A computer network facilitates interpersonal communications, allows sharing of files, and allows sharing of network and computing resources and other such related. To do so, there is an essential need to configure various network services those facilitate for the above network applications.

Network configuration and setup of various services in any organization is a challenging task to configure various network services such as DHCP, DNS, Web Service, Email, etc to run various applications that are to be accessed through network. The following section explains various core and essential network services that are to be required in any organization through which various applications can be run and accessed through an organizational network.

3.3 DYNAMIC HOST CONTROL PROTOCOL

The Dynamic Host Configuration Protocol (DHCP) is a network protocol used to configure devices that are connected to a network. These devices can communicate on that network using the Internet Protocol (IP). It involves clients and a server operating in a client-server model.

Dynamic Host Configuration Protocol (DHCP) automatically assigns IP addresses and other network configuration information (subnet mask, broadcast address, etc) to computers on a network. The DHCP server maintains a database of available IP addresses and configuration information. A client configured for DHCP will send out a broadcast request to the DHCP server requesting an address. The DHCP server will then issue a "lease" and assign it to that client. The time period of a valid lease can be specified on the server. DHCP reduces the amount of time required to configure clients and allows one to move a computer to various networks and be configured with the appropriate IP address, gateway and subnet mask.

How DHCP Works?

The following are the activities between DHCP Server and DHCP Client.

- Lease Request: Client broadcasts request to DHCP server with a source address of 0.0.0.0 and a destination address of 255.255.255.255. The request includes the MAC address which is used to direct the reply.
- IP lease offer: DHCP server replies with an IP address, subnet mask, network gateway, name of the domain, name servers, duration of the lease and the IP address of the DHCP server.
- Lease Selection: Client receives offer and broadcasts to al DHCP servers that will accept given offer so that other DHCP server need not make an offer.
- The DHCP server then sends an acknowledgement to the client. The client is configured to use TCP/IP.
- Lease Renewal: When half of the lease time has expired, the client will issue a new request to the DHCP server.

Download and Install the DHCP Package

Most RedHat and Fedora Linux software product packages are available in the RPM format, whereas Debian and Ubuntu Linux use DEB format installation files. When searching for these packages, remember that the filename usually starts with the software package name and is followed by a version number, for example the file name as ‘dhcp-3.23.58-4.i386.rpm’.

Managing the DHCP daemon is easy to do, but the procedure differs between Linux distributions. Here are some things to keep in mind.

- Firstly, different Linux distributions use different daemon management systems. Each system has its own set of commands to do similar operations.
- Secondly, the daemon name needs to be known and in this case the name of the daemon is dhcpd.

dhcpd.conf File

You can define your server configuration parameters in the dhcpd.conf file which may be located in the /etc the /etc/dhcpd or /etc/dhcp3 directories depending on your version of Linux.

The dhcpd.conf configuration file formats in Debian / Ubuntu and Redhat / Fedora are identical.

Here is a quick explanation of the dhcpd.conf file.

How do you configure DHCP?

The following is the dhcpd.conf file

```
ddns-update-style interim

ignore client-updates


subnet 192.168.1.0 netmask 255.255.255.0 {

   # The range of IP addresses the server
   # will issue to DHCP enabled PC clients
   # booting up on the network

   range 192.168.1.201 192.168.1.220;

   # Set the amount of time in seconds that
   # a client may keep the IP address

  default-lease-time 86400;
  max-lease-time 86400;

   # Set the default gateway to be used by
   # the PC clients

   option routers 192.168.1.1;
   # Don't forward DHCP requests from this
   # NIC interface to any other NIC
   # interfaces

   option ip-forwarding off;

   # Set the broadcast address and subnet mask
   # to be used by the DHCP clients

  option broadcast-address 192.168.1.255;
  option subnet-mask 255.255.255.0;

   # Set the NTP server to be used by the
   # DHCP clients

  option ntp-servers 192.168.1.100;

   # Set the DNS server to be used by the
   # DHCP clients

  option domain-name-servers 192.168.1.100;

   # If you specify a WINS server for your Windows clients,
    # you need to include the following option in the dhcpd.conf file:
   option netbios-name-servers 192.168.1.100;

    # You can also assign specific IP addresses based on the clients'
    # ethernet MAC address as follows (Host's name is "laser-printer":

   host laser-printer {
       hardware ethernet 08:00:2b:4c:59:23;
      fixed-address 192.168.1.222;
    }
}
#
# List an unused interface here
#
subnet 192.168.2.0 netmask 255.255.255.0 {
}
```

DHCP Servers with Multiple NICs

DHCP servers with multiple interfaces pose two configuration challenges. The first is setting up the correct routing and the second is making sure only the required interfaces are listening to serve DHCP.

Routing

When a DHCP configured PC boots, it requests its IP address from the DHCP server. It does this by sending a standardized DHCP broadcast request packet to the DHCP server with a source IP address of 255.255.255.255.

If your DHCP server has more than one interface, you have to add a route for this 255.255.255.255 address so that it knows the interface on which to send the reply; if not, it sends it to the default gateway.

You can temporarily add a route to 255.255.255.255 using the following route add command:

```
root# route add -host 255.255.255.255 dev eth0
```

Create a permanent route to 255.255.255.255. This will vary according to your version of Linux

In Fedora / RedHat ,add the route to your /etc/sysconfig/network-scripts/route-eth0 file, if the route needs to be added to your eth0 interface. The following is an example:

```
#
# vi  /etc/sysconfig/network-scripts/route-eth0 and add the following entry and then
save
#

255.255.255.255/32 dev eth0
```

in Ubuntu / Debian, add the route to your /etc/network/interfaces file. In this case the route is added to the eth0 interface. The following is an example:

```
#
# vi/etc/network/interfaces and add the following and then save
#
iface eth0 inet static
      up route add -host 255.255.255.255 eth0
```

Listening

Once you have defined the interface for your DHCP routing, you should also ensure that your DHCP server only listens on that interface and no others. This methodology to do this varies depending on your version of Linux.

In Fedora / RedHat, the /etc/sysconfig/dhcpd file must be edited and the DHCPDARGS variable edited to include the preferred interface. Fox example interface eth0 is preferred.

```
# vi /etc/sysconfig/dhcpd then add the following and save

DHCPDARGS=eth1
```

In Debian / Ubuntu the /etc/default/dhcp3-server file must be edited and the INTERFACES variable edited to include the preferred interface. For example interface eth0 is preferred.

```
# vi /etc/default/dhcp3-server then add the following and save

INTERFACES="eth0"
```

You will be able to verify success in one of two ways. First the netstat command using the –au options will give the list of interfaces listening on the bootp (DHCP) UDP port. The following is an example:

```
root# netstat -au  | grep bootp

udp        0      0 192.168.1.100:bootps     *:*

root#
```

Secondly, your /var/log/messages file will also reveal the following defined interfaces used when the DHCPd daemon was restarted.

```
Jun  8 17:22:44  dhcpd: Listening on LPF/eth0/00:e0:18:5c:d8:41/192.168.1.0/24

Jun  8 17:22:44  dhcpd: Sending on   LPF/eth0/00:e0:18:5c:d8:41/192.168.1.0/24
```

Once, the above messages revealed when DHCPd daemon was started, the configuration is success then go for launch.

Note: Client systems running on Linux/Windows shall be configured to use DHCP.

3.4 DOMAIN NAME SYSTEM (DNS)

A DNS server, or name server, is used to resolve an IP address to a hostname or vice versa.

It is a hierarchical distributed naming system for computers, services, or any resource connected to the internet or a private network. It associates various information with domain names assigned to each of the participating entities. Most prominently, it translates easily memorized domain names to the numerical IP addresses needed for the purpose of locating computer services and devices worldwide.

By setting up a DNS server, you become part of a hierarchy of DNS servers that make up the internet. At the top of this hierarchy is the root server, represented by a dot (".") below the root server are the top level domains (such as .com,org,and so on).

Understanding DNS

The basic function of name server is to answer queries by providing the information that those queries request. A DNS name server primarily translates domain and host names into IP addresses. Each domain is typically represented by a least two DNS servers. The following are different types of DNS servers:

- **Primary (master) name server** contains authoritative information about the domains that it serves. In response to queries for information about its domains, this server provides that information marked as being authoritative. The primary is the ultimate source for data about the domain. The secondary name server only carries the same authority in that it has received and loaded a complete set of domain information from the primary.

- **Secondary (slave) name server** gets all information for the domain from the primary. As is the case for the primary, DNS considers the secondary information about the domain that it serves authoritative.

- **Caching name server** simply caches the information it receives about the locations of hosts and domains. It holds information that it obtains from other authoritative servers and reuses that information until the information expires.

- **Forwarding name server** is essentially a caching name server but is useful in cases where computers lie behind a firewall and in which only one computer can make DNS queries outside that firewall on behalf of all the internal computers.

Understanding Bind

Red Hat Linux (and most other Linux and UNIX systems) implement DNS services by using the Berkeley Internet Name Domain (BIND) software. The Internet software Consortium maintains BIND (at www.isc.org/products/BIND). The basic components of BIND include the following:

- **DNS server daemon** (/usr/sbin/named): the named daemon listens on a port for DNS services requests and then fulfills those requests based on information in the configuration files that you create. Mostly named receives requests to resolve the host names in your domain to IP address.

- **DNS configuration files** (named.conf and /var/named/ *): the/etc/named.conf file is where you add most of the general configuration information that you need to define the DNS services for your domain. Separate files in the /var/named directory contain specific zone information.

- **DNS lookup tools** to check that your DNS server is resolving host names properly. These include commands such as host, dig, and nslookup (which are part of the bind-utils software package).

To maintain your DNS server correctly, you can also perform the following configuration tasks with your DNS server:

Logging indicates what you want to log and where log files reside.)

Remote server options can set options for specific DNS servers to perform such tasks as blocking information from a bad server, setting encryption keys to you use with a server, or defining transfer methods)

There is no need to give out DNS information to everyone who requests it. Restrict access to those who request it based on the following.

Access control list can contain those hosts, domains or IP addresses that one wants to group together and apply the same level of access to DNS server. C acl records to group those addresses, and then indicate what domain information the locations in that acl can or can't access.

Listen-on ports by default, name server accepts only name server requests that come to port 53 on name server. You can add more port numbers if you want your name server to accept name-service queries on different ports.

Authentication is to verify the identities of hosts that are requesting services from DNS server, can use keys for authentication and authorization. (the key and trusted-keys statements are used for authentication.)

Quick-starting A DNS Server

The DNS server software that comes with the current Red Had Linux is Berkeley Internet name Daemon (BIND). The following components are required to configure BIND.

- **Configuration file** (/etc/named.conf) is the main DNS server configuration file.
- **Zone directory** (/var/named) is the directory containing files that keep information about the zone that you create for your DNS server.
- **Daemon process** (/usr/sbin/named) is the daemon process that listens for DNS requests and responds with information that the named.conf file presents.
- **Debugging tools (**named –checkconf, and named-checkzone) are to determine whetherthe created DNS configuration correctly.

The following are the points one has to keep it in mind in creating a DNS server:

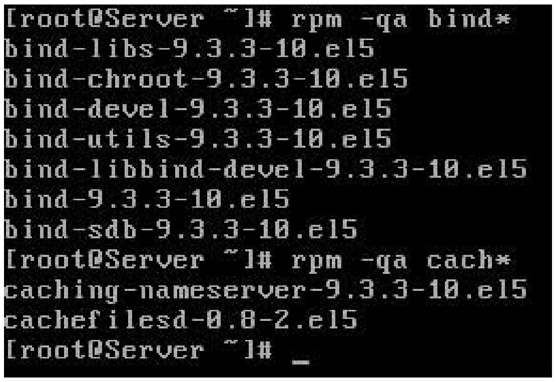
- identifying your DNS servers
- creating DNS configuration files (named.conf and /var/names/*)
- starting the named daemon
- Monitoring named activities

Configure DNS Server

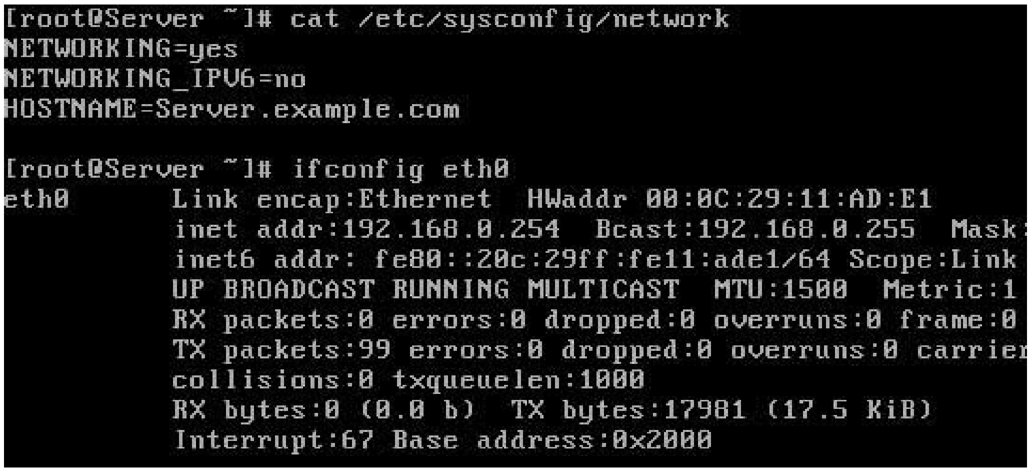
The following is a step by step procedure to configure a master DNS server.

For this activity, use three systems- one Linux server, second Linux clients and third window clients.

Step 1 - bind and caching-nameserver rpm is required to configure DNS. Check them for install, if not found then install

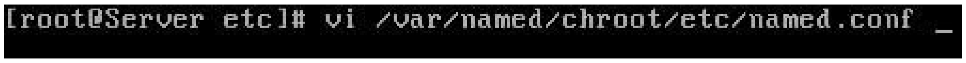

```
[root@Server ~]# rpm -qa bind*
bind-libs-9.3.3-10.el5
bind-chroot-9.3.3-10.el5
bind-devel-9.3.3-10.el5
bind-utils-9.3.3-10.el5
bind-libbind-devel-9.3.3-10.el5
bind-9.3.3-10.el5
bind-sdb-9.3.3-10.el5
[root@Server ~]# rpm -qa cach*
caching-nameserver-9.3.3-10.el5
cachefilesd-0.8-2.el5
[root@Server ~]# _
```

Step 2 - set hostname to server.example.com and ip address to 192.168.0.254

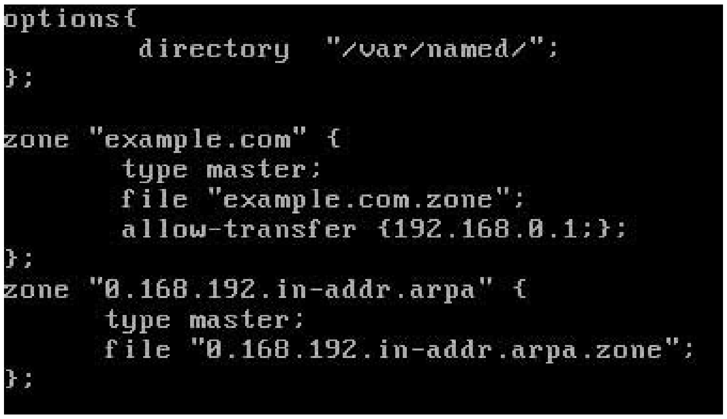

```
[root@Server ~]# cat /etc/sysconfig/network
NETWORKING=yes
NETWORKING_IPV6=no
HOSTNAME=Server.example.com

[root@Server ~]# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:0C:29:11:AD:E1
          inet addr:192.168.0.254  Bcast:192.168.0.255  Mask:
          inet6 addr: fe80::20c:29ff:fe11:ade1/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:99 errors:0 dropped:0 overruns:0 carrier
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:17981 (17.5 KiB)
          Interrupt:67 Base address:0x2000
```

Main configuration file for dns server is named.conf. By default, this file is not created in /var/named/chroot/etc/ directory. Instead of named.conf , a sample file /var/named/chroot/etc/named.caching-nameserver.conf is created. This file is used to make a caching only name server. You can also do editing in this file after changing its name to named.conf to configure master dns server or you can manually create a new named.conf file.

In our example, create a new named.conf file


```
[root@Server etc]# vi /var/named/chroot/etc/named.conf _
```

Or do editing exactly as shown here in image

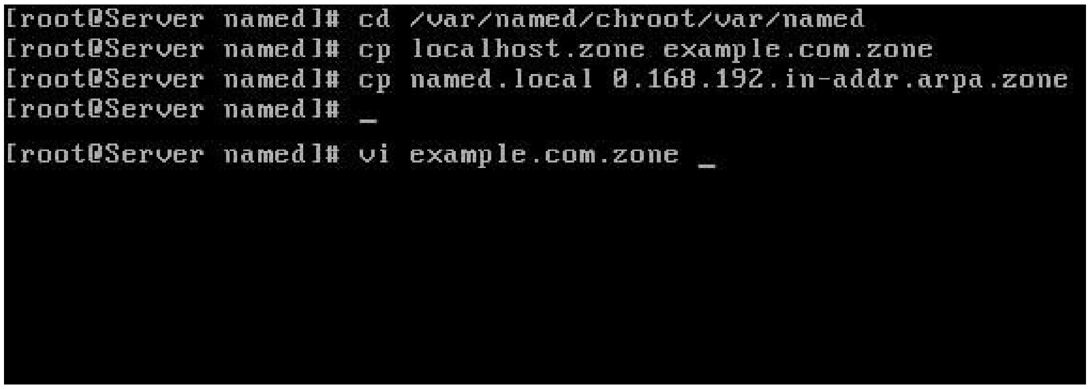

```
options{
        directory  "/var/named/";
};

zone "example.com" {
       type master;
       file "example.com.zone";
       allow-transfer {192.168.0.1;};
};
zone "0.168.192.in-addr.arpa" {
      type master;
      file "0.168.192.in-addr.arpa.zone";
};
```

save this file with :wq and exit

Configure zone file

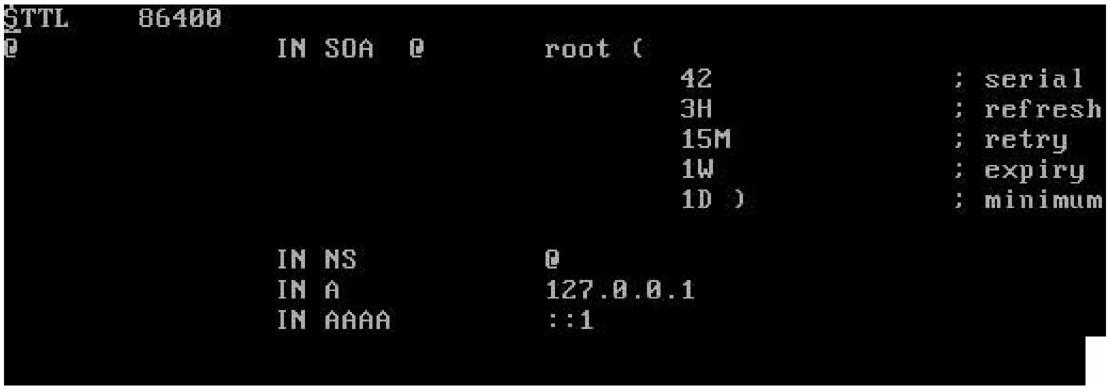
Defined two zone files example.com.zone for forward zone and 0.168.192.in-addr.arpa for reverse zone. These files will be stored in /var/named/chroot/var/named/ location.

Use two sample files for creating these files.



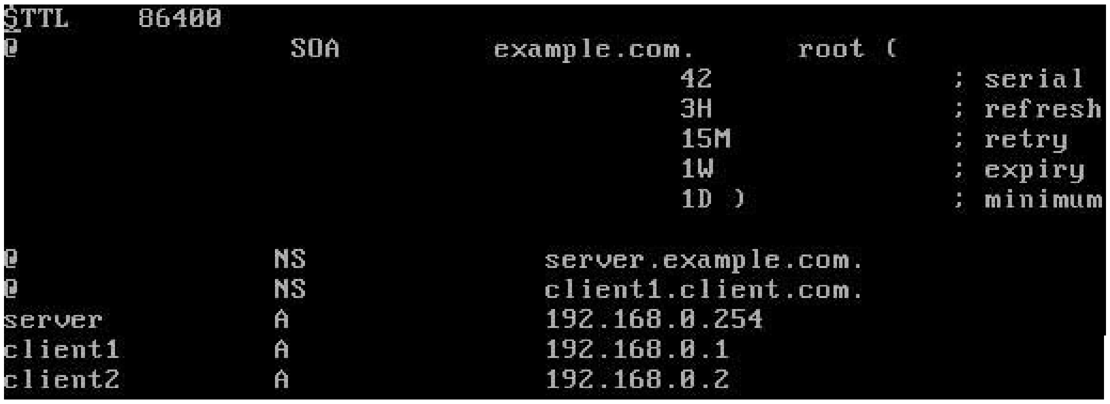
```
[root@Server named]# cd /var/named/chroot/var/named
[root@Server named]# cp localhost.zone example.com.zone
[root@Server named]# cp named.local 0.168.192.in-addr.arpa.zone
[root@Server named]# _
[root@Server named]# vi example.com.zone _
```

By default this file will look like this



```
$TTL    86400
@               IN SOA  @       root (
                                        42              ; serial
                                        3H              ; refresh
                                        15M             ; retry
                                        1W              ; expiry
                                        1D )            ; minimum

                IN NS           @
                IN A            127.0.0.1
                IN AAAA         ::1
```

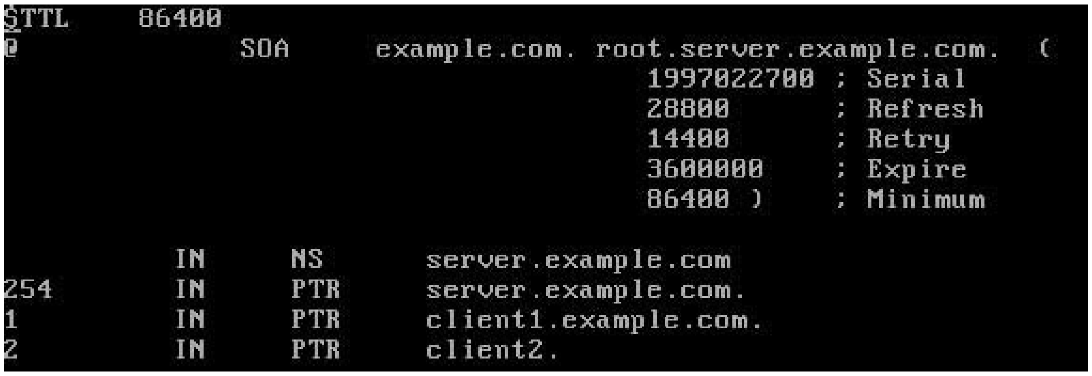
Change this file exactly as shown in image below



```
$TTL    86400
@                SOA        example.com.      root (
                                        42              ; serial
                                        3H              ; refresh
                                        15M             ; retry
                                        1W              ; expiry
                                        1D )            ; minimum

@               NS              server.example.com.
@               NS              client1.client.com.
server          A               192.168.0.254
client1         A               192.168.0.1
client2         A               192.168.0.2
```

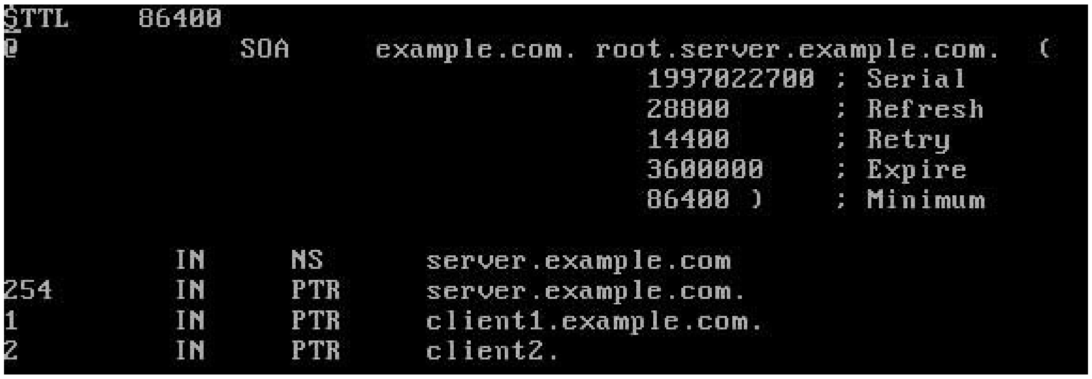
Now open reverse lookup zone file 0.168.192.in-addr.arpa

By default it will look like this



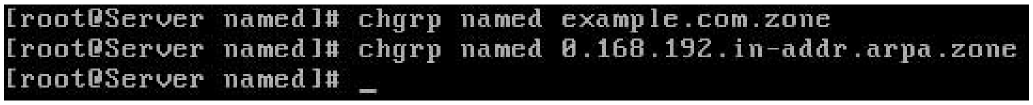
```
$TTL    86400
@             SOA     example.com. root.server.example.com.  (
                                   1997022700 ; Serial
                                   28800      ; Refresh
                                   14400      ; Retry
                                   3600000    ; Expire
                                   86400 )    ; Minimum

          IN     NS      server.example.com
254       IN     PTR     server.example.com.
1         IN     PTR     client1.example.com.
2         IN     PTR     client2.
```

Change this file exactly as shown in image below



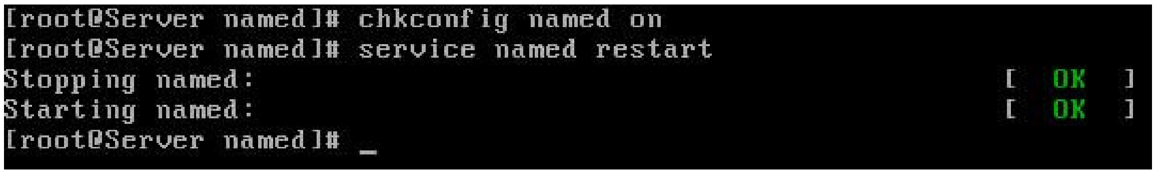
```
$TTL    86400
@             SOA     example.com. root.server.example.com.  (
                                   1997022700 ; Serial
                                   28800      ; Refresh
                                   14400      ; Retry
                                   3600000    ; Expire
                                   86400 )    ; Minimum

          IN     NS      server.example.com
254       IN     PTR     server.example.com.
1         IN     PTR     client1.example.com.
2         IN     PTR     client2.
```

Now changed the ownership of these zone files to named group



```
[root@Server named]# chgrp named example.com.zone
[root@Server named]# chgrp named 0.168.192.in-addr.arpa.zone
[root@Server named]# _
```

Now start the named service



```
[root@Server named]# chkconfig named on
[root@Server named]# service named restart
Stopping named:                                            [  OK  ]
Starting named:                                            [  OK  ]
[root@Server named]# _
```

If service restart without any error, it means successfully configured master name server.

Check Your Progress 1

1. Discuss the activities between DHCP Server and DHCP Client.

   ……………………………………………………………………………………

   ……………………………………………………………………………………

   ……………………………………………………………………………………

   ……………………………………………………………………………………

2. What are the different types of DNS servers? Explain.

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

3.5 NETWORK FILE SYSTEM

Network File System (NFS) is a server-client protocol for sharing files between computers on a common network. It allows a computer to access directories on remote computers by mounting them on a local file system as if they were a local disk The administrator on the NFS server has to define the directories that need to be activated, or exported, for access by the NFS clients, and administrators on the clients need to define both the NFS server and the subset of its exported directories to use. It is available on a variety of UNIX based operating systems, not just Linux. The server and client do not have to use the same operating system. The client system just needs to be ruining an NFS client compatible with the NFS server.

General rules to be followed

One should follow some general rules when configuring NFS.

- Only export directories beneath the / directory.
- Do not export a subdirectory of a directory that has already been exported. The exception being when the subdirectory is on a different physical device. Likewise, do not export the parent of a subdirectory unless it is on a separate device.
- Only export local filesystems.

  When mounting any filesystem on a directory, one should know that the original contents of the directory are ignored, or obscured, in favor of the files in the mounted filesystem. When the filesystem is unmounted, then the original files in the directory reappear unchanged.

Some NFS key concepts

The following are some key NFS background concepts that will help in overall understanding.

The Virtual File System (VFS)

VFS interface is the mechanism used by NFS to redirect all access to NFS-mounted files to the remote server. It is done in such a way that files on the remote NFS server appear to the user like those on a local disk.

Stateless Operation

Programs that read and write to files on a local filesystem rely on the operating system to track their access location within the file with a pointer. As NFS is a network-based file system, and networks can be unreliable, it was decided that the NFS client daemon would act as a failsafe intermediary between regular programs running on the NFS client and the NFS server.

Normally, when a server fails, file accesses timeout and the file pointers are reset to zero. With NFS, the NFS server doesn't maintain the file pointer information, the NFS client does. This means that if an NFS server suddenly fails, the NFS client can precisely restart the file access once more after patiently waiting until the server returns online.

Caching

NFS clients typically request more data than they need and cache the results in memory locally so that further sequential access of the data can be done locally instead of access from server. This is also known as a read ahead cache. Caching therefore helps to reduce the amount of network traffic while simultaneously improving the speed of some types of data access. The NFS server caches information too, such as the directory information for the most recently accessed files and a read ahead cache for recently read files.

NFS and Symbolic links

One has to be careful with the use of symbolic links on exported NFS directories. If an absolute link points to a directory on the NFS server that hasn't been exported, then the NFS client won't be able to access it. Mounting a filesystem on a symbolic link actually mounts the filesystem on the target of the symbolic link. Plan carefully before doing this.

NFS Background mounting

NFS clients use remote procedure call (RPC) suite of network application helper programs to mount remote filesystems. If the mount cannot occur during the default RPC timeout period, then the client retries the mount process until the NFS number of retires has been exceeded. The default is 10,000 minutes, which is approximately a week. The difficulty here is that if the NFS server is unavailable, the mount command will hang for a week until it returns online.

Hard and Soft mounts

The process of continuous retrying, whether in the background or foreground, is called a hard mount. NFS attempts to guarantee the consistency of data with these constant retries. With soft mounts, repeated RPC failures cause the NFS operation to fail not hang and data consistency is therefore not guaranteed. The advantage is that the operation completes quickly, whether it fails or not. In such case it is better not to place critical data that needs to be updated regularly or executable programs in such a location. NFS has currently various versions such as version 2,3 and 4. Version 1 was a prototype.

NFS Daemons

NFS isn't a single program, but a suite of interrelated programs that work together to get the job done. The following are several daemons that are started when a system goes into run level 3 or multi-user mode. The mountd and nfsd daemons are run on systems that are servers. The automatic startup of the server daemons depends on the existence of entries that are labeled with the NFS file-system type in /etc/dfs/sharetab. To support NFS file locking, the lockd and statd daemons are run on NFS clients and servers.

i) automountd Daemon

   This daemon handles the mounting and unmounting requests from the autofs service. The syntax of the command is as follows:

   automountd [**-Tnv**] [-D *name*=*value*]

   The command behaves in the following ways:

–T enables tracing.

-n disables browsing on all autofs nodes.

-v selects to log all status messages to the console.

-D *name*=*value* substitutes *value* for the automount map variable that is indicated by *name*.

The default value for the automount map is /etc/auto_master. Use the **-T** option for troubleshooting

ii) *lockd Daemon*

This daemon supports record-locking operations on NFS files. The lockd daemon manages RPC connections between the client and the server for the Network Lock Manager (NLM) protocol.

iii) *mountd Daemon*

This daemon handles file-system mount requests from remote systems and provides access control. The mountd daemon checks /etc/dfs/sharetab to determine which file systems are available for remote mounting and which systems are allowed to do the remote mounting. One can use the **-v** option and the **-r** option with this command.

The **-v** option runs the command in verbose mode. Every time an NFS server determines the access that a client should be granted, a message is printed on the console. The information that is generated can be useful when trying to determine why a client cannot access a file system.

The **-r** option rejects all future mount requests from clients. This option does not affect clients that already have a file system mounted.

iv) nfs4cbd Daemon

This daemon is for the exclusive use of the NFS version 4 client that manages the communication endpoints for the NFS version 4 callback program. The daemon has no user-accessible interface.

v) nfslogd Daemon

This daemon provides operational logging. NFS operations that are logged against a server are based on the configuration options that are defined in /etc/default/nfslogd. When NFS server logging is enabled, records of all RPC operations on a selected file system are written to a buffer file by the kernel

vi) statd Daemon

This daemon works with lockd daemon to provide crash and recovery functions for the lock manager. The statd daemon tracks the clients that hold locks on an NFS server. If a server crashes, on rebooting, statd daemon on the server contacts statd on the client. The client statd can then attempt to reclaim any locks on the server. The client statd also informs the server statd when a client has crashed so that the client's locks on the server can be cleared.

vii) rpcbind: (portmap in older versions of Linux)

The primary daemon upon which all the others rely, rpcbind manages connections for applications that use the RPC specification. By default, rpcbind listens to TCP port 111 on which an initial connection is made. This is then used to negotiate a range of TCP ports, usually above port 1024, to be used for subsequent data transfers. You need to run rpcbind on both the NFS server and client.

viii) nfs

Starts the RPC processes needed to serve shared NFS file systems. The nfs daemon needs to be run on the NFS server only.

ix) nfslock

Used to allow NFS clients to lock files on the server via RPC processes. The nfslock daemon needs to be run on both the NFS server and client.

x) netfs

It allows RPC processes run on NFS clients to mount NFS filesystems on the server. The netfs daemon needs to be run on the NFS client only.

Installing NFS

RedHat Linux installs nfs by default, and nfs is also activated when the system boots. One can check whether nfs installed or not using the RPM command in conjunction with the grep command to search for all installed nfs packages.

The following is an example to check:

```
[root# rpm -qa | grep nfs
redhat-config-nfs-1.1.3-1
nfs-utils-1.0.1-3.9
[root#
```

A blank list means that there is a need to install the required packages.

There is also a need to have the RPC rpcbind package installed, and the rpm command can tell whether it is installed or not. Use rpm in conjunction with grep, to check all the rpcbind applications installed or not.

The following is an example:

```
[root# rpm -q rpcbind
rpcbind-4.0-57
[root#
```

A blank list means that there is a need to install the required packages.

If nfs and rpcbind are not installed, they can be added fairly easily once find the nfs-utils and rpcbind RPMs. Remember that RPM filenames usually start with the software's name and a version number, as in nfs-utils-1.1.3-1.i386.rpm.

A Typical NFS Scenario

A small office has an old Linux server that is running out of disk space. The office cannot tolerate any down time, even after hours, because the server is accessed by overseas programmers and clients at nights and local ones by day. Budgets are tight and the company needs a quick solution until it can get a purchase order approved for a hardware upgrade. Another Linux server on the network has additional disk capacity (for ex. at /data partition) and the office would like to expand into it as an interim expansion NFS server.

Configuring NFS on The Server

Both the NFS server and NFS client have to have parts of the NFS package installed and running. The server needs rpcbind, nfs, and nfslock operational, as well as a correctly configured /etc/exports file.

The following explains how exports file is to be configured:

The /etc/exports File

The /etc/exports file is the main NFS configuration file, and it consists of two columns. The first column lists the directories you want to make available to the network. The second column has two parts. The first part lists the networks or DNS domains that can get access to the directory, and the second part lists NFS options in brackets.

For example for the scenario as mentioned above, suppose requires the following:

Read-only access to the /data/files directory to all networks

Read/write access to the /home directory from all servers on the xxx.xxx.1.0 /24 network, which is all addresses from xxx.xxx.1.0 to xxx.xxx.1.255

Read/write access to the /data/test directory from servers in the my-site.com DNS domain

Read/write access to the /data/database directory from a single server xxx.xxx.1.203.

In all cases, use the sync option to ensure that file data cached in memory is automatically written to the disk after the completion of any disk data copying operation.

```
#/etc/exports

/data/files         *(ro,sync)

/home               xxx.xxx.1.0/24(rw,sync)

/data/test          *.my-site.com(rw,sync)

/data/database        xxx.xxx.1.203/32(rw,sync)
```

After configuring /etc/exports file, there is a need to activate the settings, but first make sure that NFS is running correctly.

Starting NFS on the Server

Configuring an NFS server is straightforward:

Use the chkconfig command to configure the required nfs and RPC rpcbind daemons to start at boot and also activate NFS file locking to reduce the risk of corrupted data.

The following is the example:

```
root@# chkconfig  nfs on

root@# chkconfig  nfslock on

root@# chkconfig  rpcbind on
```

Use the init scripts in the /etc/init.d directory to start the nfs and RPC rpcbind daemons.

The following are examples to use the start option, but by using stop and restart options one can stop or restart the service as when needed.

```
root# service rpcbind start
root# service nfs start
root# service nfslock start


root# service rpcbind stop
root# service rpcbind restart
```

Test whether NFS is running correctly with the rpcinfo command or not. The following I an example to list the running RPC programs.

```
root# rpcinfo -p localhost
   program vers proto   port
    100000    2   tcp    111  portmapper
    100000    2   udp    111  portmapper
    100003    2   udp   2049  nfs
    100003    3   udp   2049  nfs
    100021    1   udp   1024  nlockmgr
    100021    3   udp   1024  nlockmgr
    100021    4   udp   1024  nlockmgr
    100005    1   udp   1042  mountd
    100005    1   tcp   2342  mountd
    100005    2   udp   1042  mountd
    100005    2   tcp   2342  mountd
    100005    3   udp   1042  mountd
    100005    3   tcp   2342  mountd
root#
```

Accessing NFS Server Directories from the Client

In most cases, users want their NFS directories to be permanently mounted. This requires an entry in the /etc/fstab file in addition to the creation of the mount point directory.

The /etc/fstab File

The /etc/fstab file lists all the partitions that need to be auto-mounted when the system boots. Edit the /etc/fstab file for   NFS directory to be made permanently available to users on the NFS.

For example, mount the /data/files directory on server test(IP address xxx.xxx.1.100) as NFS-type filesystem using the local /mnt/nfs mount point directory.

```
#/etc/fstab

#Directory                   Mount Point    Type   Options       Dump   FSCK

xxx.xxx.1.100:/data/files   /mnt/nfs        nfs    soft,nfsvers=2  0      0
```

This example is used the soft and nfsvers options; The table 1 outlines these and other useful NFS mounting options:

Table 1: NFS Option

Option	**Description**
bg	Retry mounting in the background if mounting initially fails
fg	Mount in the foreground
soft	Use soft mounting
hard	Use hard mounting
rsize=n	The amount of data NFS will attempt to access per read operation. The default is dependent on the kernel. For NFS version 2, set it to 8192 to assure maximum throughput.
wsize=n	The amount of data NFS will attempt to access per write operation. The default is dependent on the kernel. For NFS version 2, set it to 8192 to assure maximum throughput.
nfsvers=n	The version of NFS the mount command should attempt to use
tcp	Attempt to mount the filesystem using TCP packets: the default is UDP.
intr	If the filesystem is hard mounted and the mount times out, allow for the process to be aborted using the usual methods such as CTRL-C and the kill command.

Configuring NFS on the Client

NFS configuration on the client requires to start the NFS application; create a directory on which to mount the NFS server's directories that are exported via the /etc/exports file, and finally to mount the NFS server's directory on local system’s directory, or mount point.

Starting NFS on the Client

Use the following steps to configure NFS on the client.

Use the chkconfig command to configure the required nfs and RPC rpcbind daemons to start at boot. Activate nfslock to lock the files and reduce the risk of corrupted data.

```
root# chkconfig  netfs on
root# chkconfig  nfslock on
root# chkconfig  rpcbind on
```

Use the init scripts in the /etc/init.d directory to start the nfs and RPC rpcbind daemons. Use the following example les to start, stop and restart the processes:

```
root# service rpcbind start
root# service netfs start
```

```
root# service nfslock start
root# service rpcbind stop
root# service rpcbind restart
```

Test NFS whether is running correctly or not with the rpcinfo command. The flowing is an examplw:

.

```
root# rpcinfo -p
   program vers proto   port
    100000    2   tcp    111  portmapper
    100000    2   udp    111  portmapper
    100024    1   udp  32768  status
    100024    1   tcp  32768  status
    100021    1   udp  32769  nlockmgr
    100021    3   udp  32769  nlockmgr
    100021    4   udp  32769  nlockmgr
    100021    1   tcp  32769  nlockmgr
    100021    3   tcp  32769  nlockmgr
    100021    4   tcp  32769  nlockmgr
    391002    2   tcp  32770  sgi_fam
root#
```

3.6 WEB SERVER

The primary function of a web server is to cater web page to the request of clients using the Hypertext Transfer Protocol (HTTP). This means delivery of HTML documents and any additional content that may be included by a document, such as images, style sheets and scripts. The server that sends your web browser the code to display a web page is called a web server. There are countless web servers all over the Internet serving countless websites to people all over the world. Whether you need a web server to host a website on the Internet a Red Hat Enterprise Linux server can function as a web server using the Apache HTTP server. The Apache HTTP server is a popular, open source server application that runs on many UNIX-based systems as well as Microsoft Windows.

A user agent, commonly a web browser initiates communication by making a request for a specific resource using HTTP and the server responds with the content of that resource or displays an error message, if not available. The resource is typically a real file on the server's secondary storage, but this is not necessarily the case and depends on how the web server is implemented.

Samba Server

Samba is a software package that comes with Red Hat Linux to share file systems and printers on a network with computers that use the session massage block (SMS) protocol. SMB is the protocol that is delivered with windows operating systems for sharing files and printers. In Red Hat Linux, the Samba software package contains a variety of daemon processes, administrative tools, user tools, and configuration files.

The default Samba configuration file is smb.conf, which is in /etc/samba directory (/etc/samba/smb.conf). If you need to access features that are not available through the samba server configuration file you can edit /etc/samba/smb.conf file as required.

Daemon processes consist of smbd (the SMB daemon) and nmbd (the NetBIOS name server). The smbd is what makes the file sharing and printing services you add to your Red Hat Linux computer available to windows client computers. The following are some of the clients that Samba supports:

Window 9X

Window 2000

Window NT

Window ME

Window XP

Window for workgroups

Ms Client 3.0 for DOS

OS/2

Dave for Macintosh computer

Samba for Linux

As for administrative tools for samba, you have several shell commands at your disposal. You can check your configuration file using the testparm and testprns commands. The smbstatus command tells you which computers are currently connected to your shared

Samba configuration file (smb.conf)

```
# smb.conf is the main Samba configuration file. You find a full commented
# version at /usr/share/doc/packages/samba/examples/smb.conf.SUSE if the
# samba-doc package is installed.
# Date: 2008-05-28
[global]
   workgroup = WORKGROUP
   printing = cups
   printcap name = cups
   printcap cache time = 750
   cups options = raw
   map to guest = Bad User
   include = /etc/samba/dhcp.conf
   logon path = \\%L\profiles\.msprofile
   logon home = \\%L\%U\.9xprofile
   logon drive = P:
   usershare allow guests = Yes
   add machine script = /usr/sbin/useradd  -c Machine -d /var/lib/nobody -s /bin/false
%m$
   domain logons = No
   domain master = No
   security = user
   usershare max shares = 100
[homes]
   comment = Home Directories
   valid users = %S, %D%w%S
   browseable = No
   read only = No
   inherit acls = Yes
```

```
[profiles]
   comment = Network Profiles Service
   path = %H
   read only = No
   store dos attributes = Yes
   create mask = 0600
   directory mask = 0700
[users]
   comment = All users
   path = /home
   read only = No
   inherit acls = Yes
   veto files = /aquota.user/groups/shares/
[groups]
   comment = All groups
   path = /home/groups
   read only = No
   inherit acls = Yes
[printers]
   comment = All Printers
   path = /var/tmp
   printable = Yes
   create mask = 0600
   browseable = No
[print$]
   comment = Printer Drivers
   path = /var/lib/samba/drivers
   write list = @ntadmin root
   force group = ntadmin
   create mask = 0664
   directory mask = 0775

## Share disabled by YaST
# [netlogon]

[public]
   comment = RAID drive share
   inherit acls = Yes
   path = /local/public
   read only = No
```

How to configure Samba server

The following is the step by step procedure to configure a Samba Server

1. Server IP address is xxx.xxx.0.254 and machine name is Server1
2. Windows machine IP address is xxx.xxx.0.2 and machine name is client2
3. Firewall Should be disabled.

```
#chkconfig iptables off
#service iptables stop
#chkconfig ip6tables off
#service ip6tables stop
```

4. To check if all rpms required for samba are installed or not.

```
#rpm –qa | grep samba, then it displays the following
Samba-<version-name>
Samba-common-<version-name>
Samba-client-<version- name>
System-config-samba-<version-name>
```

5. If not, install it using

```
#rpm –ivh samba
```

6. Copy the original configuration file as smb.conf.bck

```
#cp /etc/samba/smb.conf /etc/samba.conf.bck
```
(This will keep a backup copy of your configuration file.)

7. Now edit the configuration file

```
#vi /etc/samba/smb.conf
```

8. To share home directory edit this part of /etc/samba/smb.conf

```
[homes]
comment=Home directories
browseable =no
writable=yes
```

9.To share printer edit this part of /etc/samba/smb.conf

```
[printers]
comment = All printers
path = /var/spool/samba
browseable = no
```

10.To configure share called IHNCSHARE edit this part of /etc/samba/smb.conf

```
[IHNC's share]
comment = Testing Samba Server
path = /samba
valid users = user1, user2
```

11. Save the file with wq command

12. To check your configuration file type testparm

```
#testparm
```

13. Create two samba users user1 and user2

```
useradd user1
useradd user2
```

14. Create smbpassword for both the users

```
smbpasswd –a user1
smbpasswd –a user2
```

15. Now you stop/start the samba services

```
#service smb stop
#service smb start
```

16. To see what is shared from your server through samba for a particular user

```
#smbclient -L Server1 –U user1 or smbclient –L //xxx.xxx.0.1 –U user1
```

17. To see what is shared for a particular host

```
#smbclient –L mac2
```

18. Now, permanently make samba server on

```
#chkconfig smb on
```

Check Your Progress 2

1. List the components required to configure BIND.

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

2. What is Samba Server? Explain its importance.

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

3.7 SUMMARY

In this unit, installation, configuration and setup of various network services such as Dynamic Host Control Protocol (DHCP), Domain Name System (DNS), Network File System (NFS) and Samba server are explained in detail with examples. This knowledge nay help to understand the concepts and install, configure and commissioning of other network services such as Email, FTP and such related services also. Student has to practice in real time to have more exposure and built confidence in configuration of network services.

3.8 ANSWERS TO CHECK YOUR PROGRESS

Check Your Progress 1

1. The following are the activities between DHCP Server and DHCP Client.

   - Lease Request: Client broadcasts request to DHCP server with a source address of 0.0.0.0 and a destination address of 255.255.255.255. The request includes the MAC address which is used to direct the reply.

   - IP lease offer: DHCP server replies with an IP address, subnet mask, network gateway, name of the domain, name servers, duration of the lease and the IP address of the DHCP server.

   - Lease Selection: Client receives offer and broadcasts to al DHCP servers that will accept given offer so that other DHCP server need not make an offer.

   - The DHCP server then sends an acknowledgement to the client. The client is configured to use TCP/IP.

   - Lease Renewal: When half of the lease time has expired, the client will issue a new request to the DHCP server.

2. The following are different types of DNS servers:

   Primary (master) name server contains authoritative information about the domains that it serves. In response to queries for information about its domains, this server provides that information marked as being authoritative. The primary is the ultimate source for data about the domain. The secondary name server only carries the same authority in that it has received and loaded a complete set of domain information from the primary.

   Secondary (slave) name server gets all information for the domain from the primary. As is the case for the primary, DNS considers the secondary information about the domain that it serves authoritative.

   Caching name server simply caches the information it receives about the locations of hosts and domains. It holds information that it obtains from other authoritative servers and reuses that information until the information expires.

   Forwarding name server is essentially a caching name server but is useful in cases where computers lie behind a firewall and in which only one computer can make DNS queries outside that firewall on behalf of all the internal computers.

Check Your Progress 2

1. The following components are required to configure BIND.

   Configuration file (/etc/named.conf) is the main DNS server configuration file.

   Zone directory (/var/named) is the directory containing files that keep information about the zone that you create for your DNS server.

   Daemon process (/usr/sbin/named) is the daemon process that listens for DNS requests and responds with information that the named.conf file presents.

Debugging tools (named –checkconf, and named-checkzone) are to determine whetherthe created DNS configuration correctly.

2. Samba is a software package that comes with Red Hat Linux to share file systems and printers on a network with computers that use the session massage block (SMS) protocol. SMB is the protocol that is delivered with windows operating systems for sharing files and printers. In Red Hat Linux, the Samba software package contains a variety of daemon processes, administrative tools, user tools, and configuration files.

   The default Samba configuration file is smb.conf, which is in /etc/samba directory (/etc/samba/smb.conf). If you need to access features that are not available through the samba server configuration file you can edit /etc/samba/smb.conf file as required.

3.9 FURTHER READINGS

1. Computer Networks by Andrew S Tanenbaum , Fifth Edition
2. SA2, Redhat System Administration I & II, Student Workbook
3. Cisco Certified Network Associate Study Guide, Seventh Edition by Todd Lammle
4. Redhat Enterprise Linux System Administration
5. http://en.wikipedia.org/wiki/dhcp
6. http://en.wikipedia.org/wiki/dns
7. http://en.wikipedia.org/wiki/nfs