RHCE™

RED HAT® CERTIFIED ENGINEER

10

Systems Administration and Security

CERTIFICATION OBJECTIVES

As a Red Hat Linux systems manager, you probably wear several hats, one of which is security manager. This is especially true if you work for a small company. Even if you work for a large organization that has a dedicated network or systems security staff, most of the administrators are probably responsible for other operating systems; you're probably responsible for security policies on your Linux systems.

You may spend very little time thinking about Linux security, or it may turn out to be a full-time job. For most Linux systems administrators, the amount of time spent on securing systems falls somewhere between these two extremes. The level of security you choose to configure depends on many factors, including the purpose of the system and the overall security policies of your company or organization, as well as the size and number of computers in the company.

For example, a Red Hat Linux system at home does not require as much security as a Red Hat Linux server that is being used to process credit card orders for a Web site.

Red Hat Linux comes with a large and varied assortment of tools for handling security. This includes tools for managing the security on individual Linux computers and tools for managing security for an entire network of systems, both Linux and otherwise. In this chapter, we look at some of the tools Red Hat Linux provides for managing security. We start out by looking at tools for controlling access to individual Linux host systems; then we look at tools for securing networks.

exam Watch

You'll need to know how to protect your computer and network. Sometimes this means you'll turn off, deactivate, or even uninstall a service. Other times, you'll set specific levels of security for different users. You can even regulate the type of traffic coming in, going out, and being transferred through your computer.

You have different ways to secure your system and network. The Network Information System (NIS) can provide a common database of authentication and configuration files for your network. The PAM (Pluggable Authentication Module) system lets you configure how users are allowed to log in or access different services. System logging often provides the clues that you need to solve a lot of problems. The Extended Internet Services Daemon governs a lot of services that do not have their own individual daemons. IP Aliases allow you to set up more than one IP address on a specific network card. With iptables, you can set up firewalls to accept or block many different kinds of network traffic. Network Address Translation allows you to protect computers inside your network by hiding their address information.

CERTIFICATION OBJECTIVE 10.01

Configuring NIS Clients

Generally, access to a Red Hat Linux system requires a valid username and password. One problem with a large network of Linux systems is that "normally," each user requires an account on every Linux computer.

The Network Information System (NIS) allows you to set up one centrally managed database of usernames and passwords for your Unix and Linux systems. With NIS, you can maintain one password database on an *NIS server* and configure the other systems on the network as *NIS clients.* When a user logs into an NIS client, that system first checks its local password file, usually /etc/passwd. If it can't find your username, it looks up the corresponding file on the NIS server.

NIS clients and NIS servers are organized in *NIS domains.* You can have multiple NIS domains on a single network, but clients and servers can belong to only one domain. If you are using NIS, you can find out the name of your NIS domain by using this command:

```
domainname
```

NIS domains are different from BIND domains. In fact, for security reasons, your NIS domain name should be different from your BIND domain name. If you are coming from the Microsoft Windows NT world, NIS domains are analogous to LAN manager domains.

NIS provides you with more than a shared authorization database. With NIS, you can provide shared access to any kind of information. By default, NIS under Red Hat Linux shares the following files:

- /etc/passwd
- /etc/group
- /etc/hosts
- /etc/rpc
- /etc/services
- /etc/protocols
- /etc/mail/*

You can configure NIS to share other files as well. This is easy to configure in the NIS configuration file, /var/yp/Makefile.

NIS services require at least one *NIS master server.* This is where the centralized NIS database files, known as *maps,* are stored. NIS changes require an update to the map on the master server. You can have only one NIS master server per NIS domain. (NIS maps are stored in the /var/yp/*DOMAIN* directory, where *DOMAIN* is the name of your NIS domain.)

For larger networks or redundancy, you may also want an *NIS slave server.* NIS slaves take copies of the NIS maps from the master server. NIS clients can then get their configuration files from either the master server or a slave server. You can have multiple NIS slave servers on a network.

NIS clients are systems that use information from an NIS server. NIS clients don't store any information that is contained in the NIS databases; whenever that information is needed, it is retrieved from a server.

exam Watch

You may notice that most NIS commands start with yp. This is a holdover from the previous name of NIS when it was known as the Yellow Pages service.

NIS Components on Red Hat Linux

The */usr/lib/yp* directory includes the utilities you need to configure and manage NIS services. The *ypinit* program can configure an NIS server. Table 10-1 lists the files needed to configure an NIS server.

Although NIS was designed to enable you to manage security by controlling who has access to the systems on your network, NIS is not a very secure product. Anyone who knows your NIS domain name and can connect to your network can read all the information stored in your NIS databases, such as /etc/passwd.

You can do a couple of things to help protect your NIS database. The */var/yp/securenets* file can control who can connect to your NIS server. This file is easy to configure. Only two lines are required for a LAN:

```
host 127.0.0.1
255.255.255.0  192.168.0.0
```

The first line allows access from the local computer. The second line may look a bit backward, but it allows access from all of the computers with IP addresses on the 192.168.0.0 network.

TABLE 10-1 NIS Configuration Files and Commands

File	Description
/usr/lib/yp/ypinit	Shell script to build initial database maps on an NIS server in /var/yp; **ypinit -m** builds the databases for a master server.
/var/yp/Makefile	Configuration file. Edit this file to control which files are shared via NIS. Implement the changes from the /var/yp directory with the *make* command.
/usr/sbin/ypserv	NIS server daemon. Remember to use /sbin/chkconfig to make sure it will start when you boot Linux.
/usr/sbin/yppasswdd	NIS password update daemon. Allows users to change their NIS passwords with the yppasswd command. Remember to use /sbin/chkconfig to make sure it starts when you boot Linux.
/etc/ypserv.conf	The ypserv daemon configuration file.
/var/yp/securenets	Controls which systems can access NIS databases. See the ypserv man page for an example.

Once you've configured an NIS server, it's easy to configure an NIS client. Just use *authconfig*. Figure 10-1 shows the authconfig screen used to configure NIS. This will configure your system to use the *ypbind* daemon, and add the appropriate entries in the /etc/yp.conf, /etc/nsswitch.conf, and /etc/pam.d/system-auth files. All you need is the name of the NIS domain, and the name of the computer where it's located.

The other command you need to know about when running an NIS client is *yppasswd*. All users can manage their NIS password with this command.

on the Job

One security risk to keep in mind if you use NIS is that anyone with access to the root account on any system that uses NIS can use the *su* - username ***command (note the space on both sides of the dash) to switch to any account in your NIS database.***

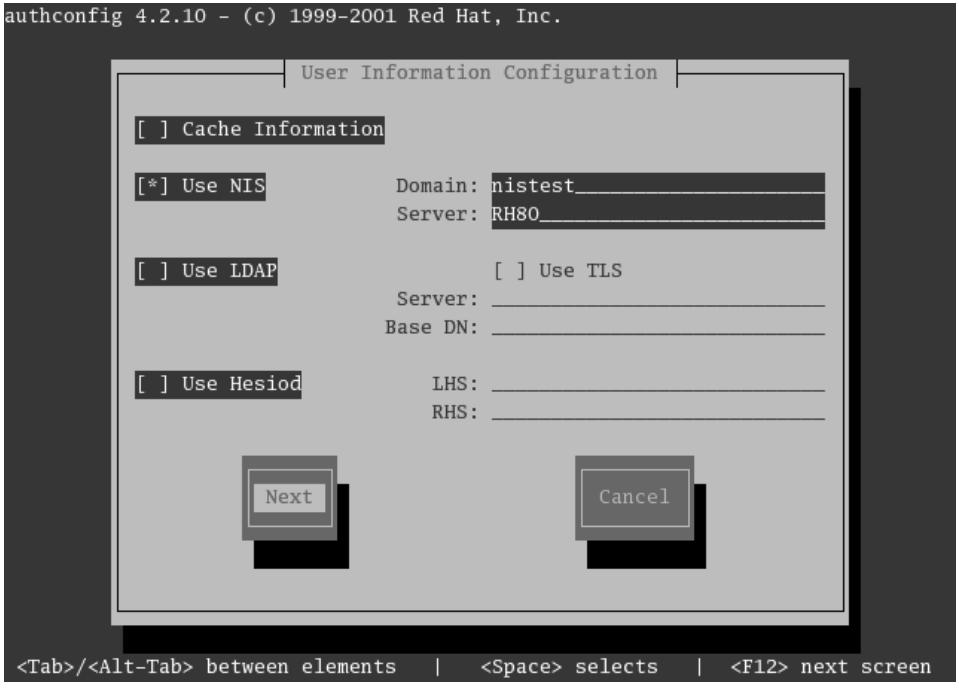
The Name Service Switch File

The Name Service Switch file (*/etc/nsswitch.conf*) governs the search order. For example, when an NIS client looks for a computer host name, it might start with the following entry from /etc/nsswitch.conf:

```
hosts: files nisplus nis dns
```

FIGURE 10-1

Configuring an NIS Client with authconfig



This line tells your computer to search through name databases in the following order:

1. Start with the database of host names and IP addresses in */etc/hosts*.
2. Next, search for the host name in a map file based on NIS+ (NIS Version 3).
3. Next, search for the host name in a map file based on NIS (Version 2).
4. If none of these databases includes the desired host name, refer to the DNS server.

CERTIFICATION OBJECTIVE 10.02

Basic Host Security

A network is only as secure as the most open system in that network. Although no system can be 100 percent secure, you can follow certain basic host measures to enhance the security on any given system and, consequently, your network. When

devising security measures, you have to plan for two types of security violations: user accidents and break-ins.

Accidents happen because users lack adequate training or are unwilling to follow procedures. If security is too burdensome, productivity may suffer, and your users will try to get around your rules. Password security falls into this category.

When a cracker breaks in to your system, some crackers may be looking for secrets such as credit card information. Others may just want to bring down your system. You can do several things to keep your network secure. Monitor Red Hat errata for the latest issues. With the up2date utility, you can keep your Red Hat system updated with the latest packages.

As you'll see later in this chapter, you can manage your computer's response to certain requests through the /etc/hosts.allow and /etc/hosts.deny files. You can set up protection within the kernel through firewalls based on iptables or ipchains. One simple way to promote security is to uninstall as many network access programs as possible.

Password Security

Good password security is important. Good passwords include a combination of letters and numbers that aren't easily guessed. Good password security requires users to change their password on a regular basis.

Password security also means disabling or deleting unused accounts. These accounts are a common way for a cracker to try to break into your system.

You can also check system log files for suspicious activity. Login records are kept in a database in /var/log/wtmp. While you can't read this file directly, you can use the utmpdump command to make this file readable. For example, the **utmpdump /var/log/wtmp** command lists recent login activity. Take a look at Figure 10-2. Note the login from IP address 172.132.4.8. If you don't have any users from a computer with that IP address, you have a reason for concern.

Security Updates

Another step you can take to keep your Red Hat Linux system secure is to install the latest errata releases from Red Hat. These contain patches or fixes for problems in applications or the operating system that could result in security violations. A list of the latest errata is available as of this writing at www.redhat.com/apps/support/errata.

FIGURE 10-2

Suspicious login activity

```
  ] [Wed Jun 26 12:18:39 2002 EDT]
[6] [01253] [5   ] [LOGIN   ] [tty5        ] [                ] [0.0.0.0
  ] [Wed Jun 26 12:18:39 2002 EDT]
[6] [01254] [6   ] [LOGIN   ] [tty6        ] [                ] [0.0.0.0
  ] [Wed Jun 26 12:18:39 2002 EDT]
[8] [01248] [ud  ] [        ] [            ] [2.4.18-3        ] [0.0.0.0
  ] [Wed Jun 26 12:18:39 2002 EDT]
[6] [01249] [1   ] [LOGIN   ] [tty1        ] [                ] [0.0.0.0
  ] [Wed Jun 26 12:18:39 2002 EDT]
[7] [01249] [1   ] [root    ] [tty1        ] [                ] [0.0.0.0
  ] [Wed Jun 26 12:24:57 2002 EDT]
[1] [13109] [~~  ] [runlevel] [~           ] [2.4.18-3        ] [0.0.0.0
  ] [Wed Jun 26 15:19:51 2002 EDT]
[5] [01954] [15  ] [        ] [            ] [2.4.18-3        ] [0.0.0.0
  ] [Wed Jun 26 15:19:51 2002 EDT]
[8] [01954] [15  ] [        ] [            ] [2.4.18-3        ] [0.0.0.0
  ] [Wed Jun 26 15:19:53 2002 EDT]
[5] [02080] [x   ] [        ] [            ] [2.4.18-3        ] [0.0.0.0
  ] [Wed Jun 26 15:19:53 2002 EDT]
[7] [02098] [:0  ] [mj      ] [:0          ] [                ] [172.132.4.8
  ] [Wed Jun 26 15:20:07 2002 EDT]
[7] [02277] [/0  ] [mj      ] [pts/0       ] [                ] [0.0.0.0
  ] [Wed Jun 26 15:20:48 2002 EDT]
[7] [02286] [/1  ] [mj      ] [pts/1       ] [                ] [0.0.0.0
  ] [Wed Jun 26 15:20:52 2002 EDT]
[8] [02278] [/1  ] [mj      ] [pts/1       ] [                ] [0.0.0.0
  ] [Wed Jun 26 15:20:56 2002 EDT]
--More--
```

Red Hat provides a built-in service to check for updates called up2date that you can configure if your computer is directly connected to the Internet. Just run up2date from a command line in the X Window of your choice. If you haven't already done so, you'll need to register the settings on your computer. Then follow the prompts; up2date connects to rhn.redhat.com for updates. A sample result is shown in Figure 10-3, which suggests an update to three packages, including the kernel.

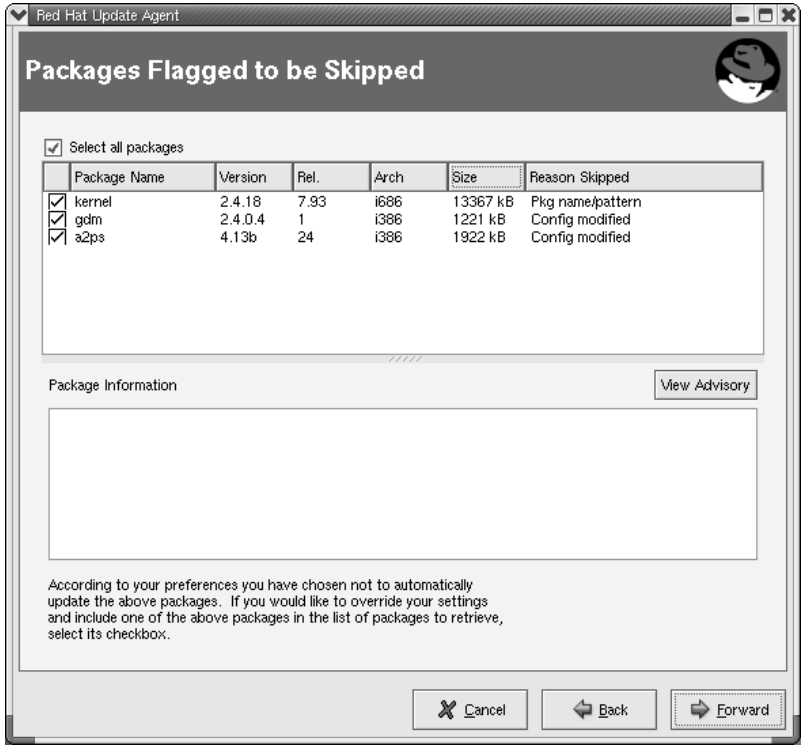
Delete Extra Services

One simple way to promote security on your system is to delete the packages associated with network services that you aren't going to use. For example, a cracker can't use Telnet to break into your system if the Telnet RPM is not installed. Any firewall or other configuration that you may add to the service still means that you are theoretically vulnerable to an attack through that service. If you're not going to use a network service, you may want to remove the associated RPM packages.

To review currently installed network services, check the /etc/xinetd.d, the /etc/rc.d/init.d directories.

FIGURE 10-3

up2date at work



CERTIFICATION OBJECTIVE 10.03

The Pluggable Authentication Module (PAM) System

Red Hat Linux uses the Pluggable Authentication Modules (PAM) system to check for authorized users. PAM includes a group of dynamically loadable library modules that govern how individual applications verify their users. You can modify PAM configuration files to suit your needs.

PAM was developed to standardize the user authentication process. For example, the login program uses PAM to require usernames and passwords at login. Open the /etc/pam.d/login file. Take a look at the first line:

```
auth  required    /lib/security/pam_securetty.so
```

This line means that root users can log in only from secure terminals as defined in the /etc/securetty file.

exam Watch

PAM modules are documented in the /usr/share/doc/pam-versionnumber/txts directory. For example, the functionality of the pam_securetty.so module is described in the README.pam_securetty file.

The configuration files shown in the /etc/pam.d directory are named after applications. These applications are "PAM aware." In other words, you can change the way users are verified for applications such as the console login program. Just modify the appropriate configuration file in /etc/pam.d.

Pluggable Authentication Modules (PAM) and Associated Files

The PAM system divides the process of verifying users into four separate tasks. These are the four different types of PAM modules:

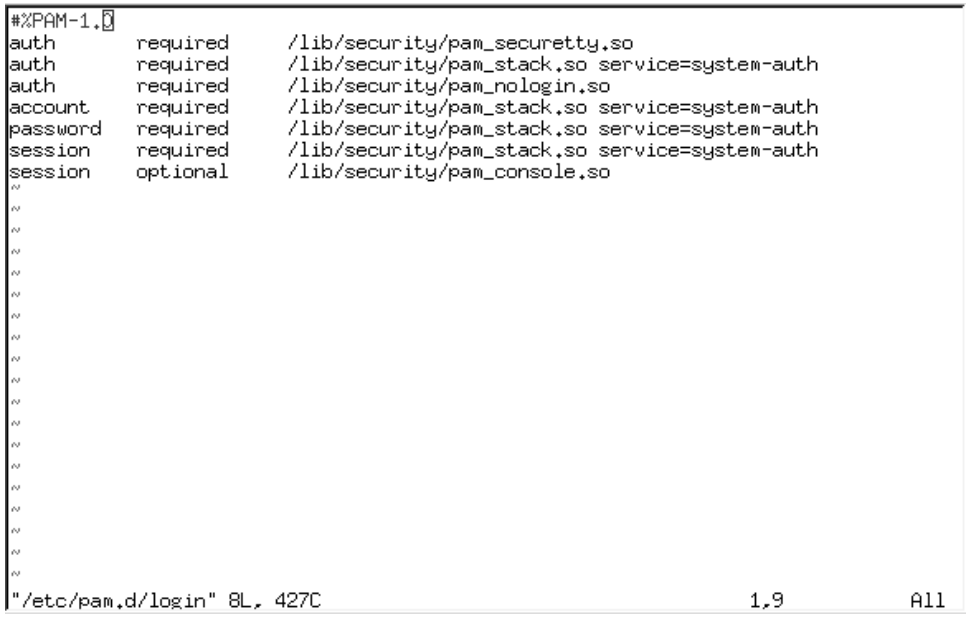
- **Authentication management** Establishes the identity of a user. For example, a PAM **auth** command may decide whether to prompt for a username and or a password.
- **Account management** Allows or denies access according to the account policies. For example, a PAM **account** command may deny access according to time, password expiration, or a specific list of restricted users.
- **Password management** Manages other password policies. For example, a PAM **password** command may limit the number of times a user can try to log in before a console is reset.
- **Session management** Applies settings for an application. For example, the PAM **session** command may set default settings for a login console.

The code shown in Figure 10-4 is an example PAM configuration file, /etc/pam.d/login. Every line in all PAM configuration files is written in the following format:

```
module_type  control_flag  module_path  [arguments]
```

The module_type, as described previously, is **auth**, **account**, **password**, or **session**. The control_flag determines what PAM does if the module succeeds or fails. The module_path specifies the location of the actual PAM module file. Finally, as with regular shell commands, you can specify arguments for each module.

FIGURE 10-4 The PAM login module



The control_flag field requires additional explanation. It determines how the configuration file reacts when a module flags success or failure. There are four different control flags, each described in Table 10-2.

To demonstrate how control flags work, take a look at the /etc/pam.d/reboot configuration file. The first **auth** command checks the **pam_rootok.so** module. If the root user runs the reboot command, and the control_flag is *sufficient,* the other **auth** commands in this file are ignored. Linux runs the reboot command.

TABLE 10-2 PAM Control Flags

control_flag	Description
required	If the module works, the command proceeds. If it fails, go to the next command in the configuration file—but the command will still fail.
requisite	Stop the process if the module fails.
sufficient	If the module works, the login or other authentication proceeds. No other commands need be processed.
optional	PAM ignores module success or failure.

The second **auth** command is run only for nonroot users; it just governs the console parameters. The module associated with the **account** command (pam_permit.so) accepts all users, even those who've logged in remotely. In other words, this configuration file would allow any root user, local or remote, to reboot your Linux computer.

```
#%PAM-1.0
auth       sufficient   /lib/security/pam_rootok.so
auth       required     /lib/security/pam_console.so
#auth       required     /lib/security/pam_stack.so service=system-auth
account    required     /lib/security/pam_permit.so
```

The third line is commented out by default. If you make this line active, it refers to the system-auth configuration file, which requires root user privileges. Remote users who know your root password are still allowed to reboot your computer.

Alternatively, you might add the pam_securetty.so module, which would keep remote users from rebooting your system. This module is described in more detail earlier in this chapter.

on the job

Allowing just any user to shut down a server system is* not *normal for corporate servers, but it is a commonly accepted practice on workstations. In this way, users can shut down their own laptop or desktop without having to change to the root account.

PAM Configuration Example: /etc/pam.d/login

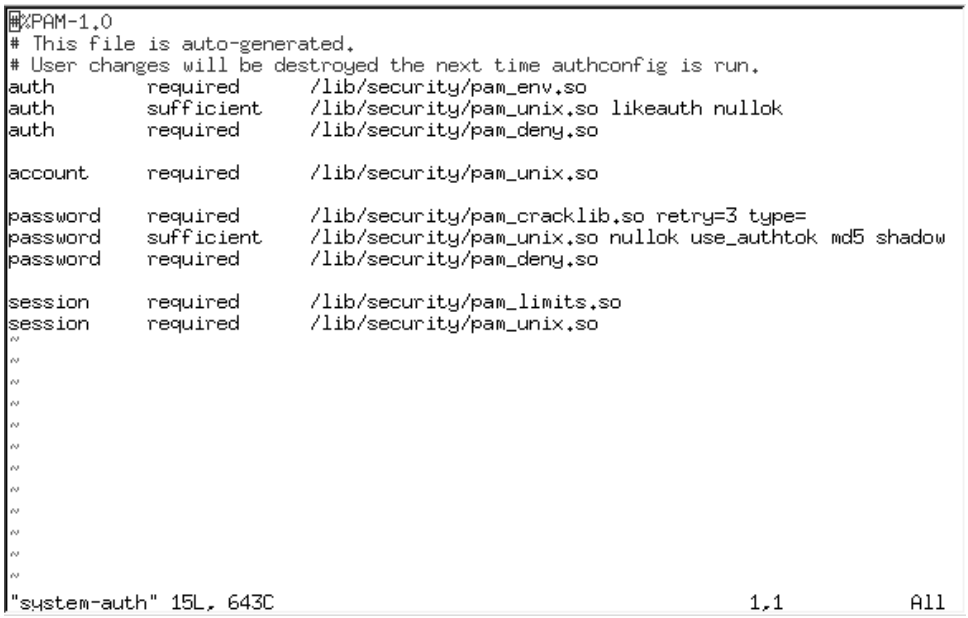
This section refers back to the /etc/pam.d/login configuration file shown in Figure 10-4. When a user opens a text console and logs in, Linux goes through this configuration file line by line. The first line in /etc/pam.d/login was already analyzed in the previous section. The next line brings the login program through the following service, system-auth, which also happens to be a PAM configuration file.

```
auth  required    /lib/security/pam_stack.so service=system-auth
```

Essentially, this calls the **auth** commands in the /etc/pam.d/system-auth configuration file shown in Figure 10-5. This sets up environment variables and allows different users to log in. The last **auth** line in /etc/pam.d/system-auth checks the /etc/nologin file. If this file exists, no regular users are allowed to log into your console.

FIGURE 10-5

The /etc/pam.d/system-auth configuration file



The **account** and **password** commands in /etc/pam.d/login also refer to the /etc/pam.d/system-auth configuration file. The **account** command in /etc/pam.d/system-auth refers to the pam_unix.so module, which sets up the normal username and password prompts. There are two **password** commands in /etc/pam.d/system-auth. The file shown in Figure 10-5 sets a maximum of three retries and sets up the use of md5 and shadow passwords.

exam Watch

The authconfig utility can modify the Linux password configuration through the /etc/pam.d/system-auth configuration file.

Finally, there are two **session** commands in the /etc/pam.d/login file. The first command refers to the /etc/pam.d/system-auth configuration file, which can allow you to set limits on individual users through /etc/security/limits.conf, and environment variables through the pam_unix.so module. The second command manages file permissions while users are logged onto your Linux computer.

EXERCISE 10-1

Configuring PAM

In this exercise, you can experiment with some of the PAM security features of Red Hat Linux.

1. Make a backup copy of /etc/securetty: cp /etc/securetty /etc/securetty.sav.
2. Edit /etc/securetty and remove the lines for tty3 through tty8. Save the changes and exit.
3. Use ALT-F3 (CTRL-ALT-F3 if you're running X Window) to switch to virtual console number 3. Try to log in as *root.* What happens?
4. Repeat this process as a regular user. What happens?
5. Use ALT-F2 to switch to virtual console number 2 and try to log in as root.
6. Restore your original /etc/securetty file: mv /etc/securetty.sav /etc/securetty.

exam Watch

Make sure you understand how Red Hat Linux handles user authorization through the /etc/pam.d configuration files. When you test these files, make sure you create a backup of* everything *in PAM before making any changes, because any errors that you make to a PAM configuration file can disable your system completely (it is that secure).

CERTIFICATION OBJECTIVE 10.04

System Logging

An important part of maintaining a secure system is keeping track of the activities that take place on the system. If you know what usually happens, such as understanding when users log into your system, you can use log files to spot unusual activity. Red Hat Linux comes with several utilities you can use to monitor activity on a system. These utilities can help you identify the culprit if there is a problem.

Red Hat Linux comes with two logging daemons. The kernel log daemon service, klogd, logs kernel messages and events. The syslog daemon, syslogd, logs all other process activity. You can use the log files that syslogd generates to track activities on your system. If you are managing multiple Red Hat Linux systems, you can configure the syslogd daemon on each system to log messages to a central host system.

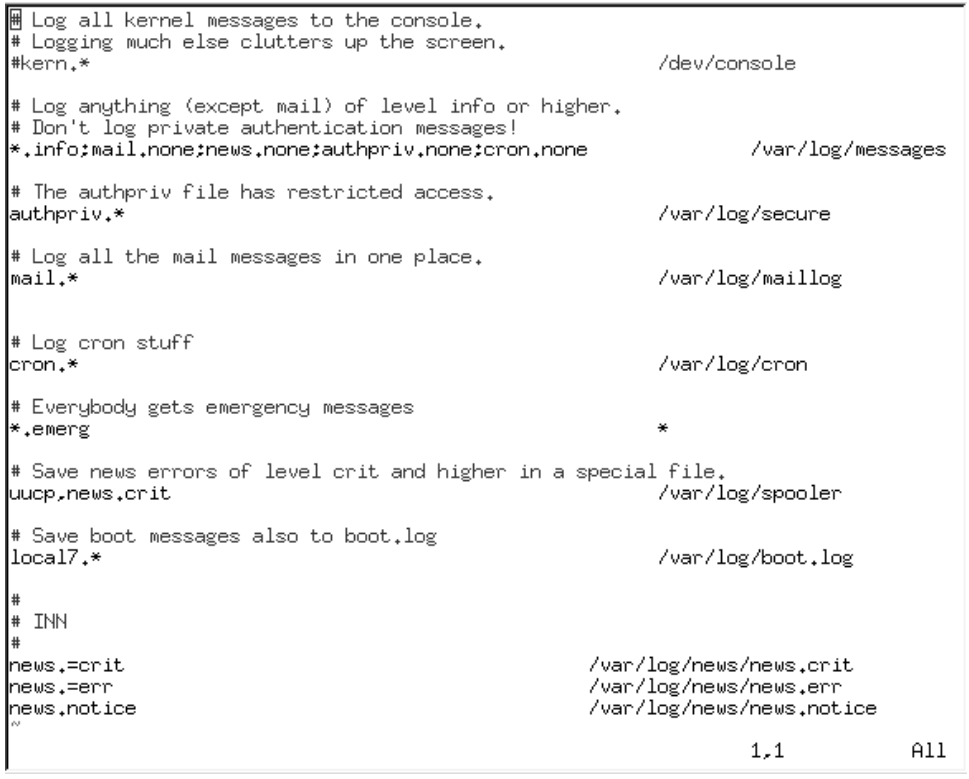
Both daemons are typically active by default, and both can be activated by the /etc/rc.d/init.d/syslog script. Once these daemons start, syslog examines /etc/syslog.conf to find configured logging options.

System Log Configuration File

You can configure what syslogd records through the /etc/syslog.conf configuration file. As shown in Figure 10-6, it includes a set of rules for different facilities: auth, auth-priv, cron, daemon, kern, lpr, mail, mark, news, security, syslog, user, and uucp.

FIGURE 10-6

The /etc/syslog.conf log configuration file



```
# Log all kernel messages to the console.
# Logging much else clutters up the screen.
#kern.*                                                 /dev/console

# Log anything (except mail) of level info or higher.
# Don't log private authentication messages!
*.info;mail.none;news.none;authpriv.none;cron.none              /var/log/messages

# The authpriv file has restricted access.
authpriv.*                                              /var/log/secure

# Log all the mail messages in one place.
mail.*                                                  /var/log/maillog


# Log cron stuff
cron.*                                                  /var/log/cron

# Everybody gets emergency messages
*.emerg                                                 *

# Save news errors of level crit and higher in a special file.
uucp,news.crit                                          /var/log/spooler

# Save boot messages also to boot.log
local7.*                                                /var/log/boot.log

#
# INN
#
news.=crit                                          /var/log/news/news.crit
news.=err                                           /var/log/news/news.err
news.notice                                         /var/log/news/news.notice
~
                                                                1,1           All
```

Each facility is associated with several different levels of logging, known as the priority. In ascending order, log priorities are: debug, info, notice, warn, err, crit, alert, emerg. The "none" priority logs all messages at all levels.

For each facility and priority, log information is sent to a specific log file. For example, take the following line from /etc/syslog.conf:

```
*.info;mail.none;news.none;authpriv.none;cron.none          /var/log/messages
```

This line sends log information from all of the given facilities to the /var/log/messages file. This includes:

- All facility messages of info level and higher
- All log messages related to mail, news, authpriv (authentication), and cron

You can use the asterisk as a wildcard in /etc/syslog.conf. For example, a line that starts with *.* tells the syslog daemon to log everything. A line that starts with auth.* means you want to log all messages from the auth facility.

By default, syslogd logs all messages of a given priority or higher. In other words, a **cron.err** line will include all log messages from the cron daemon at the *err, crit, alert,* and *emerg* levels.

Most messages from syslogd are written to files in the /var/log directory. You should scan these logs on a regular basis and look for patterns that could indicate a security breach.

Managing Logs

Logs can easily become very large and difficult to read. By default, the *logrotate* utility creates a new log file on a weekly basis. You can also configure /etc/logrotate.conf to compress, mail, and remove desired log files. By default, the cron daemon runs logrotate on a regular basis, using the configuration files located in the /etc/logrotate.d directory.

As you can see in Figure 10-7, this process works fairly well; five or more weeks of logs are kept for a number of log facilities.

The Red Hat Log Viewer

There is a new Red Hat GUI tool that can help you scan though applicable logs. It can be useful if you don't remember the locations of the key log files and don't

FIGURE 10-7

Typical log files in /var/log

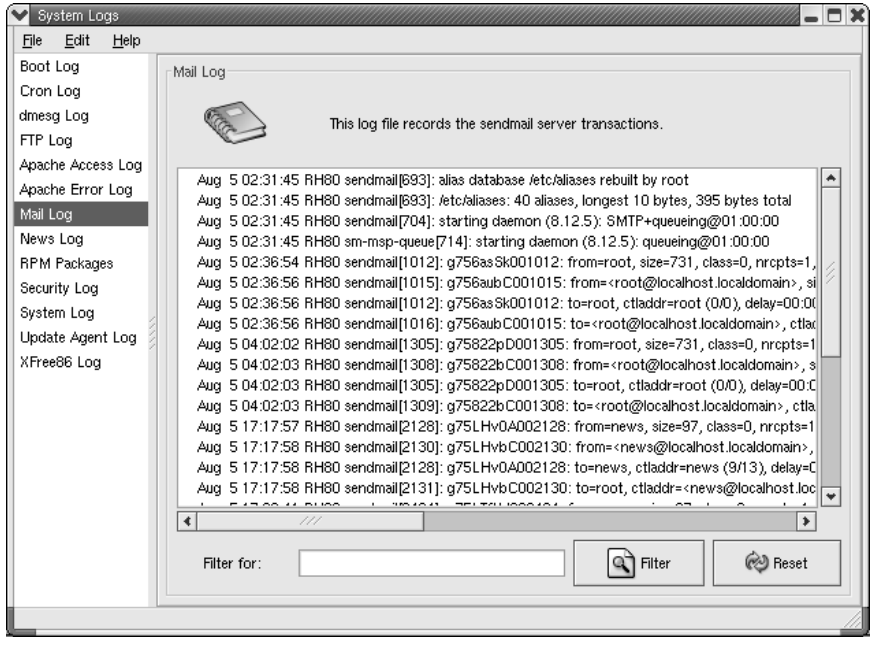
```
[mj@RH80 mj]$ ls /var/log/
boot.log     fax        maillog.1   rpmpkgs.1          spooler    wtmp
boot.log.1   gdm        maillog.2   rpmpkgs.2          spooler.1  wtmp.1
boot.log.2   httpd      maillog.3   rpmpkgs.3          spooler.2  xdm-errors
boot.log.3   ksyms.0    maillog.4   rpmpkgs.4          spooler.3  xferlog
boot.log.4   ksyms.1    messages    sa                 spooler.4  xferlog.1
cron         ksyms.2    messages.1  samba              squid      xferlog.2
cron.1       ksyms.3    messages.2  scrollkeeper.log   up2date    xferlog.3
cron.2       ksyms.4    messages.3  secure             up2date.1  xferlog.4
cron.3       ksyms.5    messages.4  secure.1           up2date.2  XFree86.0.log
cron.4       ksyms.6    news        secure.2           up2date.3
cups         lastlog    pgsql       secure.3           up2date.4
dmesg        maillog    rpmpkgs     secure.4           vbox
[mj@RH80 mj]$
```

remember to look through /etc/syslog.conf for those locations. In the Red Hat GUI, run the redhat-logviewer command to open up the tool shown in Figure 10-8.

As you can see, the Red Hat Log Viewer simply provides a front end. For example, you can review the information shown in Figure 10-8 simply by looking through the latest mail log files in the /var/log directory. And as of this writing, the regular text log files provide more complete information.

FIGURE 10-8

The Red Hat Log Viewer



CERTIFICATION OBJECTIVE 10.05

The Extended Internet Services Daemon (xinetd)

Linux typically supports network communication between clients and servers. For example, you can use Telnet to connect to a remote system. The Telnet client on your computer makes a connection with a Telnet server daemon on the remote system.

To establish the connection on a TCP/IP network, a client application needs the IP address of the server, and the *port number* associated with the server daemon. All common TCP/IP applications have a standard port number; some examples are shown in Table 10-3.

If you don't specify the port number, TCP/IP assumes that you're using the default port for the specified service. Clients can't connect unless the corresponding server is running on the remote system. If you are managing a server, you may have a number of server daemons to start when Linux is booted.

The *xinetd* (which stands for *Extended Internet Services Daemon*) program can start a number of these server daemons simultaneously. The xinetd program listens for connection requests for all of the *active* servers with scripts in the /etc/xinetd.d directory.

Each file in the /etc/xinetd.d directory specifies a particular service you want to allow xinetd to manage. By default, scripts in this directory are disabled. The following

TABLE 10-3 Typical TCP/IP Port Numbers

Port Number	Service
21	FTP
23	Telnet
25	SMTP (outgoing mail)
80	HTTP
443	HTTPS (secure HTTP)
631	Internet Printing Protocol (CUPS configuration)
901	SWAT (Samba Configuration)

code shows a sample of the /etc/xinetd.d/ntalk configuration file, with this service disabled:

```
# default: off
# description: The ntalk server accepts ntalk connections, for chatting \
#       with users on different systems.
service ntalk
{
     disable     = yes
     socket_type             = dgram
     wait                    = yes
     user                    = nobody
     group                   = tty
     server                  = /usr/sbin/in.ntalkd
}
```

This is a typical /etc/xinetd.d configuration file. The fields are described in Table 10-4. This is a versatile configuration file; other fields are described in the man pages for xinetd.conf. Read this man page; the only_from and no_access fields may be of particular interest.

exam Watch

CIDR notation is based upon "Classless Inter-Domain Routing." Under CIDR, you do not need to specify the full IPv4 subnet address; 192.168.0.0/255.255.255.0 is the same as 192.168.0.0/24. As of this writing, the RHCE exam does not require any detailed understanding of IPv6 addresses.

TABLE 10-4 Typical /etc/xinetd.d Configuration Parameters

Field	Description of Field Entry
disable	Yes by default, which disables the service
socket_type	Specifies the communication stream
wait	Yes for single-threaded applications, or No for multithreaded applications
user	Account under which the server should run
group	Group under which the server should run
server	The server program
only_from	Host name or IP address allowed to use the server. CIDR notation (e.g. 192.168.0.0/24) is okay
no_access	Host name or IP address not allowed to use the server. CIDR notation is okay

You have two ways to activate a service. You can edit the configuration file directly by changing the **disable** field from no to yes. Then make the xinetd daemon reread the configuration files with the **/sbin/service xinetd reload** command.

Alternatively, you can use the **/sbin/chkconfig** *servicename* **on** command, which automatically makes this change and makes xinetd reread the configuration file.

In some cases, it is possible to limit xinetd-based services by username. One prime example with an FTP server is the /etc/ftpaccess file, which allows you to restrict (or expand) user privileges by User ID number or username. As with other default Red Hat network configuration files, the default /etc/ftpaccess file is instructive. Try it out with your own users!

exam Watch

Always remember to make sure that a service will be active after a reboot. The */sbin/chkconfig* servicename *on* ***command is one way to do this for xinetd services. Otherwise, anything you configure may not work after your computer is rebooted.***

EXERCISE 10-2

Configuring xinetd

In this exercise, we will enable the Telnet service using xinetd. Attempt to establish a Telnet session using the command **telnet localhost**. Telnet is disabled by default in Red Hat Linux, so your attempt should fail, unless you have already enabled Telnet.

1. Edit /etc/xinetd.d/telnet and change the value of **disable** from yes to no.
2. Tell xinetd to reread its configuration file using the command:

```
kill -SIGUSR1 'cat /var/run/xinetd.pid'
```

3. Try the **telnet localhost** command again. It should work.
4. Use the /sbin/chkconfig command to disable Telnet. Do you have to restart or reload xinetd? What happens when you use /sbin/chkconfig to enable Telnet? Does it change the /etc/xinted.d/telnet configuration file?

tcp_wrappers and the libwrap Packages

The best way to prevent a cracker from using a service is to remove it completely from your Linux system. But what if you still need some Extended Internet Services (xinetd) packages?

You can achieve some measure of security by disabling or removing unused services in /etc/xinetd.conf. But you need to take other measures to protect yourself against attacks through enabled services. With xinetd, you have two approaches. You can set up fields in individual /etc/xinetd.d configuration files to block computers by host name or IP address. Alternatively, you can set this up for some or all xinetd services through the /etc/hosts.allow or /etc/hosts.deny file. This system is known as tcp_wrappers, which is enabled by default.

When xinetd receives a network request for a service, it passes the request on to tcp_wrappers. This system logs the request and then checks its access rules. If there are no limits on the particular host or IP address, tcp_wrappers passes control back to xinetd to start the needed service.

The key files are /etc/hosts.allow and /etc/hosts.deny. The philosophy is fairly straightforward; clients listed in *hosts.allow* are allowed access; clients listed in *hosts.deny* are denied access. When xinetd receives a request, the tcp_wrappers system takes the following steps:

1. It searches /etc/hosts.allow. If tcp_wrappers finds a match, it grants access.
2. It searches /etc/hosts.deny. If tcp_wrappers finds a match, it denies access.
3. If the host isn't found in either file, access is automatically granted to the client.

You use the same access control language in both /etc/hosts.allow and /etc/hosts.deny to tell tcp_wrappers which clients to allow or deny. The basic format of the lines in both files is this:

```
daemon_list : client_list
```

The simplest version of this format is:

```
ALL : ALL
```

This specifies all services managed by xinetd and makes the rule applicable to all hosts on all IP addresses. If you set this line in /etc/hosts.deny, all access is prohibited to all services. However, you can create finer filters. For example, the following line:

```
in.telnetd : 192.168.1.5
```

in /etc/hosts.allow allows the client with an IP address of 192.168.1.5 to connect to your system through Telnet. The same line in /etc/hosts.deny would prevent the computer with that IP address from using Telnet to connect to your system. You can specify clients a number of different ways, as shown in Table 10-5.

As you can see in Table 10-5, there are two different types of wildcards: *ALL* can be used to represent any client or service. The dot specifies all hosts with the specified domain name or IP network address.

You can set up multiple services and addresses with commas. Exceptions are easy to make with the *EXCEPT* operator. See the following excerpt from a /etc/hosts.allow file for an example:

```
#hosts.allow
ALL :.asafe.dom.com
in.ftpd : 192.168.25.0/255.255.255.0 EXCEPT 192.168.25.73
in.fingerd, in.rshd : 192.168.1.10
```

The first line in this file is simply a comment. The next line opens ALL xinetd services to all computers in the .asafe.dom.com domain. The following line opens FTP to any computer on the 192.168.25.0 network, except the one with an IP address of 192.168.25.73. Then, the finger and Remote Shell (rsh) services are opened to the computer with an IP address of 192.168.1.10.

The code that follows contains a hosts.deny file to see how lists can be built to control access.

```
#hosts.deny
ALL EXCEPT in.fingerd : .xyz.com
in.telnetd : ALL EXCEPT 192.168.1.10
ALL:ALL
```

TABLE 10-5 Address Fields in /etc/hosts.allow or /etc/hosts.deny

Client	Description
.example.com	Domain name. Since this domain name begins with a dot, it specifies all clients on the example.com domain.
172.16.	IP address. Since this address ends with a dot, it specifies all clients with an IP address of 172.16.*x.y*.
172.16.72.0/255.255.254.0	IP network address with subnet mask. CIDR notation not recognized.
ALL	Any client, any daemon.
user@linux1.example.com	Applies to the specific user on the given computer.

The first line in the hosts.deny file is a comment. The second line denies all services except finger to computers in the .xyz.com domain. The third line states that the only computer that is allowed to telnet to us has an IP address of 192.168.1.10. Finally, the last line is a blanket denial; all other computers are denied access to all services controlled by tcp_wrappers.

You can also use the **twist** command in /etc/hosts.allow or /etc/hosts.deny to access shell commands. For example, take the following line in a /etc/hosts.deny file:

```
in.telnetd : .crack.org : twist /bin/echo Sorry %c, access denied
```

This sends a customized error message for Telnet users on the crack.org domain. Different operators such as **%c** are described in Table 10-6. Some of these operators may be able to help you track the intruder.

EXERCISE 10-3

Configuring tcp_wrappers

In this exercise, we will use tcp_wrappers to control access to network resources. Since tcp_wrappers is enabled by default, you shouldn't have to make any modifications to /etc/xinetd.conf.

1. Verify that you can telnet to the system using the address localhost.
2. Edit /etc/hosts.deny and add the following line (don't forget to write the file):

   ```
   ALL : ALL
   ```

3. What happens when you try to telnet to the address localhost?
4. Edit /etc/hosts.allow and add the following line:

   ```
   in.telnetd : LOCAL
   ```

5. Now what happens when you try to telnet to the address localhost?
6. If you have other systems available to you, try restricting access to the Telnet service using some of the other tcp_wrappers rules.
7. Undo your changes when finished.

TABLE 10-6 tcp_wrappers Operators

Field	Description	Field	Description
%a	Client address	%h	Client host name
%A	Host address	%H	Server host name
%c	Client information	%p	Process ID
%d	Process name	%s	Server information

CERTIFICATION OBJECTIVE 10.06

Firewall Policies

A firewall sits between your company's internal LAN and an outside network. A firewall can be configured to examine every network packet that passes into or out of your LAN. When configured with appropriate rules, it can filter out those packets that may pose a security risk to your system. To understand how *packet filtering* works, you have to understand a little bit about how information is sent across networks.

Before you send a message over a network, the message is broken down into smaller-sized units called *packets*. Administrative information, including the type of data, the source address, and destination address, is added to each packet. The packets are reassembled when they reach the destination computer. A firewall examines these administrative fields in each packet to determine whether to allow the packet to pass.

Red Hat Linux comes with everything you need to configure a system to be a firewall. Three basic Linux firewall commands are available: **ipfwadm**, **ipchains**, and **iptables**. The first command, ipfwadm, was associated with Linux kernel 2.0.x and is now generally obsolete. The ipchains command was developed for Linux kernel 2.2.x and is still in active use, even on Linux distributions based on Linux kernel 2.4.x.

The RHCE exam explicitly requires that you know how to use iptables, which was developed for Linux kernel 2.4.x. Therefore, this chapter focuses on iptables.

Configuring iptables

The philosophy behind iptables is based on "chains." These are sets of rules applied to each network packet. Each rule does two things: it specifies the conditions a packet must meet to match the rule, and it specifies the action if the packet matches.

Before you can set up iptables commands, you need to make sure that the appropriate modules are part of your Linux kernel. Check your current rules. Run the **/sbin/iptables -L** command to list the current chains. If you see error messages similar to the following:

```
iptables: Incompatible with this kernel
```

you'll need to upgrade your modules. Use the **/sbin/rmmod** ***modulename*** command to delete any ipchains-related modules. Then use the **/sbin/insmod ip_tables** command to add the iptables kernel module. Now you're ready to start configuring iptables rules.

The iptables command uses the following basic format:

```
iptables -t tabletype <Action / Direction> <Packet Pattern> -j <What to do>
```

Now let us analyze this command, step by step. First there is the **-t** ***tabletype*** switch. There are two basic *tabletype* options for iptables:

- **filter** This sets a rule for filtering packets.
- **nat** This sets up Network Address Translation, which is discussed in the last section of this chapter.

The default is **filter**; if you don't specify a **-t** ***tabletype,*** iptables assumes that you're trying to affect a filtering rule. Next is the <Action / Direction>. There are four basic Actions that you can take with an iptables rule:

- **-A (--append)** Appends a rule to the end of a chain.
- **-D (--delete)** Deletes a rule from a chain. Specify the rule by the number or the Packet Pattern.
- **-L (--list)** Lists the currently configured rules in the chain.
- **-F (--flush)** Flushes all of the rules in the current iptables chain.

If you're appending to (-A) or deleting from (-D) a chain, you'll want to apply it to network data traveling in one of three directions:

- **INPUT** All incoming packets are checked against the rules in this chain.

- **OUTPUT** All outgoing packets are checked against the rules in this chain.
- **FORWARD** All packets being sent to another computer are checked against the rules in this chain.

Next, you need to configure a <Packet Pattern>. Your firewall checks every packet against this pattern. The simplest pattern is by IP address:

- **-s *ip_address*** All packets are checked for a specific source IP address.
- **-d *ip_address*** All packets are checked for a specific destination IP address.

Packet Patterns can be more complex. In TCP/IP, packets are transported using the TCP, UDP, or ICMP protocol. You can specify the protocol with the **-p** switch, followed by the destination port (**--dport**). For example, the **-p tcp --dport 80** parameters prevent users outside your network from looking for an HTTP connection.

Once iptables matches a Packet Pattern, it needs to know what to do with that packet, which leads to the last part of the command, **-j <what to do>**. There are three basic options:

- **DROP** The packet is dropped. No message is sent to the requesting computer.
- **REJECT** The packet is dropped. An error message is sent to the requesting computer.
- **ACCEPT** The packet is allowed to proceed as specified with the -A action: INPUT, OUTPUT, or FORWARD.

We will look at some examples of how you can use iptables commands to configure a firewall. The first step is always to see what is currently configured, with the following command:

```
/sbin/iptables -L
```

If iptables is properly configured, it should return chain rules in three different categories: INPUT, FORWARD, and OUTPUT.

Let's look at some examples. The following command defines a rule that rejects all traffic from the 192.168.75.0 subnet, and it sends a "destination unreachable" error message back to any client that tried to connect:

```
/sbin/iptables -A INPUT -s 192.168.75.0/24 -j REJECT
```

This rule stops users from the computer with an IP address of 192.168.25.200 from "pinging" our system (remember that ping uses the ICMP protocol):

```
/sbin/iptables -A INPUT -s 192.168.25.200 -p icmp -j DROP
```

The following command guards against TCP SYN attacks from outside our network. Assume that our network IP address is 192.168.1.0. The exclamation point (!) means "everything but":

```
/sbin/iptables -A INPUT -s !192.168.190.0/24 -p tcp -j DROP
```

Then, if you wanted to delete the rule related to the ping command in this list, use the following command:

```
/sbin/iptables -D INPUT -s 192.168.25.200 -p icmp -j DROP
```

The default rule for INPUT, OUTPUT, and FORWARD is to ACCEPT all packets. One way to stop packet forwarding is to add the following rule:

```
/sbin/iptables -A FORWARD -j DROP
```

You can save your firewall configuration to a file using the following command:

```
/sbin/service iptables save
```

This saves your chains in the /etc/sysconfig/iptables configuration file. The iptables service script then reads this file, if it is active for the appropriate runlevel when you start Linux. You can configure iptables so that it is active for all network runlevels (2, 3, 4, and 5) with the chkconfig command, as follows:

```
# /sbin/chkconfig --level 2345 iptables on
# /sbin/chkconfig --list iptables
ipchains        0:off   1:off   2:on    3:on    4:on    5:on    6:off
```

exam Watch

Knowing how to secure a Red Hat Linux system against unauthorized access is critical. Be sure you understand the concepts and commands discussed in this chapter.

CERTIFICATION OBJECTIVE 10.07

Network Address Translation

Network Address Translation (NAT) lets you hide the IP address of the computers on your network that make a connection to the Internet. NAT replaces the source address with the IP address of the firewall computer. That address information is cached.

When the firewall receives data such as a Web page, the process is reversed. As the packets pass through the firewall, the originating computer is identified in the cache. The header of each packet is modified accordingly before the packets are sent on their way.

This approach is useful for several reasons. Disguising your internal IP addresses makes it harder for someone to break into your network. NAT allows you to connect computers to the Internet without having to have an official IP address for each computer. This allows you to use the private IP addresses discussed in Chapter 1 on your internal LAN. In the Linux world, this process is known as IP masquerading.

IP Masquerading

Red Hat Linux supports a variation of NAT called *IP masquerading*. IP masquerading allows you to provide Internet access to multiple computers with a single officially assigned IP address. IP masquerading lets you map multiple internal IP addresses to a single valid external IP address.

Connecting multiple systems to the Internet using IP masquerading is a fairly straightforward process. Your firewall computer will need one network card to connect to your LAN, and a second network card for the Internet. This second network card can be a telephone modem, or it can be connected to a cable "modem" or DSL adapter. This configuration requires the following steps:

- Assign your official IP address to the network card that is directly connected to the Internet.
- Assign computers on your LAN one of the private IP addresses described in Chapter 1.
- Reserve one private IP address for the network card on your firewall that is connected to the LAN.

- Use iptables to set up IP masquerading.
- Enable IP forwarding on the firewall computer.
- Configure the computers on your LAN with the IP address of your firewall computer as their Internet gateway.

Let us take a careful look at when a message comes from a computer on a LAN, through a firewall, to the Internet. When a computer on your LAN wants a Web page on the Internet, it sends packets to the firewall. The firewall replaces the source IP address on each packet with the firewall's official IP address. It then assigns a new port number to the packet. The firewall caches the original source IP address and port number.

When a packet comes in from the Internet to the firewall, it should include a port number. If your firewall can match it with the port number assigned to a specific outgoing packet, the process is reversed. The firewall replaces the destination IP address and port number with the internal computer's private IP address and then forwards the packet back to original client on the LAN.

The next step in the process is to use iptables to enable masquerading. The following command assumes that eth1 represents the network card that is directly connected to the Internet, and that your LAN has a network address of 192.168.0.0/24:

```
/sbin/iptables -t nat -A POSTROUTING -s 192.168.0.0/24 -o eth1 -j MASQUERADE
```

The following command enables FTP access through your firewall:

```
/sbin/modprobe -a ip_conntrack_ftp ip_nat_ftp
```

Similar modules are available in your kernel directory, under /usr/src/linux-2-4/net/ipv4/netfilter. But there is one more thing. IP masquerading does not work unless you've enabled IP forwarding, as described in the next section.

IP Forwarding

IP forwarding is more commonly referred to as *routing*. Routing is critical to the operation of the Internet or any IP network. Routers connect and facilitate communication between multiple networks. When you set up a computer to find a site on an outside network, you need a gateway address. This corresponds to the IP address of your router on your LAN.

A router looks at the destination IP address of each packet. If the IP address is on one of its LANs, it routes the packet directly to the proper computer. Otherwise, it sends the packet to another gateway closer to its final destination. To use a Red Hat Linux system as a router, you must enable IP forwarding in the /etc/sysctl.conf configuration file by changing:

```
net.ipv4.ip_forward = 0
```

to

```
net.ipv4.ip_forward = 1
```

These settings take effect the next time you reboot your system. Until you reboot, you can enable forwarding directly in your kernel with the following command:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Now that you have seen some of the security capabilities of Red Hat Linux, refer to the following Scenario & Solution for some possible scenario questions and their answers.

SCENARIO & SOLUTION

You have installed an FTP server on your corporate network, and you want to restrict access to certain departments. Each department has its own subnet.	Use the /etc/hosts.deny file in the tcp_wrappers package to block FTP access (in.ftpd) to the unwanted subnets.
You have only one official IP address, but you need to provide Internet access to all of the systems on your LAN. Each computer on the LAN has its own private IP address.	Use iptables to implement IP masquerading. Make sure IP forwarding is active.
You have a LAN of Linux and Unix computers, and want to implement a single authentication database of usernames and passwords for the network.	Implement NFS file sharing on the network. Set up an NIS server. Set up the other computers on your LAN as NIS clients.
You want to modify the commands associated with halting and rebooting your computer so they're accessible only to the root user.	Set up the appropriate Pluggable Authentication Module configuration files in /etc/pam.d to use the system-auth module.

The Red Hat Firewall Configurator

You can automate the process of configuring a firewall. Run the redhat-config-securitylevel command. This brings up the Red Hat Firewall Configurator, as shown in Figure 10-9. If you've installed Red Hat Linux before, this menu should look familiar; the choices are identical to those shown during the standard Red Hat Linux installation process.

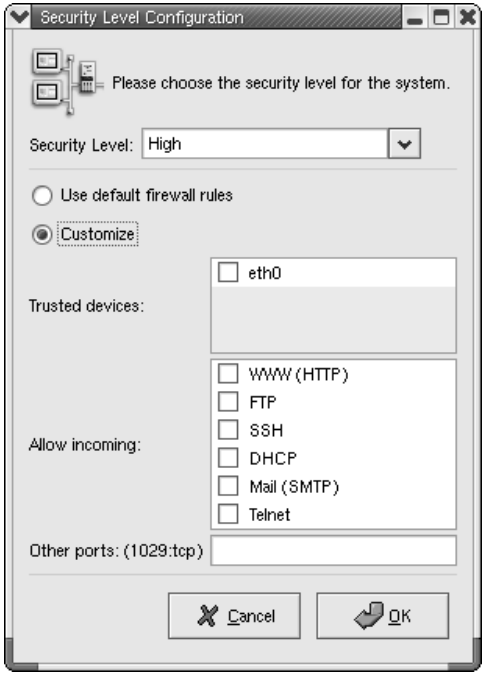
The choices here are similar to what you may find with the /usr/sbin/lokkit tool. Both tools are front ends to creating the appropriate iptables commands to control traffic coming in, going out, and moving through your computer.

Three basic security levels are available:

- High security blocks all inbound request traffic except replies from DNS servers. In other words, you can still send messages out; DNS replies allow you to browse the Internet. However, other computers won't be able to connect to servers on your computer.

FIGURE 10-9

The Red Hat Firewall Configurator



- Medium security blocks requests to many servers on your computer. Specifically, it blocks traffic to TCP/IP ports below 1023, as well as the NFS server, the X Window display, and the X Font Server. It allows you to use special services on external networks such as RealAudio.
- No security disables any rules that you've previously created using the Red Hat Firewall Configurator. It does not delete any rules that you've created directly with the iptables command.

For medium or high security, you can create exceptions to each rule, by selecting the Customize radio button. Firewalls are not applied to "Trusted Devices." If you allow incoming traffic, others can access the associated server on your computer. For example, if you select Allow Incoming WWW (HTTP), others can connect to a Web server on your computer.

The settings that you create are documented in /etc/sysconfig/iptables. But there may be more firewall rules. You may have added some firewall chains with an iptables command. You still need to apply the **/sbin/iptables -L** command to list current iptables firewall chains.

CERTIFICATION SUMMARY

One of the basic functions of a Red Hat Linux system administrator is protecting a Linux computer and a network from inside and outside attacks. Red Hat Linux includes a variety of tools that can help you establish a secure computing environment.

Red Hat Linux can be a powerful tool for securing networks from outside attack. You can use centralized account management with an NIS service. Log files can be configured to collect data from any number of services. Pluggable Authentication Modules can help you configure how individual services verify usernames and passwords. The Extended Internet Services daemon, xinetd, governs the services configured though the /etc/xinetd.d directory.

With tcp_wrappers and iptables at your disposal, you can create a firewall which can protect your Red Hat Linux system and LAN. Firewalls require a computer with at least two network cards. Routing must be enabled on that computer. The firewall can include IP masquerading to hide the IP addresses of the computers inside your LAN.

TWO-MINUTE DRILL

The following are some of the key points from the certification objectives in Chapter 10.

Configuring NIS Client

- ❑ NIS allows you to configure one centrally managed username and password database with other Linux and Unix systems on your LAN.
- ❑ With NIS, you maintain one password database on an *NIS server* and configure the other systems on the network to be *NIS clients.*
- ❑ You can configure NIS to share other configuration files, including many of those in the /etc directory.
- ❑ To configure NIS, you need to configure at least one computer as an NIS server.
- ❑ The NIS server stores the centralized NIS database files, which are also known as *maps.*
- ❑ You can have only one NIS master server per NIS domain.
- ❑ The NIS maps are stored in the /var/yp/*DOMAIN,* where *DOMAIN* is the name of your NIS domain.

Basic Host Security

- ❑ Password security requires good passwords from your users.
- ❑ You can check for suspicious login activity with the **utmpdump /var/log/wtmp** command.
- ❑ Many security updates are available through Red Hat errata releases.
- ❑ You can update many packages with up2date.
- ❑ The best way to promote security is to delete the packages associated with services that you do not need.

The Pluggable Authentication Module (PAM) System

- ❑ Red Hat Linux uses the Pluggable Authentication Modules (PAM) system to check for authorized users.

- PAM modules are called by configuration files in the /etc/pam.d directory. These configuration files are usually named after the service or command that they control.
- There are four types of PAM modules: authentication, account, password, and session management.
- PAM configuration files include lines that list the module_type, the control_flag, the path to the actual module, followed by arguments such as system-auth.
- PAM modules are well documented in the /usr/share/doc/pam-*versionnumber*/txts directory.

System Logging

- Red Hat Linux includes two logging daemons: klogd for kernel messages and syslogd for all other process activity. Both are activated by the syslog service script.
- You can use log files generated by the syslogd daemon to track activities on your system.
- Most log files are stored in /var/log.
- You can configure what is logged through the syslog configuration file, */etc/syslog.conf*.

The Extended Internet Services Daemon (xinetd)

- xinetd is the Extended Internet Services Daemon, which acts as a "super-server" for a number of other network services, such as IMAP, POP, FTP, rsh, and Telnet.
- Individual services have their own management scripts in the /etc/xinetd.d directory.
- Most xinetd services are disabled by default.
- You can activate an xinetd service with the appropriate chkconfig command, or by directly editing its xinetd script.
- xinetd listens for connection requests from client applications.

- ❑ When xinetd receives a connection request, it starts the server associated with the TCP/IP port, then waits for other connection requests.
- ❑ Red Hat Linux comes with a package known as libwrap or tcp_wrappers. This package, which is enabled by default, allows you to limit access to various xinetd services.
- ❑ You configure the access rules for tcp_wrappers through the /etc/hosts.allow and /etc/hosts.deny configuration files.
- ❑ Clients listed in /etc/hosts.allow are allowed access; clients listed in /etc/hosts.deny are denied access.
- ❑ Services can also be configured in /etc/hosts.allow and /etc/hosts.deny. Remember to use the actual name of the daemon, such as in.telnetd.

Firewall Policies

- ❑ Firewalls can secure an internal network as a *packet filter* that controls the information that comes in, goes out, and is forwarded through the internal network.
- ❑ The current firewall configuration utility is iptables, which has replaced ipchains.
- ❑ The iptables utility retains a number of elements of ipchains. iptables directives are sets of rules, chained together, which are compared and then applied to each network packet.
- ❑ Each rule sets conditions required to match the rule, and then specifies the action taken if the packet matches the rule.
- ❑ Use the **/sbin/service iptables save** command to save any chains that you configure in the /etc/sysconfig/iptables configuration file.

Network Address Translation

- ❑ NAT modifies the header in packets coming from a LAN. The source address is replaced with the public address of the firewall computer, with a random port number. Both are stored in cache.
- ❑ Linux supports a variation of NAT called *IP masquerading*.

- IP masquerading allows you to provide Internet access to multiple computers with a single officially assigned IP address.
- With IP masquerading, messages for the network are sorted by the port number. The original source address is taken from the cache and added to the packet, so the message gets to the right computer.
- A firewall computer needs at least two network cards: one on the LAN, and the other on an external network such as the Internet.
- IP forwarding is more commonly known as *routing*.
- Routing is critical to the operation of the Internet or any IP network.
- A router checks the destination IP address of each packet. If the address is on a connected LAN, it sends the packet directly to that computer. If the address is outside the LAN, it sends the packet to another router closer to its final destination.
- To enable IP forwarding, edit /etc/sysctl.conf and change the line to net.ipv4.ip_forward to 1.
- To enable IP forwarding immediately, type the **echo 1 > /proc/sys/net/ipv4/ip_forward** command.

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.

Configuring NIS Clients

1. You have a network with 50 Linux workstations and five Linux servers. Most of the workstations are in public areas, and your users need to be able to log in from any workstation on the network. How might you satisfy this requirement?
   - A. Keep a master copy of /etc/passwd on one of the servers, and do a backup and restore of that copy to all the workstations every evening.
   - B. Set one of the servers up to be an NIS server. Arrange another server to be an NIS slave server. Make the workstations NIS clients.
   - C. Set the workstations up to be NIS clients.
   - D. Create a common account on every workstation and give each person the password to this account.
2. How would you set up the workstations to be NIS clients?
   - A. Edit /etc/passwd and add the line USE_NIS at the end of the file.
   - B. Start the ypbind daemon.
   - C. Add a line to start ypbind to /etc/xinetd.conf.
   - D. Run authconfig and enable NIS.
3. A user on one of the NIS workstations calls you and tells you she is having trouble changing her password using the passwd command. What should you tell her?
   - A. You'll change her password for her.
   - B. Try picking a more secure password.
   - C. Make sure the caps lock key isn't on.
   - D. She must use yppasswd to change her NIS password.

Basic Host Security

4. Which of the following are *not* good basic host security measures?
   A. Jotting down the root password on your desk blotter.
   B. Checking system log files regularly for unusual activity.
   C. Hanging on to unused accounts in case their original users want to reactivate them.
   D. Providing users with adequate training so they know how to properly use the tools at their disposal.
5. Which of the following measures is the most effective way to prevent attacks through various network services?
   A. Disable a service in the appropriate /etc/xinetd.d configuration file.
   B. Block service requests with the appropriate commands in /etc/hosts.deny.
   C. Use a firewall to drop all requests to unneeded services.
   D. Uninstall unneeded network services.

The Pluggable Authentication Module (PAM) System

6. What are the four steps that PAM breaks the authentication process into?
   A. Authentication management, account management, session management, and password management
   B. Authentication management, account management, network management, and password management
   C. Authentication management, account logging, session management, and password management
   D. Authentication management, account management, session management, and firewall management
7. You are editing the PAM configuration file by adding a module. How would you indicate the authentication process should immediately terminate and succeed if the module succeeds?
   A. Make sure the module is either an auth module or a password module, since these must always succeed.
   B. Use the required control flag.
   C. Use the sufficient control flag.
   D. It doesn't matter; the authentication process always stops as soon as a module fails.

8. You experience a moment of forgetfulness and try to log in to the root account of your server via Telnet from your Internet connection at home. Why doesn't this work?
   A. You are using iptables to filter out Telnet access to the root account.
   B. You miskeyed your password.
   C. Login to the root account is never allowed from any terminal other than the console.
   D. The network terminal device you are trying to log in from is not listed in /etc/securettys; therefore, the root account will not be allowed to log in from that terminal.

System Logging

9. Assume you normally work from a user account called sysadm. How might you configure your Red Hat Linux System to notify you whenever there is a serious problem with the kernel?
   A. Edit /etc/syslog.conf and add an entry such as this:
   ```
   kern.err        root,sysadm
   ```
   B. Recompile the kernel to include error notification and specify sysadm as the user to be notified.
   C. Write a C program to monitor the /proc/err directory and send any messages that appear there to sysadm.
   D. Edit /etc/syslog.conf and add an entry such as this:
   ```
   *.*        root,sysadm
   ```

The Extended Internet Services Daemon (xinetd)

10. You would like to restrict access to your FTP site to clients in a particular subnet. How can you do this?
    A. Use iptables to filter out FTP requests for all but the given subnet.
    B. Disable the configuration for wu-ftpd in the /etc/xinetd.d directory.
    C. Edit /etc/ftp.conf and add a reject line for all networks other than the given subnet.
    D. Add the appropriate lines to /etc/hosts.allow and /etc/hosts.deny.

11. You are using the xinetd program to start services. How could you restrict Telnet access to clients on the 192.168.170.0 network?
    A. Edit /etc/xinetd.d/telnet and add this line
    ```
    DENY EXCEPT 192.168.170.0.
    ```

B. Edit /etc/hosts.allow and add this line:

```
in.telnetd : 192.168.170.0/255.255.255.0
```

C. Edit /etc/hosts.deny and add this line:

```
in.telnetd : 192.168.170.0/255.255.255.0
```

D. Edit /etc/hosts.deny and add this line:

```
in.telnetd : ALL EXCEPT 192.168.170.0/255.255.255.0
```

12. You work at a company with several divisions. Each division is part of the headquarters LAN, but each division has its own logical subnet and its own domain. You would like to set up an internal FTP server for just one division, and you do not want to allow access to users in other divisions. What configuration would work?

A. Set up a user's workstation in each division to be the FTP server and delegate the management of that server to the user of that workstation.

B. Use NIS and set up shared virtual FTP directories.

C. Edit the /etc/hosts.deny file to specify the in.ftpd service. Deny access to all, except for the one division.

D. Edit /etc/xinetd.d/wu-ftpd and set enable = yes.

13. Based on the configuration described in question 10, what would you do to restrict access from a specific user mj with a user ID of 500?

A. Edit the /etc/xinetd.d/wu-ftpd configuration file, and add **nouser=mj**.

B. Edit the /etc/hosts.deny file and add **in.ftpd : user=mj**.

C. Edit the /etc/ftphosts file and add the user ID for mj to the list.

D. Edit the /etc/ftpaccess file and add **deny uid %500**.

Firewall Policies

14. You have just recently connected your organization's network to the Internet, and you are a little worried because there is nothing other than your router standing between your network and the Internet. You have a spare 200 MHz PC lying around that just happens to have two Ethernet cards. You also have a mixture of systems on your network that includes Macintosh, Windows 98, and Linux. What might you do to ease your mind?

A. Nothing, you're not advertising the systems on your LAN via DNS, so no one will ever find them.

B. Install Red Hat Linux on the 200 MHz PC and use iptables to set it up as a firewall.

C. Install Red Hat Linux on the 200 MHz PC and use tcp_wrappers to set it up as a firewall.

D. Install Linux on all systems on your network.

15. Consider the following command:

```
/sbin/iptables -A INPUT -s 192.168.77.77 -j REJECT
```

What effect will this have when the client with an IP of 192.168.77.77 tries to connect to your system?

A. No effect at all.

B. Access will be denied, and the client computer won't get any message on what happened.

C. Access will be denied, and the client application will get a message that the target destination is unreachable.

D. You will receive a notification message on the system console.

Network Address Translation

16. You are setting up a small office and would like to provide Internet access to a small number of users, but you don't want to pay for a dedicated IP address for each system on the network. How could Linux help with the problem?

A. Assign the official IP address to a Linux system and create accounts on that system for all of the office personnel.

B. Install Linux and configure it for IP forwarding.

C. Install a Linux router.

D. Use the Linux system to connect to the Internet; then use iptables to set up IP masquerading.

LAB QUESTION

Part I

You want to set up a secure Web server on your corporate LAN that supports inbound requests from your LAN and the Internet, but you do not want any of these requests from the Internet to get into your intranet. What can you do?

Part 2

You want to set up Telnet service on your internal LAN, accessible only to one specific IP address. You want to block access from outside the LAN. Assume that your LAN's network address is 192.168.1.0, and the IP address of the computer that should get access is 192.168.1.33. For the purpose of this lab, feel free to substitute the IP address of a second Linux computer on your network. What do you do?

SELF TEST ANSWERS

Configuring NIS Clients

1. ☑ **B**. This is an ideal situation for NIS.
   ☒ **A** is incorrect because it is labor intensive and would lead to many password and database inconsistencies. **C** is incorrect because you need at least one NIS server. **D** is incorrect because this is obviously an insecure way to run a network.

2. ☑ **D**. Although you can configure NIS clients manually, the easier way is to use the authconfig utility.
   ☒ **A** is incorrect because this is invalid syntax. **B** is incorrect because you need to do more than start ypbind. **C** is incorrect because ypbind should be started from /etc/rc.d/init.d.

3. ☑ **D**. She must use the NIS yppasswd command to change her NIS password.
   ☒ **A**, **B**, and **C** are all incorrect because the user's account is an NIS account; therefore, the only valid choice is **D**.

Basic Host Security

4. ☑ **A** and **C** are both not recommended.
   ☒ **B** and **D**, on the other hand, are both good security practices.

5. ☑ **D**. The most effective way to prevent an attack through a network service is to make sure that it is not installed.
   ☒ **A**, **B**, and **C** are all incorrect. Since the service is still installed on the system, it is still at least theoretically possible to attack through that service.

The Pluggable Authentication Module (PAM) System

6. ☑ **A**. PAM breaks the authentication process into these four steps.
   ☒ **B**, **C**, and **D** are not the four steps that PAM breaks the authentication process into.

7. ☑ **C**. The *sufficient* flag is used to indicate the authentication process should end immediately if the module succeeds.
   ☒ **A** is incorrect because any PAM module can fail and the authorization process will continue. **B** is incorrect because failure would be delayed until any other modules of the same type have been checked. **D** is incorrect because the control flag determines when the authorization process terminates.

8. ☑ **D**. The root account is allowed to log in only from terminals listed in /etc/securettys.
☒ **A** is incorrect because this is not a typical firewall function. **B** is obviously incorrect. **C** is incorrect because answer **D** explains how access can be granted.

System Logging

9. ☑ **A**. Although **D** might seem like a good choice, this would also show you all messages from every facility. It would be very difficult to pick out just the kernel messages from everything else that would be coming to your screen.
☒ **B** and **C** are obviously incorrect because there is too much effort involved.

The Extended Internet Services Daemon (xinetd)

10. ☑ **D**. This is a good situation for tcp_wrappers, and its /etc/hosts.allow and /etc/hosts.deny configuration files.
☒ **A** is incorrect because iptables is better used to filter entire protocols. **B** is incorrect because this would disable FTP completely. **C** is incorrect because there is no ftp.conf file.

11. ☑ **D**. Although **B** would allow the requested access, since no other configuration has been done for tcp_wrappers, /etc/hosts.deny will be empty, so other clients will be allowed access by default. The best choice is to restrict all access to the telnet daemon and then make an exception for clients in the requested subnet.
☒ **A** is incorrect because the syntax is wrong. **C** is incorrect because it would result in Telnet access being denied to the 192.168.170.0 network.

12. ☑ **C**. The in.ftpd service is the actual name of the ftp daemon. It is easy to block access to other divisions (and the rest of the Internet) through the /etc/hosts.deny file.
☒ **A**, **B**, and **D** are all incorrect. It's a poor idea to delegate any service to a nonadministrative user. NIS does not configure FTP. There is no enable flag for xinetd configuration files.

13. ☑ **D**. The /etc/ftpaccess file allows you to block users by username or user ID.
☒ **A**, **B**, and **C** are all incorrect, as they are not viable options for the specified configuration files.

Firewall Policies

14. ☑ **B**. Your best choice would be to take the unused PC and turn it into a firewall using Linux and iptables. If you use a router to connect to the Internet, then your firewall system sits between your LAN and the router. This results in a two-node network consisting of the router and one of the network interfaces in your firewall that serves as a buffer zone between the

Internet and your LAN. You assume that any traffic on this side of the firewall is potentially unsafe.

☒ **A** is incorrect because this is a poor way to secure a network. **C** is incorrect because although you might also want to use tcp_wrappers as part of your security strategy, it is designed to secure individual computers, not an entire network. Although **D** might be a good option in general, it won't necessarily make your network more secure.

15. ☑ **C**. Because of the REJECT target, the client will receive an error message. If the target was DENY, the client would not get any error message.

☒ **A**, **B**, and **D** do not describe what happens with this firewall when the client with an IP address of 192.168.77.77 tries to connect to your system.

Network Address Translation

16. ☑ **D**. If you need to connect several systems to the Internet but have only one official IP address to use, IP masquerading is the perfect solution.

☒ **A** is incorrect unless your users want to telnet to a single system and use a command line interface. **B** and **C** are essentially the same answer and are both incorrect because a router will not help in this situation.

LAB ANSWER

Part 1

Scenario 1: Cost is not an object. This means you can build a DMZ, demilitarized zone, using two firewalls and a separate Web server, all running Linux. You should have the Web server dedicated only to the Web. You configure two more Linux hosts, each with two network cards, and essentially isolate the intranet behind one firewall. You then put the Web server in the middle, placing the second firewall between the Web server and the Internet. You configure the firewall on the intranet with IP masquerading to ensure anonymity for all your intranet hosts.

Scenario 2: You have one old computer available, and the Web server is a separate computer. Use your one computer as the firewall between you and the Internet and only forward HTTP packets to the Web server IP address directly; use NAT for all intranet requests going out to the Internet for HTTP and FTP. Disallow all other services.

Part 2

When you set up any xinetd service such as Telnet, there are several steps in the process. You'll need to modify the xinetd Telnet configuration file, and set up filtering in one of three ways: in the /etc/xinetd.d/telnet configuration file, through tcp_wrappers, or the appropriate firewall commands:

1. First, you want to enable Telnet. Make sure that the Telnet RPM is installed.
2. Activate Telnet. Use the **/sbin/chkconfig telnet on** command to revise the /etc/xinetd.d/telnet configuration script.
3. Edit the /etc/xinetd.d/telnet configuration file. Add the **only_from = 192.168.1.33** line.
4. Save the configuration file and reload the xinetd service script with the **/sbin/service xinetd reload** command. Try accessing Telnet from the local computer. What happens?
5. Try accessing Telnet from the computer with the IP address of 192.168.1.33. What happens? Try again from a different computer on your LAN.
6. Restore the previous /etc/xinetd.d/telnet configuration file. Don't forget to reload the xinetd service script with the **/sbin/service xinetd reload** command.
7. Edit /etc/hosts.deny. Add the **in.telnetd : ALL EXCEPT 192.168.1.33** line.
8. Try accessing Telnet from the computer with the IP address of 192.168.1.33. What happens? Try again from a different computer on your LAN.
9. Restore the previous /etc/hosts.deny file.
10. Save any iptables chains. Back up /etc/sysconfig/iptables, if that file currently exists to ~/bak.iptables.
11. Flush current firewall rules with the **/sbin/iptables -F** command.
12. Block the Telnet port, 23, for all IP addresses except 192.168.1.33 with the **/sbin/iptables -A INPUT -s ! 192.168.1.33 -p tcp --dport 23 -j DROP** command.
13. Try accessing the Telnet server from the computer with the IP address of 192.168.1.33. What happens? Try again from a different computer on your LAN.
14. Flush current firewall rules with the **/sbin/iptables -F** command.
15. Restore any previous firewall rules with the **/sbin/iptables-restore < ~/bak.iptables** command.
16. EXTRA CREDIT: Repeat these commands for other services and networks.