



9

Network Management

CERTIFICATION OBJECTIVES

- 9.01 DNS/BIND
- 9.02 Network File System (NFS)
- 9.03 DHCP

- 9.04 Keeping Servers in Sync



- Two-Minute Drill

- Q&A Self Test

More complex networking services in Red Hat Enterprise Linux require more advanced administration methods. While graphical tools such as the Network Configuration tool are available to assist in configuring all aspects of Linux networking, the best way to learn networking is by practicing with the key command line utilities and associated configuration files.

This chapter starts with a discussion of the Domain Name System (DNS). DNS is a service that translates human-readable domain names such as `www.mommabears.com` to IP addresses such as `209.197.248.180`, and vice versa. Next, this chapter continues to a description of the Network File System (NFS), which is a powerful and versatile way of sharing filesystems between servers and workstations. DHCP allows a Linux computer to serve dynamic IP addresses. Finally, the section on keeping servers in sync demonstrates how you can set up a RHEL 3 computer to synchronize with a central time server.

INSIDE THE EXAM

More Network Services

Both Red Hat exams require that you configure a Linux workstation as a client on a network. On a network with Linux computers, that naturally includes using DNS servers, connecting to shared NFS directories, and taking IP address information from DHCP servers.

The RHCE exam requires that you configure Linux servers on a network. The Red Hat Exam Prep guide suggests that you can expect to configure DNS and NFS servers during the exam. While DHCP servers aren't explicitly listed in the Red Hat Exam Prep guide, they are listed in the Red

Hat curriculum for the RHCE exam (in the RH300 course outline).

In this chapter, the NFS and NTP services should be installed automatically with RHEL 3. I proceed to you about the packages required for the DNS and DHCP services. For each service, you'll need to remember to use a command such as **chkconfig** to make sure it starts the next time you boot Linux. It also will help you get full credit for the work you do on the Red Hat exams. I show you how to configure each service for basic operation. However, for a detailed discussion on how to secure these services, you'll need to read Chapter 10.

As you learn about these network services, you're learning about the services that you might configure and/or troubleshoot on the Red Hat exams. Take the time you need to understand the configuration files associated with each of these services, and practice making them work on your Linux computer. In some cases, two computers running Linux will be useful to practice what you learn in this chapter.

CERTIFICATION OBJECTIVE 9.01

DNS/BIND

DNS is the Domain Name System, which maintains a database that can help your computer translate domain names such as `www.redhat.com` to IP addresses such as `216.148.218.197`. As no individual DNS server is large enough to keep a database for the entire Internet, they can refer requests to other DNS servers.

DNS is based on the **named** daemon, which is built on the BIND (Berkeley Internet Name Domain) package developed through the Internet Software Consortium. More information is available from the BIND home page at www.isc.org/products/BIND. Red Hat Enterprise Linux 3 (and Red Hat Linux 9) includes BIND version 9.2. While this version of BIND supports the use of the `/usr/sbin/rndc` configuration interface, RHEL 3 still includes sample files based on the older `/etc/named.conf` configuration file. However, you can use the **rndc** command to manage DNS operation, in the same way that you used **apachectl** to manage the Apache server.

You can configure a DNS server by directly editing the DNS configuration files. Alternatively, you can configure a DNS server using the Red Hat Domain Name Service configuration tool. Careful use of both tools can help you learn more about DNS.

exam

Watch

I've found the Red Hat Domain Name Service tool to be less than reliable. It sometimes aborted unpredictably on my installations of

RHEL 3. In other words, the only reliable way to configure DNS on your RHEL 3 computer is by direct editing of configuration files.

You can set up four different types of DNS servers:

- A master DNS server for your domain(s), which stores authoritative records for your domain.
- A slave DNS server, which relies on a master DNS server for data.
- A caching-only DNS server, which stores recent requests like a Proxy server. It otherwise refers to other DNS servers.
- A forwarding-only DNS server, which refers all requests to other DNS servers.



redhat-config-bind is the successor to bindconf. Red Hat Enterprise Linux 3 includes a link from bindconf to redhat-config-bind.

DNS Packages

If you're just configuring your Linux computer as a DNS client, you can skip this section. The basic DNS client configuration files are automatically installed with even a minimal installation of RHEL 3.

On the other hand, if you're configuring your Linux computer as a DNS server, you'll need to install the packages associated with the DNS Name Server package group. You can do so with the Red Hat Package Management utility described in Chapter 4. However, there are only four RPM packages associated with DNS:

- **bind** Includes the basic name server software, including `/usr/sbin/named`.
- **bind-utils** Contains tools such as **dig** and **host** that allow you to ask a DNS server for more information about a specific Internet host. It should already be installed in any minimum installation of RHEL 3.
- **caching-nameserver** Allows you to set up a local cache of a remote DNS server to speed access from the computers on your network to a remote network such as the Internet.
- **redhat-config-bind** A GUI configuration tool useful for adding host and reverse address lookup data. It's not officially a part of the DNS Name Server package group.

Naturally, these tools are easy to install from any Red Hat network installation source that you may have created in Chapter 2. Different options and commands for installing RPMs from a remote installation source are described in Chapter 4.

The DNS Configuration Files

There are DNS configuration files required to configure your Linux computer as a client and as a server. There are two DNS client configuration files: `/etc/hosts` and `/etc/resolv.conf`. They are fairly straightforward, as described in the next section.

There are a number of additional configuration files that support the use of DNS as a server, as described in Table 9-1. You may have to create some of these files as you configure DNS.

In the following sections, I'll show you how to configure the files that you need for a working DNS server. But first, you should know how to configure your computer as a DNS client. One thing to remember is that all of the files in `/var/named` include

TABLE 9-1 DNS Server Configuration Files

DNS Configuration File	Description
<code>/etc/sysconfig/named</code>	You can set up different configuration and data file directories through this file.
<code>/etc/named.conf</code>	The main DNS configuration file. Incorporates <code>/etc/named.custom</code> , and calls data files in the <code>/var/named</code> directory.
<code>/etc/named.custom</code>	If you use the Red Hat DNS configuration tool, you can include additional settings in this file.
<code>/etc/rndc.key</code>	The authentication key required to support requests to the DNS server.
<code>/var/named/0.0.127.in-addr.arpa.zone</code>	The reverse zone file for the localhost computer.
<code>/var/named/netaddr.in-addr.arpa.zone</code>	The reverse zone file for the LAN; on an IPv4 network, it's the first three octets, written backwards. For example, for a network address of 192.168.30.0, the reverse zone file is 30.168.192.in-addr.arpa.zone.
<code>/var/named/domain.zone</code>	A database for a <i>domain</i> such as example.com.
<code>/var/named/localhost.zone</code>	The zone file for the localhost computer.
<code>/var/named/named.ca</code>	A list of root DNS servers on the Internet.
<code>/var/named/named.local</code>	A reverse zone record for the localhost.
<code>/var/named/named.stats</code>	Statistics from your DNS server.

6 Chapter 9: Network Management

a dot at the end of each domain name. For example, `/var/named/localhost.zone` lists the local computer as:

```
localhost.
```

In contrast, the domain names in `/etc/named.conf` and `/etc/named.custom` do not have dots at the end of their domain names.

A DNS Client

When your computer looks for another computer on a TCP/IP network such as the Internet, it typically looks in two places: `/etc/hosts` and any DNS servers that you've set up for your network. The order is determined by a single line in `/etc/host.conf`. The default is:

```
order hosts,bind
```

When your computer searches for another, this line tells your computer to search first through your `/etc/hosts` database. The following line in my `/etc/hosts` is what drives a Web browser address to my local IP address:

```
127.0.0.1    Enterprise3    localhost.localdomain    localhost
```

While you could theoretically also configure every computer on the Internet in your `/etc/hosts` configuration file, it's not realistic. However, if your LAN is small, you could add the IP address and hostname of each computer on your network in the `/etc/hosts` file. You could then duplicate this file on each computer on your LAN. Then you could use an external DNS server, provided by your ISP, for Internet access.

It's easy to configure a Linux computer as a DNS client. You may have already done so during the RHEL 3 installation process. If you have a working DHCP server, it probably provided you with the settings you need as a DNS client. Alternatively, you can configure your computer as a DNS client with the Red Hat Network Configuration utility described in Chapter 4. You can even configure your computer directly through your `/etc/resolv.conf` file. It's easy to do; the following version of this file lists two DNS servers on the Internet:

```
nameserver 207.217.120.83
nameserver 207.217.126.81
```

If you want to add or change the DNS servers for your computer, you can open this file directly in the text editor of your choice.



In the following section, you'll get a chance to learn about DNS nameserver configuration files from the ground up. Many of the lessons associated with a caching-only name server apply to slave and master DNS servers.

As with a number of Samba variables, “name server” and “nameserver” are both in common use with respect to servers that conform to Domain Name Service (DNS), and I use both variations throughout the book.

A Forwarding-Only Name Server

This first type of DNS server is simple. It requires a single command in the `/etc/named.conf` configuration file. As you can see, it's straightforward; I've set it to refer to the DNS servers that I had configured as a client:

```
options {
    directory "/var/named";
    forward only;
    forwarders {
        207.217.120.83;
        207.217.126.81;
    };
};
```

You can activate this configuration once you've activated this DNS server. With this configuration, any computer that looks to the local DNS server is forwarded to the IP addresses shown.

If the data can't be found in those DNS servers, it's up to those servers to ask others for the information. The alternative is to use the `/var/named/named.ca` file described later in this chapter.

A Caching-Only Name Server

When you request a Web page such as `www.osborne.com`, your network asks the configured DNS server for the associated IP address. This is usually known as a name query. If the DNS server is outside your network, this request can take time. If you have a caching-only name server, these queries are stored locally, which can save significant time while you or others on your network are browsing the same sites on the Internet.

When configuring a caching-only name server, the first step is to look at the `/etc/named.conf` configuration file. An abbreviated version of this file, configured for a caching name server, is shown in Figure 9-1.

The **directory** command tells your DNS server where to look for data files. All other files specified in `named.conf` are located in the `/var/named` directory. The `/etc/named.conf` file continues with the **controls** command, which allows you to use the **rndc** command to control DNS. The next three **zone** commands support access to various domains:

- The first zone “dot” refers regular requests to `/var/named/named.ca`, the list of root servers for the Internet.
- The second zone, for `localhost`, sets the forward zone for the local computer.
- The third zone is a reverse zone for the local computer.

FIGURE 9-1

`/etc/named.conf`,
configured for
a caching
nameserver

```
// generated by named-bootconf.pl

options {
    directory "/var/named";
    // query-source address * port 53;
};

//
// a caching only nameserver config
//
controls {
    inet 127.0.0.1 allow { localhost; } keys { rndckey; };
};

zone "." IN {
    type hint;
    file "named.ca";
};

zone "localhost" IN {
    type master;
    file "localhost.zone";
    allow-update { none; };
};

zone "0.0.127.in-addr.arpa" IN {
    type master;
    file "named.local";
    allow-update { none; };
};

include "/etc/rndc.key";
```



*If you have an older hardware firewall, it might expect DNS communication on TCP/IP port 53. If it does, you'll want to activate the query-source address *port 53 command.*

named.ca

If your DNS servers don't have the IP address for a domain name in its database, it can refer to other databases. By default, it refers to the `/var/named/named.ca` file, which is a list of the root DNS servers for the Internet. An excerpt is shown in Figure 9-2. This list changes from time to time and must be maintained. I include a shell script that you can use to maintain this file later in this section.

FIGURE 9-2

The root DNS servers are stored in `named.ca`

```
; This file holds the information on root name servers needed to
; initialize cache of Internet domain name servers
; (e.g. reference this file in the "cache . <file>"
; configuration file of BIND domain name servers).
;
; This file is made available by InterNIC
; under anonymous FTP as
; file /domain/named.cache
; on server FTP.INTERNIC.NET
;
; last update: Nov 5, 2002
; related version of root zone: 2002110501
;
; formerly NS.INTERNIC.NET
;
; .
; 3600000 IN NS A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET. 3600000 A 198.41.0.4
;
; formerly NS1.ISI.EDU
;
; .
; 3600000 NS B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET. 3600000 A 128.9.0.107
;
; formerly C.PSI.NET
;
; .
; 3600000 NS C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET. 3600000 A 192.33.4.12
;
; formerly TERP.UMD.EDU
;
; .
; 3600000 NS D.ROOT-SERVERS.NET.
D.ROOT-SERVERS.NET. 3600000 A 128.8.10.90
"/var/named/named.ca" 79L, 2499C
```

FIGURE 9-3

The localhost
.zone DNS
data file

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

                                1D IN NS      @
                                1D IN A      127.0.0.1
~
```

localhost.zone

The default `/var/named/localhost.zone` file is shown in Figure 9-3. It incorporates a basic DNS entry for the local computer, which you can use as a template for other computers on your network.

Reverse lookups with named.local

The `/var/named/named.local` file provides a reverse-lookup record for your computer, as `localhost`. As shown in Figure 9-4, the PTR record (on the last line in the file) is 1, which associates the loopback address, 127.0.0.1, with your computer.

Configuring a Simple Domain

Return to the `/etc/named.conf` configuration file. With what we know about DNS configuration files, we can create a Master DNS server. Assume that you've defined a simple domain for the computers on your network; call it `example.com`.

FIGURE 9-4

The named.local
reverse DNS file

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

                                IN      NS      localhost.

1         IN      PTR      localhost.
~
```



The `example.com` domain is a generic domain that can't be assigned on the Internet. You can therefore use this domain name on your private network (assuming that you don't have a domain of your own).

Look at the zone “0.0.127.in-addr.arpa” IN line near the bottom of the sample file. This says that the zone 0.0.127.in-addr.arpa will be defined, that the localhost is the master server (**type master**) for that zone, associated data is stored in a **file** called `named.local`, and no other DNS server is allowed to “update” or change the IP address associated with the localhost (**allow-update { none; }**).



Not all characters are allowed in hostnames. DNS can read only regular letters, numbers, and the hyphen (-) character. Unlike Linux, DNS does not distinguish between upper- and lowercase characters; for example, `Mail.Example.Com` is equivalent to `mail.example.com`.

Now you can add your network to your DNS server. Start by inserting a new zone section in the `/etc/named.conf` file, before the **include “/etc/rndc.key”**; command. (If you're also going to use Red Hat's GUI Domain Name Service tool, you should *instead* add this information to `/etc/named.custom`.)

```
zone "example.com" IN {
    type master;
    file "example.com.zone";
};
```

Note again that we do not use a period at the end of any address in `/etc/named.conf`. If you want to create a slave server, the information that you'd enter is similar:

```
zone "example.com" IN {
    type slave;
    file "slave.example.com.zone";
    masters {
        192.168.30.5
    };
};
```

The task for a slave server is easier; it periodically checks with the master DNS server, in this case the computer with an IP address of 192.168.30.5. When it does, it automatically reads the master DNS server data and creates the `slave.example.com.zone` file in the `/var/named` directory.

Creating a Zone File

If you're creating a master DNS server, you'll need to create a zone file. Based on the configuration so far, this will be an `example.com.zone` file in the `/var/named` directory. I've created one for my own network as shown in Figure 9-5.

Now let's decipher some of the language in this file:

`$TTL 3D` means that the default Time To Live (TTL) for data on this DNS server is three days. It's also common to have a TTL of 86400, which corresponds to the number of seconds in a day. You can specify individual TTLs for each entry in this file.

The SOA (Start Of Authority) record is the preamble to all zone files. It describes the zone where it comes from (a computer called `enterprise3.example.com`), the administrator e-mail for this DNS server (`hostmaster@example.com`). The SOA record command line also specifies a number of other parameters:

- Serial number is based on the date and version number. Based on the one shown in Figure 9-5, this database file is the first one created for this server on February 12, 2004.
- The refresh frequency determines how long the local DNS *slave* server waits before checking for updates from any master DNS servers.
- The retry frequency specifies how often the local DNS slave server retries contacting the master server.
- If there is no response from a DNS master server before the end of the expiration period, the local server stops accepting requests for the given domain.
- The Time To Live is the minimum amount of time other DNS servers should keep the local zone information in their remote cache.

The NS is the Name Server resource record, which refers to the name of the DNS server computer, which in this case is `enterprise3.example.com`.

The MX is the Mail Exchange record, which directs e-mail information to a particular computer, in this case, `mail.example.com`. Some number such as what's shown is required with any MX record.

In my `example.com.zone` file shown in Figure 9-5, I also have entries for three computers in this DNS zone. Each is associated with an A (address) record, its IP

FIGURE 9-5

An example
.com.zone file

```
;
; Zone file for Mike's example.com domain
;
; This is the full database file for this particular zone
; (which is typically a LAN or a network domain)
;
$TTL 3D
@      IN      SOA     enterprise3.example.com. hostmaster.example.com. (
                                200402121      ; serial number = today's date + rev number
                                8H              ; refresh frequency (8 hours)
                                2H              ; retry frequency (2 hours)
                                4W              ; expiration period for data (4 weeks)
                                3D )            ; Time to live of at least three days

                                IN      NS      enterprise3.example.com. ; Specifying the nameserver

                                IN      MX      10 mail.example.com.      ; See below for actual server

allaccess  IN      A      192.168.30.1
enterprise3 IN      A      192.168.30.2
entwks3    IN      A      192.168.30.5

mail       IN      CNAME   enterprise3
ftp        IN      CNAME   enterprise3
www        IN      CNAME   enterprise3
```

address. The final three entries specify canonical names (CNAME) for specific computers. For this zone file, `www.example.com` is also known as `enterprise3.example.com`.

CNAME allows you to assign several names to each computer. In this case, `mail`, `ftp`, and `www` are all aliases for the `enterprise3` computer. A CNAME is not a legal hostname for an e-mail address. For example, `admin@ftp.example.com` won't work based on the `example.com.zone` file shown.

If you have more than one mail server, you can add a number before the name of each MX computer. For example, based on the following commands:

```
IN      MX      10      mail.example.com.
IN      MX      20      mail2.example.com.
```

any e-mail directed to this domain is first sent to `mail.example.com`.

Save this file, and restart `named` with the **service named restart** command. Examine the results with the **host -l example.com** command. I've shown the result from my zone file in Figure 9-6.

FIGURE 9-6

Listing a working
DNS zone

```
[root@Enterprise3 root]# host -l example.com
example.com SOA enterprise3.example.com. hostmaster.example.com. 200402121 28800
7200 2419200 259200
example.com name server enterprise3.example.com.
example.com mail is handled by 10 mail.example.com.
allaccess.example.com has address 192.168.30.1
enterprise3.example.com has address 192.168.30.2
entwks3.example.com has address 192.168.30.5
ftp.example.com is an alias for enterprise3.example.com.
mail.example.com is an alias for enterprise3.example.com.
www.example.com is an alias for enterprise3.example.com.
example.com SOA enterprise3.example.com. hostmaster.example.com. 200402121 28800
7200 2419200 259200
[root@Enterprise3 root]#
```

The Reverse Zone

Now programs can convert the names in your-domain.com to real IP addresses. You're ready for the next step: a *reverse zone* file, which allows a DNS server to convert backwards, from an IP address to a hostname. Reverse zone lookups are used by many servers of different kinds (FTP, IRC, WWW, and others) to decide if they even want to talk to a computer asking for information. It's a common way for a mail server to check if an e-mail has come from a valid domain. Therefore, for full access to all Internet services, you need a reverse zone. Start by adding another zone to the `/etc/named.conf` configuration file:

```
zone "30.168.192.in-addr.arpa" IN {
    type master;
    file "example.com.rr.zone";
    allow-update { none; };
};
```

This is similar to the `0.0.127.in-addr.arpa` zone as described in `/etc/named.conf`. I've created an `example.com.rr.zone` file in the `/var/named` directory for the computers on my network, as shown in Figure 9-7.

Once again, restart **named** and examine the output of `host -l your-domain.com`. If the results do not look similar to the actual zone file, look for error messages in `/var/log/messages`.



Reverse zones can be used by several different services, such as sendmail and Apache. The reverse zone DNS database allows a server to verify if the name of a requesting computer matches its IP address, which can keep crackers from trying to “spoof” your system.

FIGURE 9-7

A reverse DNS
zone file

```
$TTL      3D
@         IN      SOA      enterprise3.example.com. hostmaster.example.com. (
                                200402121      ; serial number
                                28800           ; refresh frequency
                                14400           ; retry frequency
                                3600000         ; expiration period
                                3D              ; minimum TTL
)
         IN      NS       enterprise3.example.com
1         IN      PTR      allaccess.example.com
2         IN      PTR      enterprise3.example.com
5         IN      PTR      entwks3.example.com
~
"example.com.rr.zone" 12L, 587C
```

Starting named

Make sure your computer is connected to a network. Now you can start your DNS server through the **named** daemon with the **service named start** command. View the syslog message file (usually called `/var/log/messages`) with the **tail -f /var/log/messages** command. If there are problems, you'll see error messages here.

If there are any error messages, the **named** daemon will display the file with the error. Stop the service with the **service named stop** command and check the applicable configuration files.

Now test the setup. Use the **dig** command to examine your work. For example, if you use **dig** to look up the address of `www.redhat.com`, you'll see something like the output shown in Figure 9-8.

The **dig** command asks your DNS server to look for the `www.redhat.com` server. It then contacts one of the name server computers listed in `/etc/resolv.conf`. If that doesn't work, it goes to one of the name servers listed in the `named.ca` file and makes its requests from there. The request may be passed on to other DNS servers. Therefore, it can take some time before you see an answer.

Once you're satisfied with your configuration, you'll want to make sure that DNS starts the next time you reboot Linux. Naturally, that may be the only way that you would get full credit for your work on the RHCE exam. The following command makes sure that the **named** daemon starts the next time you boot Linux in the standard login runlevels:

```
# chkconfig --level 35 named on
```

FIGURE 9-8

DNS query
using dig

```
[root@Enterprise3 root]# dig www.redhat.com

; <<>> DiG 9.2.2 <<>> www.redhat.com
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 31616
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 3, ADDITIONAL: 3

;; QUESTION SECTION:
;www.redhat.com.                IN      A

;; ANSWER SECTION:
www.redhat.com.                600     IN      A      216.148.218.197
www.redhat.com.                600     IN      A      216.148.218.195

;; AUTHORITY SECTION:
redhat.com.                    600     IN      NS      ns1.redhat.com.
redhat.com.                    600     IN      NS      ns2.redhat.com.
redhat.com.                    600     IN      NS      ns3.redhat.com.

;; ADDITIONAL SECTION:
ns1.redhat.com.                300     IN      A      66.187.233.210
ns2.redhat.com.                600     IN      A      66.77.185.41
ns3.redhat.com.                600     IN      A      63.240.14.66

;; Query time: 214 msec
;; SERVER: 207.217.126.81#53(207.217.126.81)
;; WHEN: Wed Jan 21 10:39:01 2004
;; MSG SIZE rcvd: 166

[root@Enterprise3 root]#
```

Common DNS Pitfalls

DNS is an Internet-wide database of domain names and IP addresses. If you want your DNS server to participate, make sure the information that goes into the database is up to date and properly formatted. Many network outages can be traced to poorly administered DNS servers. A few examples of common DNS errors are described in the following sections.

Timing

Sometimes, all you need to do is wait a while. Whenever a change is made in a DNS database, it takes some time before the change is noted (a.k.a. propagated) to other DNS servers on the Internet. Therefore, whenever you change something such as the IP address associated with a Web server, it's advisable to keep the old IP address available for that Web server until the new IP address has time to propagate.

The Serial Number Wasn't Incremented

The single most common DNS error occurs when an administrator makes updates to a zone file, restarts DNS, and notices that no one else on the Internet knows about the

updates. If another DNS server doesn't detect a new serial number on a zone file, it assumes the file is the same, and sticks with its cache. No data is taken from the update, and other DNS servers don't get the revised information.

When you update a zone file, update the serial number. It's best done with the current date in the format shown in the examples (four-digit year, two-digit month, and two-digit date, followed by a one-digit increment number). If you've updated the DNS more than once today, increment the last number as well.

The Reverse Zone Isn't Delegated

Not all network administrators have control over their DNS servers. Some administrators contract with an ISP for this service.

Now assume you're that ISP administrator. You'll need to assign this customer a range of IP addresses for their domain name. Then you'll need to assign the domain name and IP addresses to a specific DNS "zone of authority."

Next, you'll also need to set up the reverse zone. For example, if you assign the 192.168.1. network (192.168.1. is short for the IPv4 network address of 192.168.1.0), you'll need to add NS records in the forward zone, and PTR records in the reverse zone.



From an end-user perspective, DNS might be considered the glue that holds the Internet together. Pay special attention to the nuances of the configuration files, so that network-wide problems are avoided.

The Red Hat DNS Configuration Tool

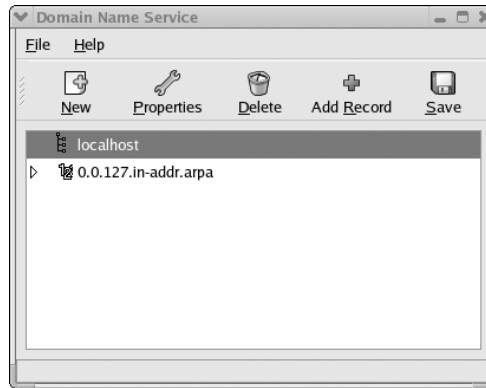
Red Hat has created a number of excellent GUI configuration tools. They are "front-ends" which can help many administrators create the configuration files that they need. Unfortunately, I do not believe that the Red Hat Domain Name Service tool should be included in this category. As of this writing, Red Hat has incorporated changes into a different RPM package optimized for Fedora Linux; this updated RPM package, `system-config-bind`, is not supported by Red Hat through an enterprise subscription.

In any case, it is best if you learn how to configure Linux services, including DNS, directly from the configuration files. As a Linux systems administrator, you may not always have access to the GUI. You may need to administer servers remotely, which makes GUI configuration difficult at best.

If you want to try the Red Hat Domain Name Service configuration tool shown in Figure 9-9, back up your DNS configuration files first: `/etc/named.conf`, `/etc/named.custom`, and the files in the `/var/named` directory. In RHEL 3, this is based on the `redhat-config-bind` RPM.

FIGURE 9-9

The Red Hat
Domain Name
Service
configuration tool



Keep It Working

Assuming you're running your own DNS, you also need to keep the `/var/named/named.ca` file up to date. The easiest way to do this is by using **dig**, which is also known as the DNS Information Groper. If your DNS is working properly and is connected to the Internet, you can run commands such as **dig www.osborne.com**.

Let us analyze the **dig** command further. First, run **dig** with no arguments. You will get information from the local `named.ca` file. Then use this command to query one of the listed root servers with a command such as **dig a.ROOT-SERVERS.NET**. The output should resemble a `named.ca` file. Save it to a file with a command such as **dig @a.root-servers.net.ns > named.ca.new**) and replace the old `named.ca` file. Remember to reload the **named** daemon after replacing the `named.ca` file.

Alternatively, the following script can be run automatically to update `named.ca`. The text of the base script is also available in the DNS-HOWTO available from the Linux Documentation Project at www.tldp.org. At the time of this writing, the base script in the HOWTO is still written for BIND 8. If you're using this file, change all of the "root.hints" filenames to "named.ca", and then replace the mail-alias "hostmaster" with a working e-mail address.

Once you're satisfied with the configuration, you can set up a crontab entry to run this script on a monthly basis in the `/etc/cron.monthly` directory.

```
#!/bin/sh
#
# Update the nameserver cache information file once per month.
# This is run automatically by a cron entry.
#
# Original by Al Longyear
```

```

# Updated for bind 8 by Nicolai Langfeldt
# Miscellaneous error-conditions reported by David A. Ranch
# Ping test suggested by Martin Foster
#
(
  echo "To: hostmaster <hostmaster>"
  echo "From: system <root>"
  echo "Subject: Automatic update of the named.ca file"
  echo

  PATH=/sbin:/usr/sbin:/bin:/usr/bin:
  export PATH
  cd /var/named

  # Are we online? Ping a server on the Internet
  case 'ping -qnc 1 www.redhat.com' in
    *'100% packet loss'*)
      echo "The network is DOWN. named.ca NOT updated"
      echo
      exit 0
      ;;
  esac

  dig @A.ROOT-SERVERS.NET . ns > named.ca.new 2>>&1

  case 'cat named.ca.new' in
    *Got answer*)
      # It worked
      :;;
    *)
      echo "The named.ca file update has FAILED."
      echo "This is the dig output reported:"
      echo

      cat named.ca.new
      exit 0
      ;;
  esac

  echo "The named.ca file has been updated to contain the following
  information:"
  echo
  cat named.ca.new

  chown root.root named.ca.new

```

```
chmod 444 named.ca.new
rm -f named.ca.old
mv named.ca named.ca.old
mv named.ca.new named.ca.hints
/etc/rc.d/init.d/named restart
echo
echo "The nameserver has been restarted to ensure that the update
    is complete."
echo "The previous named.ca file is now called
/var/named/named.ca.old."
) 2>>&1 | /usr/lib/sendmail -t
exit 0
```

EXERCISE 9-1

Set Up Your Own DNS Server

Following the example files shown previously, set up your own DNS server. Set it up to serve the domain called rhce.test. As long as your domain is private, it doesn't matter that rhce.test does not match the standard domain name types such as .com or .net.

1. Edit the /etc/named.conf file to reflect the configuration files that you plan to use. Name the zone file rhce.test.zone and set it to be a master domain.
 2. Edit the file /var/named/rhce.test.zone and place the proper zone information in it. Start by adding in the header with the serial number and expiration information.
 3. Add the SOA resource record (RR) with a proper administrative e-mail address contact.
 4. Add NS and MX RRs for the domain. Use the 192.168.*.* address range. If you're configuring an actual TCP/IP network with static IP addresses, feel free to use the assigned IP addresses on your network.
 5. Add several hosts to the zone file. Use WWW, FTP, and mail for a few.
 6. Save the zone file and then restart named with the **service named restart** command.
 7. Use the **dig** command to check the rhce.test domain. If it works, you have a working DNS server.
-

CERTIFICATION OBJECTIVE 9.02**Network File System (NFS)**

The Network File System (NFS) is the standard for sharing files and printers on a directory with Linux and Unix computers. It was originally developed by Sun Microsystems in the mid-1980s. Linux has supported NFS (both as a client and a server) for years, and NFS continues to be popular in organizations with Unix- or Linux-based networks.

You can create shared NFS directories directly by editing the `/etc/exports` configuration file, or you can create them with Red Hat's NFS Configuration tool. As you need an NFS server before you can configure an NFS client, I describe how to create NFS servers first.

exam
watch

It is a good skill for all Linux administrators to know how to connect to a shared NFS directory. While the Red Hat Exam Prep guide does not explicitly require RHCT candidates to

have this knowledge, it is consistent with the spirit of that exam. The Red Hat Exam Prep guide does explicitly require that RHCE candidates know how to configure an NFS server.

NFS Server Configuration and Operation

NFS servers are relatively easy to configure. All that is required is to export a filesystem, either generally or to a specific host, and then mount that filesystem from a remote client. I've shown you how to configure an NFS server to install RHEL 3 over a network. In this chapter, you'll learn the basics of NFS server configuration and operation.

Required Packages

Two RPM packages are closely associated with NFS: `portmap` and `nfs-utils`. They should be installed by default in RHEL 3. Just in case, you can use the `rpm -q packagename` command to make sure these packages are installed. The `rpm -ql packagename` command

provides a list of files installed from that package. The `nfs-utils` package includes a number of key files. The following is not a complete list:

- `/etc/rc.d/init.d/nfs` (control script for NFS)
- `/etc/rc.d/init.d/nfslock` (control script for `lockd` and `statd`)
- `/usr/share/doc/nfs-utils-1.0.5` (documentation, mostly in HTML format)
- Server daemons in `/usr/sbin`: `rpc.mountd`, `rpc.nfsd`
- Server daemons in `/sbin`: `rpc.lockd`, `rpc.statd`
- Control programs in `/usr/sbin`: `exportfs`, `nfsstat`, `nhfsgraph`, `nhfsnums`, `nhfsrun`, `nhfsstone`, `showmount`
- Status files in `/var/lib/nfs`: `etab`, `rmtab`, `statd`, `state`, `xtab`

The `portmap` RPM package includes the following key files (also not a complete list):

- `/etc/rc.d/init.d/portmap` (control script)
- `/usr/share/doc/portmap-4.0` (documentation)
- Server daemon in `/sbin`: `portmap`
- Control programs in `/usr/sbin`: `pmap_dump`, `pmap_set`

Configuring NFS to Start

Once configured, you can set up NFS to start during the Linux boot process, or you can start it yourself with the **service `nfs` start** command. NFS also depends on the `portmap` package, which helps secure NFS directories that are shared through `/etc/exports`. Because of this dependency, make sure to start the `portmap` daemon before starting NFS, and don't stop it until after stopping NFS.



Remember that both the `portmap` and `nfs` daemons must be running before NFS can work.

The `nfs` service script starts the following processes:

- **`rpc.mountd`** Handles mount requests
- **`nfsd`** Starts an `nfsd` kernel process for each shared directory
- **`rpc.rquotad`** Reports disk quota statistics to clients

If any of these processes are not running, NFS won't work. Fortunately, it's easy to check for these processes. Just run the **rpcinfo -p** command. As with other service scripts, if you want it to start when RHEL 3 boots, you'll need to run a command such as:

```
# chkconfig --level 35 nfs on
```

Alternatively, you can use the Red Hat Service Management utility described in Chapter 4 to make sure NFS starts the next time you boot RHEL 3.

Configuring NFS for Basic Operation

NFS is fairly simple. The only major NFS configuration file is `/etc/exports`. Once configured, you can export these directories with the **exportfs -a** command. Each line in this file lists the directory to be exported, the hosts it will be exported to, and the options that apply to this export. You can export a given directory only once. Take the following examples from an `/etc/exports` file:

```
/pub                (ro,sync) someone.mylocaldomain.com(rw,sync)
/home               *.mylocaldomain.com(rw,sync)
/opt/diskless-root  diskless.mylocaldomain.com(rw,no_root_squash,sync)
```

In the preceding example, the `/pub` directory is exported to all users as read-only. It is also exported to one specific computer with read/write privileges. The `/home` directory is exported, with read/write privileges, to any computer on the `.mylocaldomain.com` network. Finally, the `/opt/diskless-root` directory is exported with full read/write privileges (even for root users) on the `diskless.mylocaldomain.com` computer.

All of these options include the `sync` flag. This requires all changes to be written to disk before a command such as a file copy is complete. This is a fairly new change, which Red Hat first implemented on Red Hat Linux 8.0.

Be very careful with `/etc/exports`; one common cause of problems is an extra space between expressions. For example, if you type in a space after the comma in **(ro,sync)**, your directory won't get exported, and you'll get an error message.

Wildcards and Globbing

In Linux network configuration files, you can specify a group of computers with the right wildcard. This process in Linux is also known as *globbing*. What you do for a wildcard depends on the configuration file. The NFS `/etc/exports` file uses "conventional" wildcards;

for example, `*.mydomain.com` specifies all computers within the `mydomain.com` domain. In contrast, `/etc/hosts.deny` is less conventional; `.mydomain.com`, with the leading dot, specifies all computers in that same domain.

For IPv4 networks, wildcards often require some form of the subnet mask. For example, `192.168.0.0/255.255.255.0` specifies the `192.168.0.0` network of computers with IP addresses that range from `192.168.0.1` to `192.168.0.254`. Some services support the use of CIDR (Classless Inter-Domain Routing) notation. In CIDR, since `255.255.255.0` masks 24 bits, CIDR represents this with the number 24. If you're configuring a network in CIDR notation, you can represent this network as `192.168.0.0/24`. For details, see the discussion for each applicable service in Chapters 7 through 11.

Activating the List of Exports

Once you've modified `/etc/exports`, you need to do more. First, this file is simply the default set of exported directories. You need to activate them with the **`exportfs -a`** command. The next time you boot RHEL 3, if you've activated **`nfs`** at the appropriate runlevels, the `nfs` start script automatically runs the **`exportfs -r`** command, which synchronizes exported directories. You can see this for yourself in the `/etc/rc.d/init.d/nfs` script.

When you add a share to `/etc/exports`, the **`exportfs -r`** command adds the new directories. However, if you're modifying, moving, or deleting a share, it is safest to first temporarily unexport all filesystems with the **`exportfs -ua`** command before reexporting the shares with the **`exportfs -a`** command.

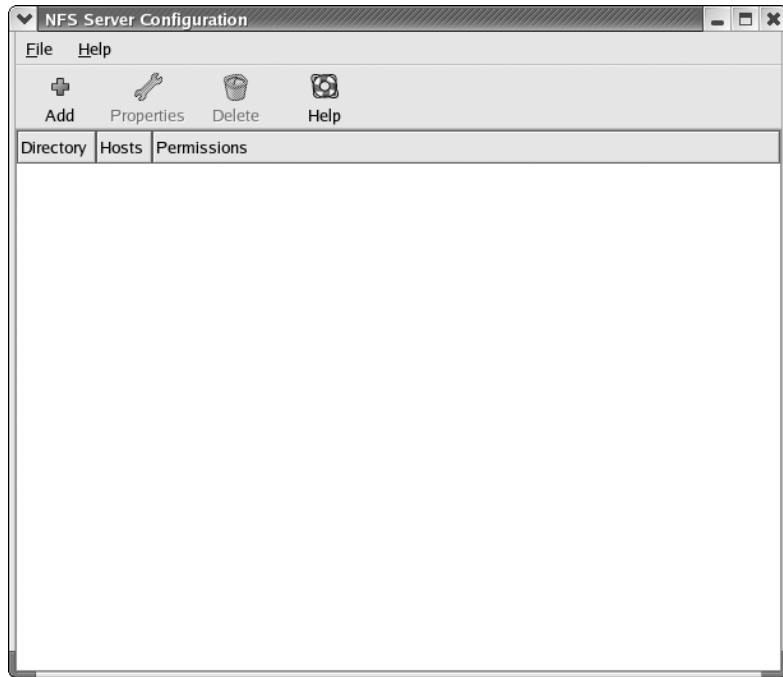
Once exports are active, they're easy to check. Just run the **`showmount -e`** command on the server. If you're looking for the export list for a remote NFS server, just add the name of the NFS server. For example, the **`showmount -e enterprise3`** command looks for the list of exported NFS directories from the `enterprise3` computer. If this command doesn't work, you may have blocked NFS messages with a firewall.

NFS Server Configuration Tool

Naturally, as it is easy to configure `/etc/exports`, the Red Hat NFS Server Configuration tool is easy to use and is reliable. To start this tool, type the **`redhat-config-nfs`** command in a GUI terminal, or click Main Menu | System Settings | Server Settings | NFS. This opens the NFS Server Configuration window shown in Figure 9-10. After you go through these steps, you'll see how much simpler it is to add a line to `/etc/exports`.

FIGURE 9-10

NFS Server
Configuration

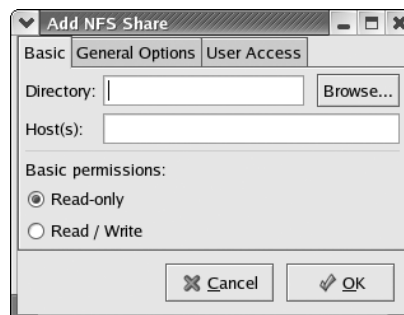


To add a shared NFS directory, take the following steps:

1. Click Add or File | Add NFS Share. This opens the Add NFS Share window shown in Figure 9-11.
2. Under the Basic tab, add the directory that you want to share. If you want to limit access to a specific host or domain, add the appropriate names to the

FIGURE 9-11

The Add NFS
Share window



Host(s) text box. If you want to allow access to all users, enter an asterisk (*). Set read-only or read/write permissions as desired. Click the General Options tab.

3. Under the General Options tab, you can set several parameters for this share, as described in Table 9-2. Note that **sync** is the only option that's active by default. The default is sufficient unless you receive specific instructions for an NFS share on your exam. Click the User Access tab.
4. Under the User Access tab, you can set different parameters for remote users, as defined in Table 9-3.
5. Once you've finished configuring your shared NFS directory, click OK. The directory is automatically exported with the **exportfs -r** command and, as long as you aren't blocking access with firewalls, should now be ready for use.
6. If you have problems, check for firewalls. Check for limitations in `/etc/hosts.allow` and `/etc/hosts.deny`. For more information on firewall management, read Chapter 10. If necessary, use the **service nfs stop** and **service nfs start** commands to restart the NFS service. If there are still problems, you may find more information in your `/var/log/messages` file.

exam

watch

Watch *If you use Red Hat's NFS Server Configuration tool, don't forget to activate NFS at the appropriate runlevels*

(3 and 5) so your shared directories are available when your instructor reboots your computer to see what you've done.

Option	Description
<code>-a</code>	Automatically mount the file system
<code>-b</code>	Block size (in bytes)
<code>-c</code>	Cache size (in bytes)
<code>-d</code>	Debugging options
<code>-e</code>	Export options
<code>-f</code>	Force mount
<code>-g</code>	Group ID
<code>-h</code>	Help
<code>-i</code>	Interval (in seconds)
<code>-j</code>	Journaling options
<code>-k</code>	Kernel options
<code>-l</code>	Link options
<code>-m</code>	Mount options
<code>-n</code>	No mount
<code>-o</code>	Options
<code>-p</code>	Permissions
<code>-q</code>	Quiet
<code>-r</code>	Read-only
<code>-s</code>	Size (in bytes)
<code>-t</code>	File system type
<code>-u</code>	User ID
<code>-v</code>	Verbose
<code>-w</code>	Write options
<code>-x</code>	Extended options
<code>-y</code>	Yield
<code>-z</code>	Zero

Parameter	Corresponding /etc/exports Command and Explanation
Allow connections from ports 1024 or higher	insecure As this sends NFS requests above ports 1024, it is not blocked by most firewalls.
Allow insecure file locking	insecure_locks If you have an older NFS client, this does not check user permissions to a file.
Disable subtree checking	no_subtree_check If you export a subdirectory such as /mnt/inst, this does not check higher level directories such as /mnt for permissions.
Sync write operations on request	sync Data is written upon request. Active by default.
Force sync of write operations immediately	no_wdelay Data is written to the share immediately.

TABLE 9-3 NFS Tool User Access Options

Parameter	Corresponding /etc/exports Command and Explanation
Treat remote root user as local root	no_root_squash Remote root users get root privileges on the shared directory.
Treat all client users as anonymous users	all_squash All remote users are mapped as an anonymous user. In RHEL 3, that user is nfsnobody, which you can see in /etc/passwd.
Specify local user ID for anonymous users	anonuid=userid Allows you to map remote users to a specific user ID such as pcgquest.
Specify local group ID for anonymous users	anongid=groupid Allows you to map remote groups to a specific group ID such as pcgquest.

Unfortunately, the NFS Server Configuration tool does not activate NFS at the appropriate runlevels for the next time you boot Linux. Therefore, the following exam watch is important.

NFS Client Configuration and Operation

Now you can mount a shared NFS directory from a client computer. The commands and configuration files are similar to those used for any local filesystem.

Mounting an NFS Directory from the Command Line

Before doing anything elaborate, you should test the shared NFS directory from a Linux or Unix client computer. But first, you should check for the list of shared NFS directories. If you're on an NFS server computer named enterprise3, the command is easy:

```
# showmount -e
```

This command assumes that the NFS server is local. If you don't see a list of shared directories, review the steps described earlier in this chapter. Make sure you've configured your /etc/exports file properly. Remember to export the shared directories. And your NFS server can't work if you haven't started the NFS daemon on your computer.

If you're on a remote NFS client computer and want to see the list of shared directories from the enterprise3 computer, run the following command:

```
# showmount -e enterprise3
```

If it doesn't work, there are a couple of more things that you'll need to check: firewalls and your /etc/hosts or DNS server. If you have a problem with your /etc/hosts or DNS

server, you can substitute the IP address of the NFS server. You'll see output similar to the following:

```
Export list for enterprise3
/mnt/inst *
```

Now if you want to mount this directory locally, you'll need an empty local directory. Create a directory such as `/mnt/remote` if required. You can then mount the shared directory from the `enterprise3` computer with the following command:

```
# mount -t nfs enterprise3:/mnt/inst /mnt/remote
```

This command mounts the `/mnt/inst` directory from the computer named `enterprise3`. This command specifies the use of the NFS protocol (`-t nfs`), and mounts the share on the local `/mnt/remote` directory. Depending on traffic on your network, this command may take a few seconds. Be patient! When it works, you'll be able to access files on `/mnt/inst` as if it were a local directory.

NFS and `/etc/fstab`

You can also configure an NFS client to mount a remote NFS directory during the boot process, as defined in `/etc/fstab`. For example, the following entry in a client `/etc/fstab` mounts the `/homenfs` share from the computer named `nfsserv`, on the local `/nfs/home` directory:

```
## Server: Directory Mount Point Type Mount Options Dump Fckorder
nfsserv:/homenfs /nfs/home nfs soft,timeout=100 0 0
```

Alternatively, an automounter, such as `autofs` or `amd`, can be used to dynamically mount NFS filesystems as required by the client computer. The automounter can also unmount these remote filesystems after a period of inactivity. For more information, see Chapter 4.

Client-Side Helper Processes

When you start NFS as a client, it adds a few new system processes, including:

- **rpc.statd** Tracks the status of servers, for use by `rpc.lockd` in recovering locks after a server crash
- **rpc.lockd** Manages the client side of file locking

Diskless Clients

NFS supports diskless clients, which are computers without a hard drive. A diskless client may use a boot floppy or a boot PROM to get started. Then embedded commands can mount the appropriate root (/) directory, swap space, the /usr directory as read-only, and other shared directories such as /home in read/write mode. If your computer uses a boot PROM, you'll also need access to DHCP and TFTP servers for network and kernel information.

Red Hat Enterprise Linux 3 includes features that support diskless clients. While not listed as part of the current Red Hat exam requirements or related course outlines, I would not be surprised to see such requirements in the future. You can find out more about how this works with the Network Installation and Diskless Environment tool, which you can start with the Main Menu | System Settings | Server Settings | Network Booting Service command.

Quirks and Limitations of NFS

NFS does have its problems. An administrator who controls shared NFS directories would be wise to take note of these limitations.

Statelessness

NFS is a “stateless” protocol. In other words, you don’t need to log in separately to access a shared NFS directory. Instead, the NFS client normally contacts `rpc.mountd` on the server. The `rpc.mountd` daemon handles mount requests. It checks the request against currently exported filesystems. If the request is valid, `rpc.mountd` provides an *NFS file handle* (a “magic cookie”), which is then used for further client/server communication for this share.

The stateless protocol allows the NFS client to wait if the NFS server ever has to be rebooted. The software waits, and waits, and waits. This can cause the NFS client to hang as discussed later.

This can also lead to problems with insecure single-user clients. When a file is opened through a share, it may be “locked out” from other users. When an NFS server is rebooted, handling the locked file can be difficult. The security problems can be so severe that NFS communication is blocked even by the default Red Hat Enterprise Linux firewall.

In theory, the recent change to NFS, setting up `sync` as the default for file transfers, should help address this problem. In theory, locked-out users should not lose any data that they’ve written with the appropriate commands.

Absolute and Relative Symbolic Links

If you have any symbolic links on an exported directory, be careful. The client interprets a symbolically linked file with respect to its own local filesystem. Unless the mount point and filesystem structures are identical, the linked file can point to an unexpected location, which may lead to unpredictable consequences.

You have a couple of ways to address this issue. You can take care to limit the use of symbolic links within an exported directory. Alternatively, NFS offers a server-side export option (**link_relative**) that converts absolute links to relative links; however, this can have counter-intuitive results if the client mounts a subdirectory of the exported directory.

Root Squash

By default, NFS is set up to **root_squash**, which prevents root users on an NFS client from gaining root access to a share on an NFS server. Specifically, the root user on a client (with a user ID of 0) is mapped to the `nfsnobody` unprivileged account.

This behavior can be disabled via the **no_root_squash** server export option in `/etc/exports`. In that case, root users who connect from a client gain root privileges on the shared NFS directory.

NFS Hangs

Because NFS is stateless, NFS clients may wait up to several minutes for a server. In some cases, an NFS client may wait indefinitely if a server goes down. During the wait, any process that looks for a file on the mounted NFS share will hang. Once this happens, it is generally difficult or impossible to unmount the offending filesystems. You can do several things to reduce the impact of this problem:

- Take great care to ensure the reliability of NFS servers and the network.
- Avoid mounting many different NFS servers at once. If several computers mount each other's NFS directories, this could cause problems throughout the network.
- Mount infrequently used NFS exports only when needed. NFS clients should unmount these clients after use.
- Set up NFS shares with the **sync** option, which should at least reduce the incidence of lost files.
- Don't configure a mission-critical computer as an NFS client, if at all possible.

- Keep NFS mounted directories out of the search path for users, especially that of root.
- Keep NFS mounted directories out of the root (/) directory; instead, segregate them to a less frequently used filesystem such as /nfs/home or /nfs/share.

Soft Mounting

Consider using the **soft** option when mounting NFS filesystems. When an NFS server fails, a soft-mounted NFS filesystem will fail rather than hang. However, this risks the failure of long-running processes due to temporary network outages.

In addition, you can use the **timeo** option to set a timeout interval, in tenths of a second. For example, the following command would mount /nfs/home with a timeout of 30 seconds:

```
# mount -o soft,timeo=300 myserver:/home /nfs/home
```

Inverse DNS Pointers

An NFS server daemon checks mount requests. First, it looks at the current list of exports, based on /etc/exports. Then, it looks up the client's IP address to find its hostname. This requires a reverse DNS lookup.

This hostname is then finally checked against the list of exports. If NFS can't find a hostname, rpc.mountd will deny access to that client. For security reasons, it also adds a "request from unknown host" entry in /var/log/messages.

File Locking

Multiple NFS clients can be set up to mount the same exported directory from the same server. It's quite possible that people on different computers end up trying to use the same shared file. This is addressed by the file locking daemon service.

NFS has historically had serious problems making file locking work. If you have an application that depends on file locking over NFS, test it thoroughly before putting it into production.

Filesystem Nesting

It is impossible to export two directories in the same filesystem if one is inside the other. For example, /usr and /usr/local cannot both be exported unless /usr/local is mounted on a separate partition from /usr.

Performance Tips

You can do several things to keep NFS running in a stable and reliable manner. As you gain experience with NFS, you might monitor or even experiment the following factors:

- Eight kernel NFS daemons, which is the default, is generally sufficient for good performance, even under fairly heavy loads. If your NFS server is busy, you may want to add additional NFS daemons through the `/etc/rc.d/init.d/nfs` script. Just keep in mind that the extra kernel processes consume valuable kernel resources.
- NFS write performance can be extremely slow, particularly with NFS v2 clients, as the client waits for each block of data to be written to disk.
- You may try specialized hardware with nonvolatile RAM. Data that is stored on such RAM isn't lost if you have trouble with network connectivity or a power failure.
- In applications where data loss is not a big concern, you may try the **async** option. This makes NFS faster because **async** NFS mounts do not write files to disk until other operations are complete. However, a loss of power or network connectivity can result in a loss of data.
- Hostname lookups are performed frequently by the NFS server; you can start the Name Switch Cache Daemon (`nscd`) to speed lookup performance.

on the
job

NFS is a powerful file-sharing system. But there are risks associated with NFS. If an NFS server is down, it could affect your entire network. It's also not sufficiently secure to use on the Internet. NFS is primarily used on secure LAN/WAN networks.

NFS Security

NFS includes a number of serious security problems and should never be used in hostile environments (such as on a server directly exposed to the Internet), at least not without strong precautions.

Shortcomings and Risks

NFS is an easy-to-use yet powerful file-sharing system. However, it is not without its problems. The following are a few security issues to keep in mind:

- **Authentication** NFS relies on the host to report user and group IDs. However, this can expose your files if root users on other computers access your NFS

shares. In other words, data that is accessible via NFS to *any user* can potentially be accessed by *any other user*.

- **Privacy** Not even Secure NFS encrypts its network traffic.
- **portmap infrastructure** Both the NFS client and server depend on the RPC portmap daemon. The portmap daemon has historically had a number of serious security holes. For this reason, portmap is not recommended for use on computers that are directly connected to the Internet or other potentially hostile networks.

Security Tips

If NFS *must* be used in or near a hostile environment, you can do some things to reduce the security risks:

- Educate yourself in detail about NFS security. If you do not clearly understand the risks, you should restrict your NFS use to friendly, internal networks behind a good firewall.
- Export as little data as possible, and export filesystems as read-only if possible.
- Use root squash to prevent clients from having root access to exported filesystems.
- If an NFS client has a direct connection to the Internet, use separate network adapters for the Internet connection and the LAN. Use the right firewall commands (**iptables** or **ipchains**) to block the routing on the TCP and UDP ports associated with portmapper, mountd, and nfsd.
- Use a firewall system such as **iptables** or **ipchains** to deny access to the portmapper, mountd, and nfsd ports, except from explicitly trusted hosts or networks. The ports are

111	TCP/UDP	portmapper	(server and client)
745	UDP	mountd	(server)
747	TCP	mountd	(server)
2049	TCP/UDP	nfsd	(server)

Use a port scanner to verify that these ports are blocked for untrusted network(s).

exam

Watch

While some may find it easier to learn with a GUI tool, these tools are usually more time consuming and less flexible than direct action from the

command line. As time is often short on the RHCE exam, I recommend that you learn how to configure and activate NFS and other services from the command line.

EXERCISE 9-2**NFS**

This exercise requires two computers, one set up as an NFS server, the other as an NFS client. On the NFS server:

1. Set up a group named IT for the Information Technology group in `/etc/group`.
2. Create the `/MIS` directory. Assign ownership to the MIS group with the `chgrp` command.
3. Set the SGID bit on this directory to enforce group ownership.
4. Update `/etc/exports` file to allow read and write for your local network. Run the following command to set it up under NFS.

```
# exportfs -a
```

5. Restart the NFS service.

On an NFS client, take the following steps:

6. Create a directory for the server share called `/mnt/MIS`.
7. Mount the shared NFS directory on `/mnt/MIS`.
8. List all exported shares from the server and save this output as `/mnt/MIS/thishost.shares.list`.
9. Make this service a permanent connection in the `/etc/fstab` file. Assume that the connection might be troublesome and add the appropriate options, such as soft mounting.
10. Reboot the client computer. Check to see if the share is properly remounted.
11. Test the NFS connection. Stop the service on the server, and then try copying a file to the `/mnt/MIS` directory. While the attempt to copy will fail, it should not hang the client.
12. Restart the NFS server.
13. Edit `/etc/fstab` again. This time assume that NFS is reliable, and remove the special options that you added in step 4.
14. Reboot the client computer. Test the service with the new settings.

Now test what happens when you shut down the server. The mounted NFS directory on the client should hang when you try to access the service. Restart the server service and see if your client service resumes.

EXERCISE 9-3

Using the NFS Server Configuration tool

In this exercise, you'll use the options associated with the NFS Server Configuration tool to experiment with creating a shared directory in `/etc/exports`. While it's best and usually fastest to edit a Linux configuration file directly, Red Hat GUI configuration tools such as the NFS Server Configuration tool can help you learn about different options for Linux services.

1. Open a GUI on a RHEL 3 computer. If not already open, you can do so with the **startx** command.
2. Start the NFS Server Configuration tool. You can run **redhat-config-nfs** from a command line interface, or click Main Menu | System Settings | Server Settings | NFS.
3. In the NFS Server Configuration tool, click Add. This opens the Add NFS Share window with the Basic tab. Set up a share for your home directory. Share it with one specific host on your LAN, with read-only permissions.
4. Click the General Options tab. Select the options of your choice. It does not matter what you select; the purpose of this lab is to demonstrate the effect of the NFS Server Configuration tool on the `/etc/exports` file.
5. Click the User Access tab. Select the options of your choice.
6. Click OK. The settings you choose are saved in `/etc/exports`.
7. Open a command line window. Right-click on the desktop, and select New Terminal from the pop-up menu that appears.
8. Open the `/etc/exports` file in the text editor of your choice. What is the relationship between the options you selected in the NFS Server Configuration tool and the command options associated with your home directory in `/etc/exports`? Close the `/etc/exports` file.

9. Back in the NFS Server Configuration tool, highlight the line associated with your home directory, and then click Properties. This opens the Edit NFS Share window with the settings that you just created.
10. Make additional changes under the three tabs in this window. After you click OK, check the results in `/etc/exports`. What happened?
11. If you don't want to actually export your home directory, highlight the appropriate line in the NFS Server Configuration tool and click Delete. What happens to `/etc/exports`?
12. Exit from the NFS Server Configuration tool.

CERTIFICATION OBJECTIVE 9.03

DHCP

There are two protocols that allow a client computer to get network configuration information from a server: DHCP (Dynamic Host Configuration Protocol) and bootp. DHCP works if you have a DHCP server on the local network. The bootp protocol is required if you're getting information from a DHCP server on another network.

DHCP servers can simplify and centralize network administration if you're administering more than a few computers on a network. They are especially convenient for networks with a significant number of mobile users. The bootp protocol is essentially just a way to access a DHCP server on a remote network.

As of this writing, Red Hat does not include any GUI tool to configure a DHCP server. You'll have to do your work in this section from the command line interface.

exam

Watch

While DHCP knowledge is not explicitly listed in the current Red Hat Exam Prep guide, it is a part of the associated curriculum. Based on their outlines, the RHCT course, RH133, teaches you how to configure a DHCP client. The

RHCE course, RH300, teaches you how to configure a DHCP server. It is important for any network administrator to know DHCP. You'll have to make your own decision regarding whether you'll need to study how to create a DHCP server for your exam.

Installing DHCP Packages

As with most network services, DHCP has a client and a server. These are based on the `dhcp-3.0pl2-6.14` and `dhclient-3.0pl2-6.14` RPM packages. The `dhclient` RPM package should be installed by default. The `dhcp` RPM package is installed by default with the Network Server package group. Install them if required.



Red Hat seems to change the commands and packages related to the DHCP client frequently. Recent versions of Red Hat have used `dhcpcd` and `pump` as DHCP client commands. If you are using a different version of Red Hat Linux, make sure you've installed the right packages and are using the right commands.

DHCP Server Configuration

A DHCP server sends messages to multiple computers on a LAN. This is also known as a multicast. It should be enabled by default in RHEL 3. You can confirm this with the `ifconfig` command. The output should resemble Figure 9-12, which includes a MULTICAST setting for the active network card.

If you don't see MULTICAST associated with your network card, someone has compiled this feature out of your kernel. For more information on the kernel management process, see Chapter 6.

Now configure the DHCP server daemon, `dhcpcd`, by creating or editing the `/etc/dhcpd.conf` configuration file. Normally, this file allows the DHCP server to randomly assign IP addresses from a specific range. If you don't already have an `/etc/dhcpd.conf` file, you can start with the `dhcpd.conf.sample` file in the `/usr/share/doc/dhcp-3.0-pl2`

FIGURE 9-12

Active network
interfaces
MULTICAST

```
[root@Enterprise3 root]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0C:29:1C:1E:9E
          inet addr:192.168.30.2  Bcast:192.168.30.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1961 errors:0 dropped:0 overruns:0 frame:0
          TX packets:415 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:762485 (744.6 Kb)  TX bytes:40639 (39.6 Kb)
          Interrupt:11 Base address:0x10c0

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:412 errors:0 dropped:0 overruns:0 frame:0
          TX packets:412 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:29010 (28.3 Kb)  TX bytes:29010 (28.3 Kb)

[root@Enterprise3 root]#
```

directory. The lines that start with a hash mark (#) are comments in the file. Let's analyze this sample file in detail:

- **ddns-update-style interim** With this command, the RHEL 3 DHCP server conforms as closely as possible to the new Dynamic DNS standard, where the DNS database is updated when the DNS server renews its DHCP lease. It is “interim” because the standards for DDNS are not complete as of this writing.
- **ignore client-updates** A good setting if you don't want to allow users on client computers to change their hostnames.
- **subnet 192.168.0.0 netmask 255.255.255.0** Describes a network with an address of 192.168.0.0 and a subnet mask of 255.255.255.0. This allows you to assign addresses in the range 192.168.0.1–192.168.0.254 to different computers on this network. Naturally, if you've configured a different network IP address, you'll want to change these settings accordingly.
- **option routers** Lists the default router. You can add additional **option routers** commands if you have more than one connection to an outside network. This information is passed to DHCP clients as the default gateway, which supports access to outside networks such as the Internet. Naturally, you'll want this command to reflect the IP address for the gateway for your network.
- **option subnet-mask** Specifies the subnet mask for the local network.
- **option nis-domain** Notes the server that provides the NIS shared authorization database. If you've configured NIS on your network, you'll want to substitute the name of your domain for domain.org. Otherwise, you should comment out this command.
- **option domain-name** Adds the domain name for your network. If your domain name is different, you'll want to change this command accordingly.
- **option domain-name-servers** Notes the IP address for the DNS server for your network. You can add more commands of this type to specify additional DNS servers.
- **option time-offset** Lists the difference from Greenwich Mean Time, also known as UTC (a French acronym), in seconds.
- **option ntp-servers** Notes any Network Time Protocol (NTP) servers for keeping the time on the local computer in sync with UTC. I describe NTP in more detail later in this chapter.

- **option netbios-name-servers** Adds the location of any Windows Internet Naming Service (WINS) servers for your network. As this is a Microsoft service, I only refer to it briefly in my description of Samba in Chapter 8.
- **option netbios-node-type 2** Peer-to-peer node searches, associated with WINS.
- **range dynamic-bootp 192.168.0.128 192.168.0.254** Notes the assignable IP addresses to *remote* networks, using the bootp protocol. Remember the space between the IP addresses!
- **default-lease-time** Specifies the lease time for IP address information, in seconds.
- **max-lease-time** Specifies the maximum lease time for IP address information, in seconds.
- **next-server** Notes the boot server for network computers. If you don't have any network computers, you can comment out this entire stanza.

You can also assign a specific IP address to a computer based on a client's Ethernet address. Just add an entry similar to the following to `/etc/dhcpd.conf`:

```
host dragonfire {
    hardware ethernet 08:00:12:23:4d:3f;
    fixed-address 192.168.0.201;
}
```

This assigns the IP address 192.168.1.201 to a client named dragonfire with the Ethernet hardware address 08:00:12:23:4d:3f.

Naturally, you'll want to modify this file accordingly for your particular network. For example, if you've configured computers on the example.com network described earlier in this chapter, you'll want to substitute example.com and the associated IP addresses in your `/etc/dhcpd.conf` file. I've done this for my network in Figure 9-13.



To assign an IP address to a specific network card on a specific computer, you need the hardware address, which can be found via the `ifconfig` command.

DHCP can be customized for individual computers. You can set up static IP addresses for servers. Once you're ready, start the `dhcpd` service with the following command:

```
# service dhcpd start
```

FIGURE 9-13

Sample DHCP
configuration file

```
ddns-update-style interim;
ignore client-updates;

subnet 192.168.30.0 netmask 255.255.255.0 {

# --- default gateway
    option routers                192.168.30.113;
    option subnet-mask            255.255.255.0;

    option nis-domain              "example.com";
    option domain-name            "example.com";
    option domain-name-servers    192.168.30.2;

    option time-offset             -18000; # Eastern Standard Time
#    option ntp-servers            192.168.1.1;
#    option netbios-name-servers   192.168.1.1;
# --- Selects point-to-point node (default is hybrid). Don't change this unless
# -- you understand Netbios very well
#    option netbios-node-type 2;

    range dynamic-bootp 192.168.30.128 192.168.30.254;
    default-lease-time 21600;
    max-lease-time 43200;

    # we want the nameserver to appear at a fixed address
    host ns {
        next-server enterprise3.example.com;
        hardware ethernet 00:0C:29:1C:1E:9E;
        fixed-address 192.168.30.2;
    }
}

"/etc/dhcpd.conf" 31L, 865C                                29,1-8      All
```

By default, this starts a DHCP server, which listens for requests on the eth0 network card. Alternatively, to have a DHCP server listen on the eth1 network interface, run the following command:

```
# service dhcpd start eth1
```

If these commands don't get a response, you probably haven't created a /etc/dhcpd.conf configuration file.

You can watch the DHCP server in action. Stop the DHCP server with the **service dhcpd stop** command. You can then restart it in the foreground with the following command:

```
# /usr/sbin/dhcpd -d -f
```



Start another Linux/Unix client. Make it look for another DHCP lease with the **dhclient -r** and **dhclient** commands, then watch the console of the server. You'll see a number of DHCP communication messages on the server that illustrates the process of leasing an IP address to a client.

Another easy way to get the MAC address for a given client is to watch the DHCP server messages.

Once you've configured your DHCP server to your satisfaction, remember to activate it at the appropriate runlevels. For example, the following command activates the DHCP daemon when you boot Linux into runlevels 3 and 5:

```
# chkconfig --level 35 dhcpd on
```

DHCP and Microsoft Windows

In order for the DHCP server to work correctly with picky DHCP clients such as Microsoft Windows 9x, the server needs to send data to the broadcast address: 255.255.255.255. Unfortunately, Linux insists on changing 255.255.255.255 into the local subnet broadcast address. The mixed message results in a DHCP protocol violation, and while Linux DHCP clients don't notice the problem, Microsoft DHCP clients do. Normally, such clients can't see DHCP OFFER messages and therefore don't know when to take an IP address offered from the DHCP server. If you're configuring a DHCP server for a network with Microsoft Windows computers, run the following command:

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

where eth0 is the name of the NIC that connects the server to the network.

Client Configuration

You can set up DHCP as a client using the **dhclient** command, or you can use the Red Hat Network Configuration tool (which you can also start with the Main Menu | System Settings | Network command). Alternatively, configuring a DHCP client at the command line is not difficult (and I believe is faster on the Red Hat exams). Make sure that the `/etc/sysconfig/network` configuration file includes the following line:

```
NETWORKING=yes
```

Next, make sure that the `/etc/sysconfig/network-scripts/ifcfg-eth0` script contains the following lines (if you're using a different network device, modify the appropriate file in `/etc/sysconfig/network-scripts` directory):

```
BOOTPROTO='dhcp'
ONBOOT='yes'
```

The next time you reboot, your network configuration should look for DHCP address information automatically from the DHCP server for your network.

on the
Ubuntu

There is a subtle difference in the Linux names for the DHCP client and server. The DHCP server daemon is `dhcpd`; the DHCP client daemon is `dhclient`. If you're experienced with older versions of Red Hat Linux, this can get even more confusing, as obsolete DHCP client daemons were named `dhcpcd` and `pump`.

Alternatively, you can use the Network Configuration tool from a GUI to configure DHCP. You can also start it from a GUI terminal console with the `redhat-config-network` command. When the tool opens, select your network card and click Edit. You should see a window similar to what is shown in Figure 9-14.

If you want to use DHCP on this computer, select the “Automatically obtain IP address settings with” option. You'll then get to choose between getting IP address information from a DHCP server on your local network, using bootp to get IP address information from a remote network, or going through a dialup connection, such as to an ISP. Once you've activated the changes, restart the network daemon with the `service network restart` command. Your network card will then look for IP address information from a DHCP server.

DHCP Client Troubleshooting

If the DHCP client configuration instructions in this chapter are not working, there may be a problem with the way the network is set up on your Linux computer. For example:

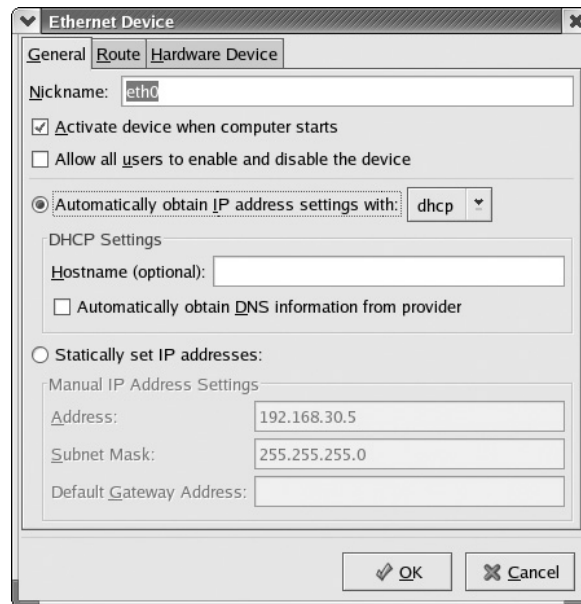
- The NIC is not configured properly. See Chapter 4 for information on reconfiguring your network card.
- If the computer is still having problems finding a DHCP server, check your firewall. If port 67 or 68 is blocked, your computer won't be able to get a message to the server.

on the
Ubuntu

The gateway daemon, starting with Red Hat 8.0, is `zebra`, which is now part of the `quagga` RPM. The `gated` and `zebra` RPM packages are now obsolete.

FIGURE 9-14

Configuring your
network card



EXERCISE 9-4

DHCP

To run this exercise, you'll need two different computers: a DHCP server and a DHCP client on the same LAN.

1. Open `/etc/dhcpd.conf`. Configure the server with an IP address range of 192.168.11.11–192.168.11.15 and with a network mask of 255.255.255.0. (If you want to use a different private IPv4 address subnet, substitute accordingly.)
2. Configure the client computer to use DHCP. Restart the network service on the client and record the IP address that it gets.
3. Add gateway and DNS server options with IP addresses of 192.168.11.254 and 12.34.45.56, respectively. If you already have a gateway and a DNS server, substitute the appropriate IP addresses. Restart the DHCP service. Restart networking on the client to make it renew the lease on the IP address.

CERTIFICATION OBJECTIVE 9.04

Keeping Servers in Sync

A number of companies keep servers in different time zones around the world. This allows users to select the server closest to them, maximizing network performance. As a Linux administrator, it may be your responsibility to make sure that all of these servers are on the same time. You can set up time synchronization on a centralized server through the Network Time Protocol (NTP).

The Network Time Protocol

NTP works by synchronizing a computer to UTC, through a connection to a time server. The current time is distributed through a hierarchy of NTP servers, through TCP/IP port 123. One point worth noting is that NTP will never run a system clock backward. If the clock on your computer is fast, NTP slows down the clock until it is in sync with UTC.

NTP Configuration

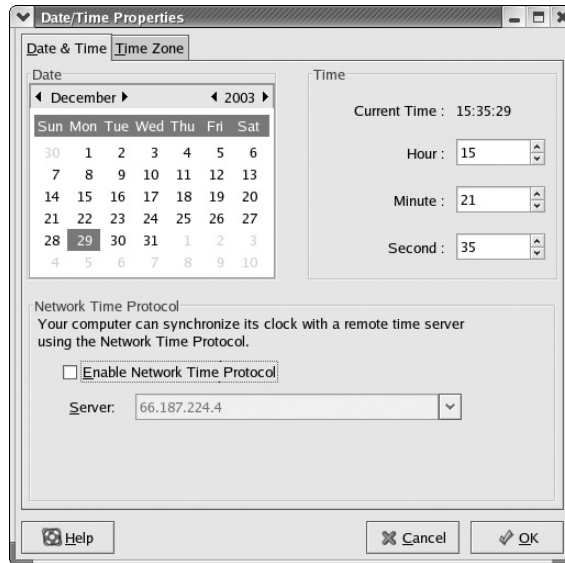
The configuration file for NTP is `/etc/ntp.conf`. In RHEL 3, it is already preconfigured to use your computer as a time server. But synchronizing your computer with your computer's hardware clock may not be enough.

To see how the default works, open the `/etc/ntp.conf` file in a text editor. You'll find a server that is listed with the **fudge** command. The default server is listed at IP address 127.127.1.0. While this isn't the normal "loopback" IP address, it is still within the "loopback" subnet; therefore, this address points right back at your computer. Alternatively, you can set it to one of the standard time servers described at www.eecis.udel.edu/~mills/ntp/servers.html.

Another way to synchronize your computer is to use the Red Hat Date/Time Properties tool to set NTP. You can start it in two ways. You can enter the **redhat-config-date** command in a GUI console, or click Main Menu | System Settings | Date & Time. This opens the Date/Time Properties dialog box shown in Figure 9-15.

When you select the Enable Network Time Protocol option and set a time server, `ntpd` is started and the name of the server is added to `/etc/ntp.conf`. If you want to set NTP to work after a reboot, you still need to make sure it starts the next time you reboot your computer with the appropriate **chkconfig** command.

FIGURE 9-15

Date/Time
Properties

CERTIFICATION SUMMARY

Networking services are an integral part of Red Hat Enterprise Linux. DNS, NFS, DHCP, and NTP are a few of the services that you can configure for this operating system.

DNS provides a database of domain names and IP addresses that help Web browsers and more find sites on the Internet. It's a distributed database, where each administrator is responsible for his or her own zone of authority. The diagnostic tool for DNS is now **dig**. You can also use the more traditional **nslookup** command, but it has been deprecated.

NFS allows you to share filesystems between Linux and Unix computers. This is a powerful method of controlling data and distributing I/O load, but there are many security concerns involved with its use. Be careful when setting up an NFS share on an unprotected network.

DHCP allows a network administrator to manage IP address assignments of the computers on a LAN from a centralized server. DHCP requires some specialized setup on both the client and the server; however, it is easy to maintain once it is configured.

NTP allows you to set up any computer to be synchronized to a central time standard, based on UTC.



TWO-MINUTE DRILL

Here are some of the key points from the certification objectives in Chapter 9.

DNS/BIND

- ☐ DNS, the Domain Name System, includes a database of computer names and IP addresses.
- ☐ DNS is based on the Berkeley Internet Name Domain (BIND), using the named daemon.
- ☐ Critical DNS configuration files include `/etc/named.conf` and the files in the `/var/named` directory.
- ☐ Caching-only DNS servers store requests and their associated IP addresses on a computer.
- ☐ Every time you change DNS, remember to update the serial number in your zone file. Otherwise, other DNS servers don't realize that you've changed anything.

Network File System (NFS)

- ☐ NFS is the standard for sharing files and printers between Linux and Unix computers.
- ☐ Key NFS processes are `rpc.mountd` for mount requests, `rpc.rquotad` for quota requests and `nfsd` for each network share.
- ☐ NFS shares are configured in `/etc/exports` and activated with the **`exportfs -a`** command.
- ☐ Clients can make permanent connections for NFS shares through `/etc/fstab`.
- ☐ If an NFS server fails, it can “hang” an NFS client. When possible, avoid using NFS on mission-critical computers.
- ☐ NFS and portmap have security problems. Limit their use when possible to secure internal networks protected by an appropriate firewall.

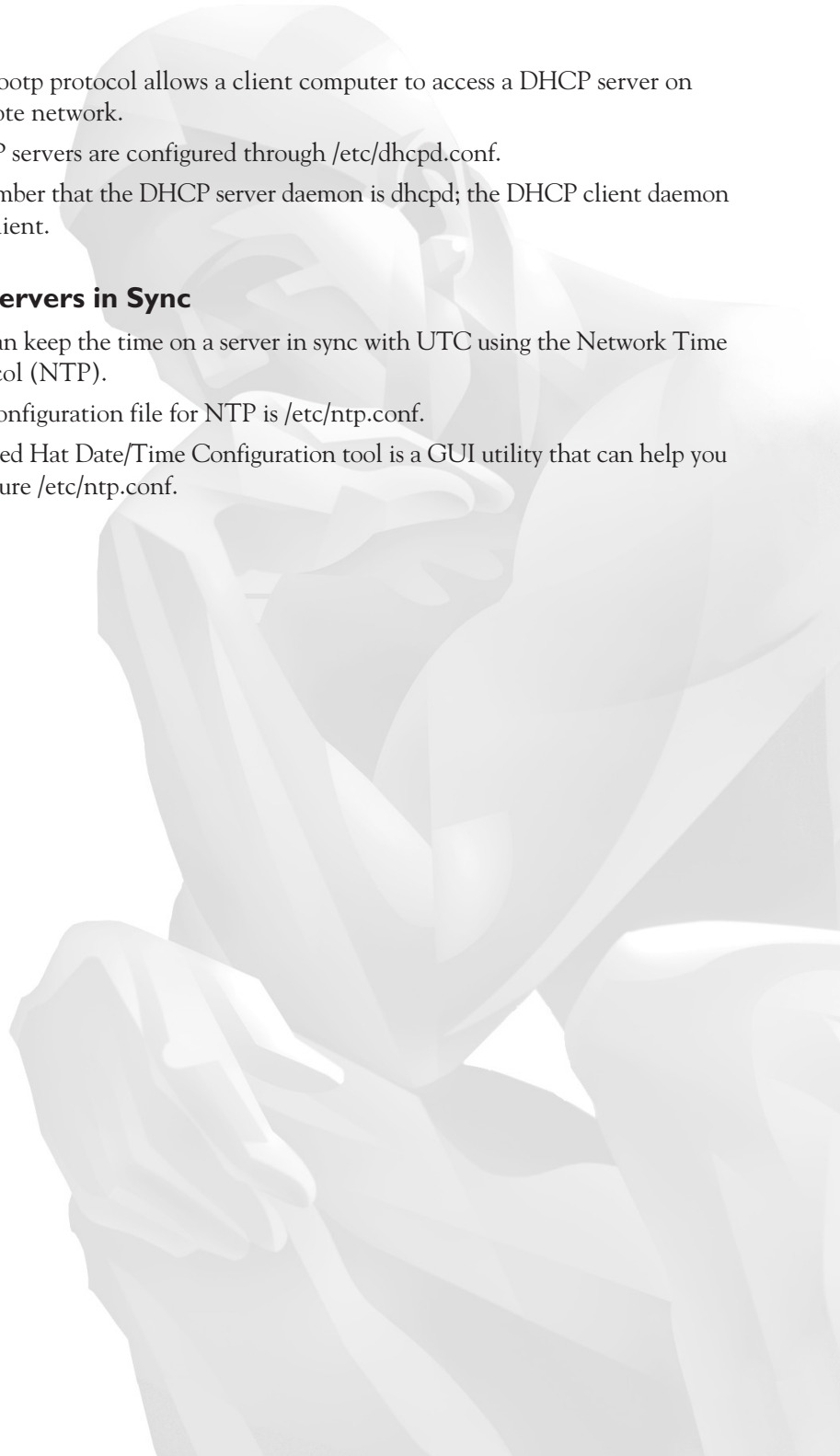
DHCP

- ☐ DHCP (Dynamic Host Configuration Protocol) allows a client computer to obtain network information (such as an IP number) from a server.

- ☐ The bootp protocol allows a client computer to access a DHCP server on a remote network.
- ☐ DHCP servers are configured through `/etc/dhcpd.conf`.
- ☐ Remember that the DHCP server daemon is `dhcpd`; the DHCP client daemon is `dhclient`.

Keeping Servers in Sync

- ☐ You can keep the time on a server in sync with UTC using the Network Time Protocol (NTP).
- ☐ The configuration file for NTP is `/etc/ntp.conf`.
- ☐ The Red Hat Date/Time Configuration tool is a GUI utility that can help you configure `/etc/ntp.conf`.



SELF TEST

The following questions will help you measure your understanding of the material presented in this chapter. Read all the choices carefully, as there may be more than one correct answer. Choose all correct answers for each question. Don't focus exclusively on these questions. There are no longer any multiple choice questions on the Red Hat exams. These questions test your understanding of the chapter. Getting results, not memorizing trivia, is what counts on the Red Hat exams.

DNS/BIND

1. Which of the following commands checks the setup of the database on your DNS server?
 - A. `dnscheck`
 - B. `BIND`
 - C. `dig`
 - D. `resolve`
2. You have added the addresses for several new servers into your primary DNS server. The zone files are formatted properly, and you've restarted the daemon called *named*. You advertise the new servers, and your help desk immediately starts getting calls that no one outside your domain can see the new servers. What is the most likely cause?
 - A. Your servers are not connected to the network.
 - B. The serial number was not incremented in the zone file.
 - C. Someone has changed the zone files without your knowledge.
 - D. The users at the other end are having ISP problems.
3. Which is an example of a properly formatted MX record?
 - A. `MX 10.mail.domain.com.`
 - B. `MX mail.domain.com.`
 - C. `MX 10 mail.domain.com`
 - D. `MX 10 mail.domain.com.`
4. Where will you find the database files associated with your DNS server?
 - A. `/etc/named`
 - B. `/var/named`
 - C. `/etc`
 - D. `/var/bind`

Network File System (NFS)

5. In the `/etc/exports` file, if we want to export the `/data` directory as read-only to all hosts and grant read and write permission to the host `superv` in `domain.com`, the proper entry is:
 - A. `/data (rw, sync) superv.domain.com (ro, sync)`
 - B. `/data (ro, sync) superv.domain.com (rw, sync)`
 - C. `/data (ro, async) *.domain.com (rw, async)`
 - D. `/data superv.domain.com (rw, async)`
6. When Linux boots, what configuration file does Linux use to see what NFS shares to mount?
 - A. `/etc/exports`
 - B. `/etc/nfs.conf`
 - C. `/etc/fstab`
 - D. `/nfs/conf`
7. Your company has just suffered an external security breach. As a result, the security group in your department has tightened the screws on all the servers, routers, and firewalls. Up until this point, all user data had been mounted over NFS, but now, nothing works. What happened?
 - A. The crackers erased the data from your NFS shared directories, and they erased the backups as well.
 - B. The NFS ports are no longer allowed through the necessary firewalls.
 - C. The two problems are unrelated. Make sure you have the disk space you need on your servers.
 - D. The filesystem is no longer shared from the server.
8. You're experiencing problems with NFS clients for various reasons, including frequent downtime on the NFS server and network outages between NFS clients and servers. Which of the following steps can help address this problem?
 - A. Remove all firewalls on individual computers inside your network.
 - B. Avoid setting up mission-critical computers as NFS clients.
 - C. Always make sure the portmap daemon is active before starting NFS.
 - D. Configure NFS with "soft" mounts.

DHCP

9. You add a new workstation to your `dhcpd.conf` file. You're in a hurry to finish, so you save and go to lunch. When you return, your phone mail is full of user complaints that they can't access

the Internet, but the local network is fine. You surmise that you accidentally changed something in the `dhcpd.conf` file. What is the most likely cause?

- A. A missing **routers** command.
- B. The subnet mask was changed.
- C. The IP range was thrown off.
- D. The broadcast address was changed.

Keeping Servers in Sync

10. Which are proper keywords that can be used in an `ntp.conf` file?

- A. server
- B. client
- C. sibling
- D. child

LAB QUESTIONS

Lab 1

Your internal network is growing, and you're having trouble keeping up with the different workstations that are being added to your network on a regular basis. You use the `good.example.com` subdomain for your internal network, and you've named your computers for your departments, such as `engr1` through `engr10.good.example.com`.

Your mail server is named `postal`, your Web server is named `www`, your FTP server is named `ftp`. You want to configure a DNS server on the computer named `names`. What do you need to do?

While you may not have enough information in this lab to create a complete and working file, you should be able to figure the outline of what you need to do, with the possible exception of specific IP addresses.

Lab 2

You'll need two Linux computers for this lab: one as an NFS server, a second as an NFS client. Let's call these computers `nfssvr.example.com` and `nfscient.example.com`. On the server, you'll want to share the `/home` directories, and provide write permissions to the client computer. On the client, you'll want to set up the `/home` directory from the NFS server to be mounted the next time you boot that client computer.

Lab 3

You'll also need two Linux computers for this lab: one as a DHCP server, a second as a DHCP client. Using the DHCP server created earlier in this chapter, set up a static IP address for the computer of your choice. You'll want to assign a specific name for that server, `precious.example.com`, and a special IP address on the 10.11.12.0 network, 10.11.12.13. Assume that you've already set up the `example.com` network as well as an appropriately configured DNS server.

Lab 4

Your network has more than 500 hosts with users in three major groups wanting to share their files within their groups. There are also 30 Windows XP clients in the publishing department that cannot use the Linux OS for their proprietary software needs. Everything is time-critical, as the outputs are related to stock quotes and therefore need to be synchronized to the same clock. What should you do?

SELF TEST ANSWERS

DNS/BIND

1. ☒ **C**. **dig** checks the configuration of the DNS server against the `resolv.conf` file.
☒ **A**, **B**, and **D** are incorrect. Neither **dsncheck** nor **resolve** are valid commands. BIND refers to the Berkeley Internet Name Domain, the basis for the DNS software.
2. ☒ **B**. Make absolutely sure that the serial number at the top of the zone file is changed each time you revise a zone file. Otherwise, other DNS servers won't know that anything has changed in your domain, and won't update their databases.
☒ **A** might be correct if something strange happened to your server, but the original server name still works internally, so your service is still connected to the network. **C** and **D** are also not likely scenarios, as root access is needed locally to edit the DNS configuration files and your ISP may or may not know the addresses of your server clients.
3. ☒ **D**. Make sure the preference is defined, and the trailing "." is included at the end of the record.
☒ **A**, **B**, and **C** are incorrectly formatted.
4. ☒ **B**. By default, the database files associated with a DNS server are stored in the `/var/named` directory.
☒ **A**, **C**, and **D** are incorrect. There are no default `/etc/named` or `/var/bind` directories, and only the standard DNS configuration files, `named.conf` and `named.custom`, are stored in the `/etc/` directory. While you could reconfigure `/etc/named.conf` to point to different directories for DNS database files, the default is `/var/named`.

Network File System (NFS)

5. ☒ **B**. This answer exports the filesystem as read-only to all users; it then specifies the computer that gets read/write permission.
☒ **A** provides read/write to all general hosts and gives the `superv.domain.com` computer read-only access. **C** provides read-only to all but allows all computers on the `domain.com` subnet to have read/write access. **D** gives read/write access only to the `superv.domain.com` computer. The **sync** and **async** options can both be appropriate under different circumstances.
6. ☒ **C**. `/etc/fstab` can be configured with all the necessary information for a computer to mount a shared NFS directory.
☒ **A** refers to `/etc/exports`, which lists the exported NFS directories. However, it does not itself mount an NFS directory. **B** and **D** are nonexistent files.

7. ☒ **B**. Ports 111, 745, 747, and 2049 must be open through firewalls to allow NFS to function. Since the security problems associated with NFS may have been to blame for the break-in, it is a good idea to restrict its use to isolated or protected subnets.
☒ While **A**, **C**, and **D** are all possibilities you should check for, but the most likely culprit is **B**.
8. ☒ **A**, **B**, **C**, and **D** are all to some extent correct. Ideally, you should be able to set NFS on computers inside a “trusted” network, which means that you can remove firewalls on computers inside that network. If you avoid setting up mission-critical computers on NFS, you’ll have fewer problems with NFS. The portmap daemon has to be running before NFS can work. To some extent, **D** is the best answer, because it can help an NFS client cope with downtime on an NFS server.
☒ None of the answers are incorrect.

DHCP

9. ☒ **A**. The lack of a router declaration in `/etc/dhcpd.conf` would keep your hosts from getting the gateway address, which is required to access the Internet from an internal network.
☒ **B**, **C**, and **D** would probably cause a general network outage.

Keeping Servers in Sync

10. ☒ **A** is correct. In `/etc/ntp.conf`, a server is an alternate time server.
☒ **B**, **C**, and **D** are incorrect. The **client**, **sibling**, and **child** commands are not relevant in the `ntp.conf` file.

LAB ANSWERS

Lab I

While you could subcontract out the task to an ISP, it’s easy to create a DNS server for your internal network. The basic files are already available on RHEL 3. All you need to do is modify these files, and add appropriate zone files to your `/var/named` directory. As there are problems with the Red Hat DNS Server Configuration tool, I’ll describe the basics on how you can set up a DNS server by directly editing the appropriate configuration files. Assume that you’re using the 10.11.12.0/255.255.255.0 network addresses for your LAN.

First, you'll need to modify the default `/etc/named.conf` configuration file. It's best to start by backing up this file. You'll need to add stanzas that refer to a zone and a reverse zone file. The stanzas are straightforward:

```
zone "good.example.com" IN {
    type master;
    file "good.example.com.zone";
};

zone "12.11.10.in-addr.apra" IN {
    type master;
    file "good.example.com.rr.zone";
    allow-update { none; }
};
```

Next, you can create the `good.example.com.zone` and `good.example.com.rr.zone` files in the `/var/named` directory. These files will contain a database of local and reverse local computer names and IP addresses for your LAN.

In the `good.example.com.zone` file, you'll want to create the forward database for your DNS server. It'll contain the records for your domain as well as the administrator e-mail address. There's not enough information in the problem to set up a full file, but the following principles apply:

You need to start the zone file with a general Time To Live (TTL) variable; for example, the following command sets a standard TTL (4 days) for data on this DNS server:

```
$TTL 4D
```

You'll need a Start Of Authority (SOA) record with the name of the DNS server and your administrative e-mail address. The format of the e-mail address is a little strange; the following line sets an e-mail address of `admin@good.example.com`. It also sets a serial number based on the date, a refresh (16 hours) and a retry frequency (4 hours), an expatriation period (2 weeks), as well as a TTL (4 days). Do note the dot at the end of each name:

```
@    IN    SOA      names.good.example.com. admin.good.example.com. (
                                200402121
                                16H
                                4H
                                2W
                                4D
```

Now you can specify the computers associated with the DNS and mail servers:

```
IN    NS      names.good.example.com.
IN    MX      10 postal.good.example.com.
```

Finally, you can specify the different computers on your network. While no specific IP addresses are given, you know that you have computers with the following names that you'll have in the good.example.com.zone file. I've added arbitrary IP addresses on the given IPv4 network. You'll have to find the proper IP addresses for yourself with **ifconfig** commands on each computer:

engr1	IN	A	10.11.12.1
engr2	IN	A	10.11.12.2
engr3	IN	A	10.11.12.3
engr4	IN	A	10.11.12.4
engr5	IN	A	10.11.12.5
engr6	IN	A	10.11.12.6
engr7	IN	A	10.11.12.7
engr8	IN	A	10.11.12.8
engr9	IN	A	10.11.12.9
engr10	IN	A	10.11.12.10
ftp	IN	A	10.11.12.11
www	IN	A	10.11.12.12
postal	IN	A	10.11.12.13

Finally, to make sure that the DNS server works the next time you boot this Linux computer, you'll want to set it to run at the appropriate runlevels with a command such as the following:

```
# chkconfig --level 35 named on
```

Lab 2

This lab is the first step towards creating a single /home directory for your network. Once you get it working on a single client/server combination, you can set it up on all clients and servers. You can then use the NIS server described in Chapter 10 for a single Linux/Unix database of usernames and passwords for your network. On the NFS server, you'll want to take the following steps:

1. Set up some users and special files that you'll remember in some of the user's home directories on the server. The details are not important—just make a note of what you've done.
2. Share the /home directory in /etc/exports. You'll want to share it with the nfsclient.example.com client. You can do this in this file with the following command:

```
/home nfsclient(rw,sync)
```

3. Export this directory with the following command:

```
# exportfs -a
```

4. Restart the NFS service:

```
# service NFS stop
# service NFS start
```

5. Make sure that the exported /home directory shows in the export list. On the local server, you can do this with the following command:

```
# showmount -e
```

6. If you have problems with any step in this process, make sure you don't have extra spaces in /etc/exports and that the NFS service is actually running with the **service nfs status** command. You may also want to check your firewall and make sure the appropriate services described in this chapter are running with the **rpcinfo -p** command.

7. Remember to make sure that the NFS server starts automatically the next time you boot that computer. One way to do so is with the following command:

```
# chkconfig --level 35 nfs on
```

Now on the NFS client, you'll want to take the following steps to connect to the shared /home directory:

8. First, you'll want to make sure that you can see the shared /home directory. If your DNS server is not working in any of these commands, you can substitute the IP address of the appropriate computer:

```
# showmount -e nfssvr.example.com
```

9. Now you'll want to mount the share that is offered on the local /home directory:

```
# mount -t nfs nfssvr.example.com:/home /home
```

10. Check to see that the mounting has worked. If it did, you'll see the NFS mount in the output to the **mount** command.
11. Now look through the mounted /home directory for the special files that you created in step 1. If you find them from the NFS client, you've succeeded in creating and connecting to the /home directory share.
12. To make the mount permanent, you'll want to add it to your /etc/fstab file. Once you've added a command such as the following to that file, the Linux client automatically mounts the shared /home directory from the NFS server.

```
nfssvr.example.com:/home    /home    nfs    soft,timeout=100    0    0
```

Lab 3

Assuming you've read the chapter, you've seen the template in the `dhcpd.conf.sample` configuration file for a static IP address:

```
host ns {
    next-server marvin.redhat.com
    hardware ethernet 12:34:56:78:AB:CD;
    fixed-address 207.175.42.254;
}
```

As described in the chapter, the **next-server** command is associated with the boot server for this computer; since there is no boot server mentioned, you won't need this command. To set up the DHCP server, take the following steps:

1. On the DHCP server computer, open the `/etc/dhcpd.conf` file. If this file doesn't exist, you haven't yet created a DHCP server on this computer.
2. Set up a new host in the DHCP configuration file:

```
host precious {
```

3. On the DHCP client, run the **ifconfig** command to find the hardware address associated with that computer's Ethernet network card. For the purpose of this exercise, assume it's `AB:CD:EF:12:34:56`; the host command line continues as follows:

```
    hardware ethernet AB:CD:EF:12:34:56
```

4. Finally, you can complete this line by setting up the static IP address that you want to assign to the DHCP client computer:

```
        fixed-address 10.11.12.13
    }
```

5. Save your changes to the `/etc/dhcpd.conf` configuration file. Restart the DHCP server daemon with the following command:

```
# service dhcpd restart
```

6. Now proceed to the DHCP client, the `precious.example.com` computer. You can release any current DHCP client with the following command:

```
# dhclient -r
```

7. Finally, you can see if the DHCP client actually takes the static IP address from the DHCP server with the following commands:

```
# dhclient
# ipconfig
```

Lab 4

You need to configure a few services on your central host. NIS can be used to manage all the users so that all hosts use the same user IDs. Then configure a central server with Samba and NFS and sufficient disk space for the four groups, restricting each service to members of each group only. Use NTP to synchronize the NFS server to an Internet time server, if available, and then have all the other hosts synchronize their time to the NFS server host on an hourly basis. As NIS is covered in the next chapter, I don't go into additional detail here.