



10

Systems Administration and Security

CERTIFICATION OBJECTIVES

- 10.01 Configuring Directory Clients and NIS Servers
- 10.02 Basic Host Security
- 10.03 The Pluggable Authentication Module (PAM) System
- 10.04 System Logging Configuration

- 10.05 The Extended Internet Services Daemon (xinetd)
- 10.06 Firewall Policies
- 10.07 Network Address Translation
- ✓ Two-Minute Drill

Q&A Self Test

As a Red Hat Enterprise 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. 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 Enterprise Linux workstation at home does not require as much security as a secure Red Hat Enterprise Linux server that is being used to process credit card orders for a Web site.

Red Hat Enterprise 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 provided by RHEL 3 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.



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) and the Lightweight Directory Access Protocol (LDAP) 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 (xinetd) 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.

INSIDE THE EXAM

Systems Administration and Security

RHCT candidates need to know how to configure a client to directory services such as NIS and LDAP. They will also want to read the System Logging section to learn how to troubleshoot a number of basic problems. However, most of this chapter is focused on RHCE requirements. While not part of the Red Hat Exam Prep guide, NIS servers are included in the RHCE prep course outline, RH300.

With respect to the RHCE exam, focus on security. Remember to remove unneeded services to minimize the ways crackers can break into your system. Learn the PAM system, and how it allows you to customize security for different services and utilities—even by user. Understand how to implement `tcp_wrappers` to secure `xinetd` services. Learn the `iptables` command in detail. Practice with this command. Experiment with the results from remote clients.

CERTIFICATION OBJECTIVE 10.01

Configuring Directory Clients and NIS Servers

By default, access to a Linux computer 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.

There are two services which allow you to set up one centrally managed database of usernames and passwords for Linux and Unix computers. These services are NIS and LDAP. With each of these services, you can maintain one password database on an NIS or LDAP server and configure the other systems on the network as clients. When a user logs into an NIS or LDAP 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 server.

I’ll start by showing you how to configure NIS and LDAP clients. First, I’ll show you how you can configure these clients using the command line interface and then use the Red Hat Authentication Configuration tool.

exam

Watch

In the Red Hat Exam Prep guide, the only requirement is to be able to connect a client to a network directory service, such as NIS or LDAP. The prep course outline for the RHCE, RH300, does include

NIS (but not LDAP) server configuration. It's possible that LDAP services will become more important in future versions of the RHCE exam. I therefore focus on NIS servers in this chapter.

NIS Client Configuration

It's fairly simple to configure an NIS client on a network. Assuming you have an NIS server, you need to do three things. First, specify the server and domain name in `/etc/yp.conf`. Next, make sure the **ypbind** client service starts the next time you boot Linux. Finally, make sure the `/etc/nsswitch.conf` file looks to the NIS service for at least the username and password database.

The change to the `/etc/yp.conf` configuration file is simple. All you need is a command such as the following, which specifies the name of the NIS domain as `nisdomain`, and the name of the NIS server as `enterprise3`:

```
domain nisdomain server enterprise3
```

Making sure that the **ypbind** client service starts the next time you boot Linux is a simple matter. Just as with other Linux services, you can make sure it starts at the appropriate runlevels with a command such as the following:

```
# chkconfig --level 35 ypbind on
```

Finally, making sure your computer looks for the NIS server for key files means modifying the `/etc/nsswitch.conf` configuration file. For example, to make sure your computer looks to the NIS server for the username and password database, you'll want to configure the following commands in that file (you can add other services such as **ldap** to the list):

```
passwd:    files nis
shadow:    files nis
group:     files nis
```

This assumes that you're using the standard version of NIS that's included with RHEL 3, NIS version 2.

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

LDAP Client Configuration

If you want to configure your RHCE 3 computer as an LDAP client, you'll need the `openldap-clients`, `openldap`, and `nss_ldap` RPM packages. The `openldap-clients` RPM is a default part of the System Tools package group. The other two packages should be installed by default on your RHEL 3 computer.

To configure an LDAP client, you'll need to configure two different `ldap.conf` configuration files: in the `/etc` and the `/etc/openldap` directories. While both files can get quite complex, for the purposes of the exam, I'll keep the definitions simple.

`/etc/ldap.conf`

The default version of the `/etc/ldap.conf` file includes a number of different commands and comments. To set up your RHEL 3 computer as an LDAP client, you'll need to be concerned with four commands in this file, as described in Table 10-1.

`/etc/openldap/ldap.conf`

You'll need to specify the `HOST` and `BASE` variables in this file, just as you did in the `/etc/ldap.conf` configuration file. Based on the parameters in the previous section, this leads to the following two commands:

```
HOST 127.0.0.1
BASE dc=example,dc=com
```

If your LDAP server is not on the local computer, and your domain is not `example.com`, you'll need to substitute accordingly.

TABLE 10-1 Some `/etc/ldap.conf` Parameters

Command	Description
<code>host 127.0.0.1</code>	Specifies the IP address for the LDAP server. This particular command assumes the LDAP server is on the local computer.
<code>base dc=example,dc=com</code>	Sets the default base distinguished name, in this case, <code>example.com</code> .
<code>ssl start_tls</code>	Required if you want Transport Layer Security (TLS) support to encrypt passwords that are sent to the LDAP server.
<code>pam_password md5</code>	Supports the MD5 encryption scheme for passwords.

The Name Service Switch File

The Name Service Switch file, `/etc/nsswitch.conf`, governs how your computer searches for key files such as password databases. You can configure it to look through NIS and LDAP server databases. For example, when an NIS client looks for a computer hostname, it might start with the following entry from `/etc/nsswitch.conf`:

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

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

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

You can configure the `/etc/nsswitch.conf` configuration file to look at an LDAP server for the desired databases. For example, if you want to set up a centralized username and password database for your network, you'll need to configure at least the following commands in `/etc/nsswitch.conf`:

```
passwd:    files ldap
shadow:    files ldap
group:     files ldap
```

You can configure a number of additional files in an LDAP or NIS centralized database; however, the details are beyond the scope of this book.

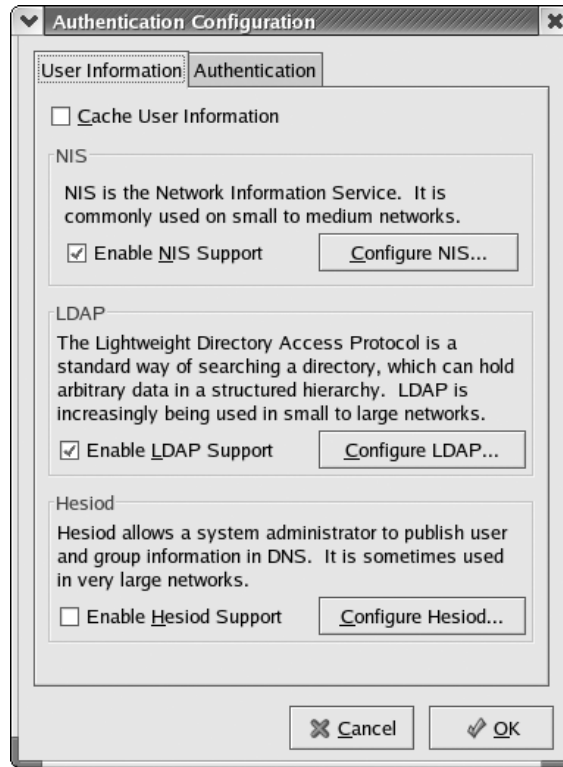
Configuring Clients with the Red Hat Authentication Tool

If you're not familiar with NIS or LDAP, it may be simpler to configure your computer as a client using the Red Hat Authentication Configuration tool. In the RHEL 3 GUI, you can start it with the **redhat-config-authentication** command or click Main Menu | System Settings | Authentication. This opens the Authentication Configuration tool shown in Figure 10-1.

If you've gone through the First Boot process described in Chapter 2, you may have already configured your computer using this tool. You can set your computer to check an NIS and an LDAP server for usernames and passwords.

FIGURE 10-1

Authentic
Configuration



If you select Enable NIS Support, click the Configure NIS button. You'll need to enter the name of the NIS domain, as well as the name or IP address of the computer with the NIS server. Make any desired changes and click OK.

If you select Enable LDAP Support, click the Configure LDAP button. Make any desired changes and click OK. You'll need to enter the following information:

- If you want to use Transport Layer Security to encrypt the passwords sent to the LDAP server, activate the Use TLS To Encrypt Connections option.
- Enter the domain name for your server in LDAP format in the LDAP Search Base DN text box. For example, if your domain is my-domain.com, enter the following:
`dc=my-domain,dc=com`
- Specify the location of the LDAP server. If it's the local computer, you can use the 127.0.0.1 IP address.

Under the Authentication tab, you can also specify the use of LDAP support for PAM-aware applications.

Once you've made your changes, click OK; it may take a few seconds for the Authentication Configuration tool to write the changes to the noted configuration files before it closes. If you've set up an NIS client, it automatically sets the **ypbind** service to start at appropriate runlevels the next time you boot Linux.

NIS Services

If you want to set up an NIS server on your computer, you'll need to install the **ypserv** RPM. It's a part of the Network Servers package group. You'll also need the **yp-tools** RPM, which should already be installed by default on a RHEL 3 computer. You can verify and install these RPM packages using the techniques described throughout the book.

NIS clients and NIS servers are organized in NIS domains. In contrast, LDAP clients rely on the LDAP server database. 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
```

If this returns a blank or (none), you'll need to define the NIS domain name for your system. It's easy to do; for example, the following command assigns the name **nisdomain** to your system:

```
# domainname nisdomain
```



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 world of Microsoft Windows servers, 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 RHEL 3 shares the following files:

- /etc/passwd
- /etc/group
- /etc/hosts

- /etc/tpc
- /etc/services
- /etc/protocols

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.

For larger networks, 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.



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.

Starting the NIS Service

Once you have edited the configuration files for your computer and network, you can start the NIS service. You can do so with the `ypinit` command in the `/usr/lib/yp` directory. This command has two key switches: `-m` is used to create a master NIS server; `-s` is used to create a slave NIS server. The process is straightforward. When you run the command to set up a master NIS server, you're prompted to enter the different hosts on the NIS domain. Here's what happened when I ran the command on my Enterprise3.example.com computer:

```
# /usr/lib/yp/ypinit -m
```

```
At this point, we have to construct a list of the hosts
which will run NIS servers. Enterprise3.example.com is
in the list of NIS server hosts. Please continue to add
the names for the other hosts, one per line. When you are
done with the list, type a <control D>.
```

```
next host to add: Enterprise3.example.com
next host to add:
```

As you can see, the process is straightforward. Once I complete the list with a CTRL-D, I'm asked to verify the list. Once I confirm, the NIS server takes a few minutes to build a database based on the files specified in `/var/yp/Makefile`. Once complete, you'll see a message to the effect:

```
Enterprise3.example.com has been set up as an NIS master server.
```

If you see errors, make sure that the NIS server daemon, `ypserv`, is running properly with the **service ypserv status** command.

You can now join this NIS domain from the NIS clients of your choice.

NIS Components on Red Hat Enterprise Linux

Once you've started the NIS Service, 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 10-2 lists the files that you can use 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`.

TABLE 10-2 Key NIS Configuration Files and Commands

File	Description
<code>/usr/lib/yp/ypinit</code>	A shell script to build initial database maps in <code>/var/yp</code> ; ypinit -m builds the databases for a master server.
<code>/var/yp/Makefile</code>	The main NIS configuration file. Edit this file to control which maps are shared via NIS. You should edit this file and run the make command in the <code>/var/yp</code> directory.
<code>/usr/lib/yp/makedbm</code>	Converts text database files to NIS maps. Called by <code>/var/yp/Makefile</code> .
<code>/usr/sbin/ypserv</code>	NIS server daemon. You should use a command such as chkconfig to make sure this starts the next time Linux boots on your computer.
<code>/usr/sbin/ypasswdd</code>	NIS password change daemon. You need to activate this service before users can change their NIS passwords with the yppass command. Use a command such as chkconfig to make sure this starts the next time Linux boots on your computer.
<code>/etc/ypserv.conf</code>	Another NIS server configuration file.
<code>/var/yp/securenets</code>	Controls which systems can access NIS databases.

NIS Access Control

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. While it isn't installed by default, this file is easy to create and configure. Only two lines are required for a LAN. You may notice that the lines appear backwards, as they specify IP network masks followed by network addresses:

```
255.0.0.0      127.0.0.1
255.255.255.0 192.168.0.0
```

The first line allows access from the local computer. The second line allows access from all of the computers with IP addresses on the 192.168.0.0 network. In RHEL 3, there's a template for this file, with comments, available in the `/usr/share/doc/ypserv-2.8` directory.



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 hyphen) to work with any account in your NIS database.

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** tool, 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.

exam

Watch

While this section does not directly relate to Red Hat exam requirements, it is fundamental to the way you secure your system and

network. I believe that anyone who wants to be an RHCE should be already following the principles described in this section.

Password Security

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

exam

Watch

On the Red Hat exams, follow the instructions carefully for any users and passwords. You do want to make sure that the person grading your exam can get into your accounts!

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 RHEL 3 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 10-2Suspicious
login activity

[5]	[01997]	[4]	]	[	]	[	]	[2.4.21-4.EL	]	[0.0.0.0
[5]	[01998]	[5]	]	[	]	[	]	[2.4.21-4.EL	]	[0.0.0.0
[6]	[01996]	[3]	]	[LOGIN	]	[tty3	]	[	]	[0.0.0.0
[6]	[01999]	[6]	]	[LOGIN	]	[tty6	]	[	]	[0.0.0.0
[5]	[01999]	[6]	]	[	]	[	]	[2.4.21-4.EL	]	[0.0.0.0
[6]	[01997]	[4]	]	[LOGIN	]	[tty4	]	[	]	[0.0.0.0
[6]	[01998]	[5]	]	[LOGIN	]	[tty5	]	[	]	[0.0.0.0
[6]	[01994]	[1]	]	[LOGIN	]	[tty1	]	[	]	[0.0.0.0
[6]	[01995]	[2]	]	[LOGIN	]	[tty2	]	[	]	[0.0.0.0
[7]	[01994]	[1]	]	[root	]	[tty1	]	[	]	[0.0.0.0
[7]	[02147]	[/0]	]	[root	]	[pts/0	]	[::0.0	]	[172.32.4.8
[7]	[02147]	[/1]	]	[root	]	[pts/1	]	[::0.0	]	[0.0.0.0

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 rh.n.redhat.com for updates. With Red Hat Enterprise Linux 3, this assumes you've already configured your system for updates through rh.n.redhat.com.

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 Samba to break into your system if the Windows File Server package group 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` and `/etc/rc.d/init.d` directories.

CERTIFICATION OBJECTIVE 10.03

The Pluggable Authentication Module (PAM) System

RHEL 3 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.

exam

Watch

PAM modules are documented in the `/usr/share/doc/pam-0.75/txts` directory. For example,

the functionality of the `pam_securetty.so` module is described in the `README.pam_securetty` file.

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 pam_securetty.so
```

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

on the job

In older versions of Red Hat Linux, the full path to the PAM module was required. It is now understood that these modules are stored in the `/lib/security` directory.

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 the `/etc/pam.d` directory.

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 (auth)** Establishes the identity of a user. For example, a PAM **auth** command decides whether to prompt for a username and or a password.
- **Account management (account)** 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 (password)** 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 (session)** Applies settings for an application. For example, the PAM **session** command may set default settings for a login console.

The code shown in Figure 10-3 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, can be **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.

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, as described in Table 10-3.

To demonstrate how control flags work, take a look at the commands from the `/etc/pam.d/reboot` configuration file:

```
auth sufficient pam_rootok.so
```

The first **auth** command checks the `pam_rootok.so` module. If the root user runs the reboot command, the **control_flag** is **sufficient**, the other **auth** commands in this file are ignored. Linux runs the reboot command. This is explained in the `README.pam_rootok` file in the `/usr/share/doc/pam-0.75/txts` directory.

```
auth required pam_console.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.

```
account    required    pam_permit.so
```

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.

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.



While it's not normal to allow just any user to shut down a corporate server, you may want to do so on a Linux workstation. In this way, users can shut down their own laptop or desktop without having to know 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 10-3. When a user opens a text console and logs in, Linux goes through this configuration file line by line. As previously noted, the first line in `/etc/pam.d/login`:

```
auth    required    pam_securetty.so
```

limits root user access to secure terminals as defined in the `/etc/securetty` file. The next line brings the login program through the following service, `system-auth`, which also happens to be a PAM configuration file.

```
auth    required    pam_stack.so service=system-auth
```

Essentially, this calls the **auth** commands in the `/etc/pam.d/system-auth` configuration file shown in Figure 10-4. This sets up environment variables and allows different users to log in.

```
auth    required    pam_nologin.so
```

The last **auth** line from `/etc/pam.d/login` checks the `/etc/nologin` file.

```
account    required    pam_stack.so service=system-auth
password    required    pam_stack.so service=system-auth
```

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

```

#%PAM-1.0
# This file is auto-generated.
# User changes will be destroyed the next time authconfig is run.
auth        required      /lib/security/$ISA/pam_env.so
auth        sufficient     /lib/security/$ISA/pam_unix.so likeauth nullok
auth        required      /lib/security/$ISA/pam_deny.so

account     required      /lib/security/$ISA/pam_unix.so

password    required      /lib/security/$ISA/pam_cracklib.so retry=3 type=
password    sufficient     /lib/security/$ISA/pam_unix.so nullok use_authtok md5
shadow
password    required      /lib/security/$ISA/pam_deny.so

session     required      /lib/security/$ISA/pam_limits.so
session     required      /lib/security/$ISA/pam_unix.so
~
~
~
~
~
~
~
"/etc/pam.d/system-auth" 15L, 688C

```

The **account** and **password** commands in `/etc/pam.d/login` also refer to the `/etc/pam.d/system-auth` configuration file. For more information, refer to the **account** command in `/etc/pam.d/system-auth`:

```
account    required    /lib/security/$ISA/pam_unix.so
```

exam

Watch

If the `/etc/nologin` file exists, regular users are not allowed to log into your console. Any regular user that tries to log in gets to read the contents of `/etc/nologin` as a message.

This refers to the `pam_unix.so` module in the `/lib/security` directory. The value of `$ISA` is normally null (empty), so you can ignore this variable. The `pam_unix.so` module sets up the normal username and password prompts.

For more information on the `/etc/pam.d/login` password command, you'll need to refer to the three **password** commands in `/etc/pam.d/system-auth`:

```
password required /lib/security/$ISA/pam_cracklib.so retry=3 type=
password sufficient /lib/security/$ISA/pam_unix.so nullok use_authok md5
```

```
shadow nis
password required /lib/security/$ISA/pam_unix.so
```

The first command from this list sets a maximum of three retries. The next command allows the use of null (zero-length) passwords, prompts the user for a password (**use_authok**), encrypts passwords using the MD5 algorithm, supports the *shadow* password suite described in Chapter 1, and allows the use of NIS passwords.

Finally, there are two **session** commands in the `/etc/pam.d/login` file:

```
session required pam_stack.so service=system-auth
session optional pam_console.so
```

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`. 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 Enterprise Linux 3.

1. Make a backup copy of `/etc/securetty` with the following command:

```
# cp /etc/securetty /etc/securetty.sav
```
2. Edit `/etc/securetty` and remove the lines for `tty3` through `tty11`. 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 step 3 as a regular user. What happens? Do you know why?
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 with the following command:

```
# mv /etc/securetty.sav /etc/securetty
```

One thing to remember is that the `/etc/securetty` file governs the consoles from where you can log into Linux as the root user. Therefore, the changes that were made do not affect regular (non-root) users.

Securing PAM by User

In this section, you’ll learn how to configure PAM to limit access to specific users. The key to this security feature is the `pam_listfile.so` module in the `/lib/security` directory. As described earlier, there are four settings in each PAM configuration command. To make sure that command respects what you do with this module, the first parts of the command should be:

```
auth required pam_listfile.so
```

The way PAM limits user access is in the last part of the command—in the details. For example, if you added the following line to a PAM configuration file, access to the associated tool would be limited to any users listed in `/etc/special`:

```
auth required pam_listfile.so onerr=succeed item=user \
sense=allow file=/etc/special
```



A backslash in a command line “escapes” the meaning of the next character; in the previous command, `file=/etc/special` is appended to the end of the command line. Due to limits in the format of this series, I’ve had to change the spacing of some lines and add backslashes to others.

To understand how this works, it’s time to break this command into its component parts. You already know the first three parts of the command from the previous section. The switches that are shown are associated with the `pam_listfile.so` module, as described in Table 10-4. Based on the topic matter for the exam, this table is limited; for full details, see `README.pam_listfile` in the `/usr/share/doc/pam-0.75/txts` directory.

TABLE 10-4 Switches for the `pam_listfile.so` Module

pam_listfile Switch	Description
onerr	If there is a problem, tell the module what to do. The options are onerr=succeed or onerr=fail .
item	You can use this switch to limit access to a terminal (tty), users in a specific file (user), groups (group), or more.
sense	If the item is found in the specified file , take the noted action. For example, if the user is in <code>/etc/special</code> , and sense=allow , then this command allows use of the specified tool.
file	Configures a file with a list, such as file = /etc/special .

Thus for the specified command, (**onerr=succeed**) an error returns success, (**item=user**) based on a specific list of users. If the user is in the specified list (**file=/etc/passwd**), allow that user (**sense=allow**) to access the specified tool. To see how this works, run through the steps in the following exercise.

e x a m

W a t c h

Make sure you understand how Red Hat Enterprise 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).

EXERCISE 10-2

Using PAM to Limit Access

You can also use the Pluggable Authentication Module system to limit access to regular users. In this exercise, you'll limit access by adding one or more users to the `/etc/nologin` file. It should work hand-in-hand with the default `/etc/pam.d/login` security configuration file, specifically the following line:

```
auth required pam_nologin.so
```

1. Look for an `/etc/nologin` file. If it doesn't already exist, create one with a message such as:

```
I'm sorry, access is limited to the root user
```

2. Access another terminal with a command such as `CTRL-ALT-F2`. Try logging in as a regular user. What do you see?

CERTIFICATION OBJECTIVE 10.04

System Logging Configuration

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 Enterprise Linux 3 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.

RHEL 3 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 Enterprise 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 through the `/etc/rc.d/init.d/syslog` script. Once these daemons start, the syslog daemon examines `/etc/syslog.conf` to find the logging options that you may have configured.

System Log Configuration File

You can configure what `syslogd` records through the `/etc/syslog.conf` configuration file. As shown in Figure 10-5, it includes a set of rules for different facilities: `auth`, `authpriv`, `cron`, `daemon`, `kern`, `lpr`, `mail`, `mark`, `news`, `security`, `syslog`, `user`, and `uucp`.

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`. There's also a generic "none" priority which 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`

FIGURE 10-5

The 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
"/etc/syslog.conf" 33L, 937C
```

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 the syslog daemon 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 10-6, 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 through applicable logs. It can be helpful if you don't remember the locations of the key log files and don't remember to look through /etc/syslog.conf for those locations. In the GNOME Desktop, you can start this tool from the console with the **redhat-logviewer** command or click Main Menu | System Tools | System Logs. Either option opens the tool shown in Figure 10-7.

However, this package took several minutes to load on my computer, time that you might not have during the Red Hat exams. As the Red Hat Log Viewer simply provides a front end, it's best if you know the files you need in the /var/log directory.

For example, you can review the information shown in Figure 10-7 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. On the other hand, the Red Hat Log Viewer does include warning and trouble icons that can help you identify

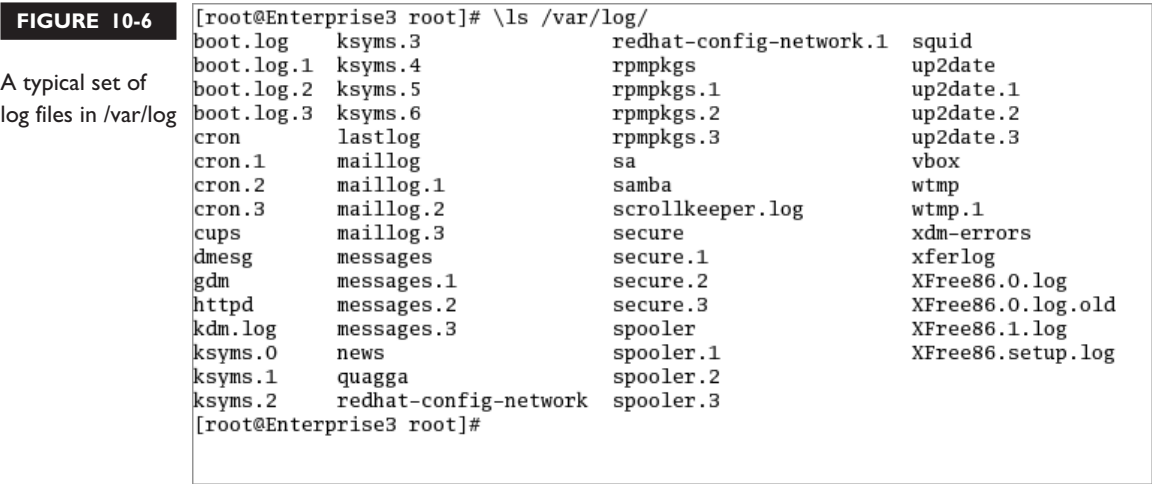
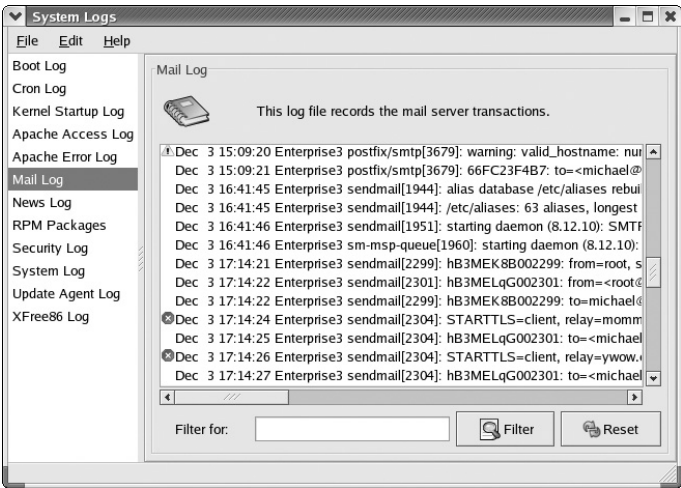


FIGURE 10-7

Red Hat System
Log Viewer



problems with specific services. Different log files and their functionality are described in Table 10-5. This is based on the previously described configuration of the `/etc/syslog.conf` file. All files shown are in the `/var/log` directory. As you can see, the System Log tool does not cover all log files.

TABLE 10-5 Standard Red Hat Log Files

Log Files	Label in the System Log Tool	Description
boot.log	Boot Log	Associated with services that start and shut down process
cron	Cron Log	Based on scripts run by the cron daemon
cups/	<i>Not included</i>	Directory of printer access, page, and error logs
dmesg	Kernel Startup Log	Basic boot messages
gdm/	<i>Not included</i>	Directory of messages associated with starting via the GNOME Display Manager; includes login failures
httpd/	Apache Access Log Apache Error Log	Directory of log files associated with the Apache Web server
ksyms	<i>Not included</i>	Messages binding kernel modules (kallsyms for Linux Kernel 2.6)
maillog	Mail Log	Log messages related to the sendmail and postfix mail servers
messages	<i>Not included</i>	Messages from other services as defined in <code>/etc/syslog.conf</code>

TABLE 10-5 Standard Red Hat Log Files (*continued*)

Log Files	Label in the System Log Tool	Description
news/	News Log	Directory of log messages related to the Internet Network News (INN) service
quagga/	<i>Not included</i>	Directory of routing messages
redhat-config-network	<i>Not included</i>	Actions taken though the Red Hat Network Configuration tool
rpm_pkgs	RPM Packages	Current list of installed RPM packages
sa/	<i>Not included</i>	List of memory usage information
samba/	<i>Not included</i>	Directory of access and service logs for the Samba server
scrollkeeper.log	<i>Not included</i>	Log related to GNOME documentation
secure	Security Log	Lists login and access messages
spooler	<i>Not included</i>	Print spool log
squid/	<i>Not included</i>	Directory of files related to Squid Proxy Server access, cache, and storage
up2date	Update Agent Log	Lists actions taken when you've updated your server using the Red Hat Update Agent (up2date)
wtmp	<i>Not included</i>	List of logins, in binary format; can be read with the utmpdump command
xdm-errors	<i>Not included</i>	Messages associated with the X Display Manager
xferlog	<i>Not included</i>	Log of files transferred from other computers, such as via FTP
XFree86	XFree86 Log	Messages during Linux GUI start process
XFree86.setup.log	<i>Not included</i>	Setup messages for the X Window System; may include configuration problems

EXERCISE 10-3**Checking Logs**

In this exercise, you'll inspect the log files on your computer to try to identify different problems. You'll use the Red Hat System Log viewer, and inspect log files directly.

1. Restart your Linux computer. Log in as the root user. Use the wrong password once. Log in properly as the root user.
 2. Open a GUI desktop interface.
 3. Open the Red Hat System Log Viewer. Click Main Menu | System Tools | System Logs. Depending on the speed of your computer and the size of your logs, this may take a few minutes.
 4. In the System Logs window, navigate to the Security Log area in the left pane. Scroll to the bottom of the screen. You should see a red “X” with a “FAILED LOGON” message.
 5. In a console, navigate to the `/var/log` directory and open the file named `secure`. Navigate to the same “FAILED LOGON” message.
 6. Look through the other logs in the System Logs window and close this tool. Close the `secure` file.
-

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. This section assumes that you have the default RHEL 3 `krb5-workstation` RPM package, which includes a more secure version of Telnet.



If you're working with Red Hat Linux 9, you can also work with the `telnet-server` RPM. Some of the commands with this older Telnet server are different from what you see here.

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 10-6.

TABLE 10-6 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)

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. There's a generic configuration file for xinetd services, /etc/xinetd.conf. The scripts in the /etc/xinetd.d directory also function as service specific configuration files.

Generic xinetd Configuration

The generic configuration for xinetd services is stored in the /etc/xinetd.conf file. It's short and fairly straightforward, so we can analyze it line by line. First, there are a number of default settings that are started with the following command:

```
defaults
```

This allows services such as POP3 to retain their default TCP/IP ports (110). This is followed by:

```
instances = 60
```

which limits the number of active services for a particular service; in this case, no more than 60 users can be logged into your Kerberos Telnet server simultaneously. The following line means that log messages are sent as specified in /etc/syslog.conf.

```
log_type = SYSLOG authpriv
```

The following line means that for each successful logon to a xinetd service, the name (or IP address) of the remote computer and the process ID (PID) are sent to the appropriate log file.

```
log_on_success = HOST PID
```

The next line means that whenever there is a failed attempt to log on to an xinetd service, the name (or IP address) of the remote computer and reason is sent to the appropriate log file.

```
log_on_failure = HOST
```

The **cps** command prevents attempts to “flood” any xinetd service; this line limits connections to 25 per second. If this limit is exceeded, xinetd waits 30 seconds before allowing a remote user to try again.

```
cps = 25 30
```

Finally, the last line supports the use of the other configuration files specified in the `/etc/xinetd.d` directory.

```
includedir /etc/xinetd.d
```

Sample xinetd Configuration

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 code shows a sample of the `/etc/xinetd.d/krb5-telnet` configuration file, with this service disabled:

```
# default: on
# description: The telnet server serves telnet sessions; it uses \
#              unencrypted username/password pairs for authentication.
service telnet
{
    flags            = REUSE
    socket_type      = stream
    wait            = no
    user            = root
    server           = /usr/Kerberos/sbin/telnetd
    log_on_failure   += USERID
    disable         = yes
}
```

TABLE 10-7 Standard Parameters for xinetd Configuration Files

Field	Description of Field Entry
flags	Supports different parameters for the service; REUSE is a default which supports continuous use of the service. Options include IPv6 to set this as a service for those types of networks.
socket_type	Specifies the communication stream.
wait	Set to 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	Hostname or IP address allowed to use the server. CIDR notation (such as 192.168.0.0/24) is okay.
no_access	Hostname or IP address not allowed to use the server. CIDR notation is okay.
log_on_failure	If there's a failed login attempt, this specifies the information sent to a log file.
disable	Yes by default, which disables the service.

This is a typical /etc/xinetd.d configuration file. The variables (and a few additional variables that you can use) are described in Table 10-7. 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.

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 **service xinetd reload** command.

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

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 Red Hat exams do not require any detailed understanding of IPv6 addresses.

exam

watch

Always remember to make sure that a service will be active after a reboot. The `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—and you may not get credit for how you configured that service on your exam.

EXERCISE 10-4

Configuring xinetd

In this exercise, you will enable the Telnet service using `xinetd`. Attempt to establish a Telnet session using the command `telnet localhost`. If you're successful, Telnet is already enabled; disable it first with the `chkconfig krb5-telnet off` command.

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:

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

Security by User or Host

The best way to prevent a cracker from using a service is to remove it completely from your Linux system. However, you may want to keep a service loaded because you're planning to use it in the near future.

You can achieve some measure of security by disabling or removing unused services in the `/etc/xinetd.d` directory. 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

hostname or IP address. Alternatively, you can block access to specific users, computers, or even networks through the `hosts.allow` or `hosts.deny` files in the `/etc` directory. 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 `hosts.allow` and `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 for commands in each file is as follows:

```
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:

```
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 10-8.

As you can see in Table 10-8, 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.

TABLE 10-8 Sample Commands in /etc/hosts.allow and /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.

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 :.example.com
telnetd : 192.168.25.0/255.255.255.0 EXCEPT 192.168.25.73
in.pop3d, in.tftpd : 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 example.com domain. The following line opens the Telnet service to any computer on the 192.168.25.0 network, except the one with an IP address of 192.168.25.73. Then the POP3 and TFTP 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.tftpd : .xyz.com
telnetd : ALL EXCEPT 192.168.1.10
ALL:ALL
```

The first line in the hosts.deny file is a comment. The second line denies all services except TFTP to computers in the .xyz.com domain. The third line states that the only computer that is allowed to access our Telnet server 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.

TABLE 10-9

tcp_wrappers Operators

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

You can also use the **twist** or **spawn** command in `/etc/hosts.allow` or `/etc/hosts.deny` to access shell commands; primarily they're intended to send messages, track access, and log problems. For example, take the following line in a `/etc/hosts.deny` file:

```
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 10-9. Some of these operators may be able to help you track the intruder.

EXERCISE 10-5

Configuring tcp_wrappers

In this exercise, you 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:

```
telnetd : 127.0.0.1
```
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.

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 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 Enterprise Linux 3 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. But the default is **iptables**, which is available by default for RHEL 3.

exam

Watch

The Linux 2.4.x kernel firewalls subsystem is also known as the netfilter/iptables project. The current RH300 course refers

to netfilter. The Red Hat firewall tools create iptables (IP version 4) commands. Therefore, I've focused this section on iptables.



RHEL 3 also includes a firewall command for IPv6 networks, *ip6tables*.

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 **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 **rmmod *modulename*** command to delete any **ipchains**-related modules. Then use the **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 configures 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***, the **iptables** command assumes that you’re trying to affect a filtering rule. Next is the **<action/direction>**. There are four basic actions associated with **iptables** rules:

- **-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** extension prevent users outside your network from looking for an HTTP connection.

Once the **iptables** command finds a packet pattern match, 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:

```
# 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:

```
# 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” your system (remember that the **ping** command uses the ICMP protocol):

```
# 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 your network IP address is 192.168.1.0. The exclamation point (!) inverts the meaning; in this case, the command applies to all IP addresses except those with a 192.168.1.0 network address (and a 255.255.255.0 subnet mask).

```
# iptables -A INPUT -s !192.168.1.0/24 -p tcp -j DROP
```

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

```
# 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:

```
# iptables -A FORWARD -j DROP
```

Maintaining Netfilter Rules

Once you’ve added the **iptables** commands of your choice, the following command saves your new firewall configuration to a file:

```
# 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:

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

exam

Watch

Knowing how to secure a Red Hat Enterprise Linux system against unauthorized access is critical.

Be sure you understand the concepts and commands discussed in this chapter.

The Red Hat Firewall Configurator

You can automate the process of configuring a firewall. RHEL 3 includes the Security Level Configuration tool. You can start it with the **redhat-config-securitylevel** command, or by clicking Main Menu | System Settings | Security Level. This is a straightforward tool, as shown in Figure 10-8.

If you've installed RHEL 3 before, this menu should look familiar; the choices are identical to those shown during the standard RHEL 3 installation process. There is a similar text-based version of this tool, which you can start with the **redhat-config-securitylevel-tui** or **lokkit** commands.

Red Hat has changed this tool for RHEL 3; there are no longer three security levels available. You can enable or disable a firewall for your computer. The default firewall loosely corresponds to the "High Security" firewall that you could have configured for Red Hat Linux 9.

- The default RHEL firewall blocks all inbound request traffic unless requested from within the network. For example, DNS replies are allowed.
- 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.

FIGURE 10-8

The Security Level Configuration tool



You can create exceptions to each rule. First, firewalls are not applied to “Trusted Devices.” Therefore, if you have multiple network card on your computer, you can apply firewall rules only to one network card—say, on the card that is directly exposed to the Internet.

As shown in Figure 10-8, you can allow incoming traffic to a number of services. For example, if you select WWW (HTTP), others can connect to a Web server on your computer. With the available settings, you can also allow incoming connections to

- A Web server, by activating the WWW (HTTP) option.
- An FTP server such as the vsFTP service, by activating the FTP option.
- A Secure Shell (SSH) service, by activating the SSH option. This is a common method for administering remote Linux computers; you’ll read more about SSH in Chapter 11.
- Telnet, by activating the Telnet option. This also works with the Kerberos-based Telnet service described earlier in this chapter.
- Mail services through the sendmail or Postfix services described in Chapter 7, by activating the Mail (SMTP) option.

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.

In this case, the text-based version of this tool is more customizable. As you can see in Figure 10-9, you can use the “Other Ports” text box to allow data to come through using other incoming TCP/IP ports.

For example, if you wanted to allow secure Web communication through your firewall, you could enter the following line in the Other Ports text box:

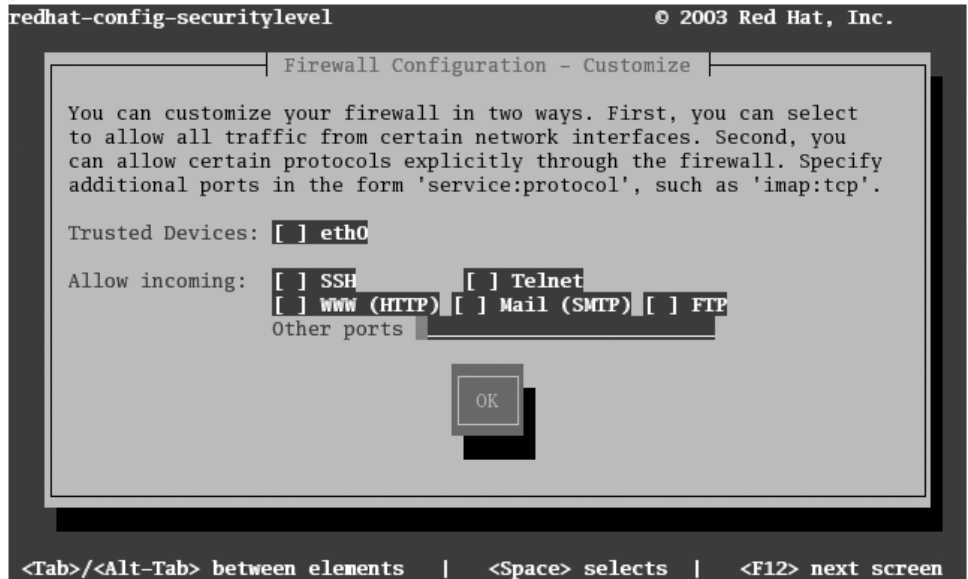
```
443:udp,443:tcp
```

This opens communication through the TCP/IP ports associated with Secure Web services, and is equivalent to the following **iptables** commands:

```
# iptables -A INPUT -p udp --dport 443 -j ACCEPT
# iptables -A INPUT -p tcp --dport 443 -j ACCEPT
```

FIGURE 10-9

Customizing
Using the Red
Hat text-based
Security Level
tool



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, which also serves as a gateway between your network and the Internet. The source address is cached on the gateway, so it knows which computer made the request.

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 Enterprise 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 that you do the following:

- 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`:

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

The following command enables FTP access through your firewall:

```
# modprobe -a ip_conntrack_ftp ip_nat_ftp
```

Similar modules are available in your kernel directory, in the following subdirectory:

```
/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 Enterprise 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 Enterprise Linux, refer to the following Scenario & Solution for some possible scenario questions and their answers.

SCENARIO & SOLUTION

You have installed an e-mail 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 e-mail access (ipop3d) 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.

CERTIFICATION SUMMARY

One of the basic functions of a Red Hat Enterprise Linux system administrator is to protect a Linux computer and a network from inside and outside attacks. RHEL 3 includes a variety of tools that can help you establish a secure computing environment.

RHEL 3 includes powerful tools 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 RHEL 3 system as well as the computers on your 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 Clients and Servers

- ☐ 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.
- ☐ 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.

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.
- ☐ 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 Enterprise 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-0.75/txts` directory.

System Logging Configuration

- ☐ Red Hat Enterprise 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, 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 Enterprise 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 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 **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. It replaces the source address with the public address of the firewall computer, with a random port number.
- ☐ 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 random 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.
- ☐ To enable IP forwarding, edit `/etc/sysctl.conf` and change the line `net.ipv4.ip_forward = 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. 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.

Configuring NIS Clients

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. What is the most efficient way to 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 `USE_NIS` command at the end of the file.
 - b. Start the `ybind` daemon, and configure it to start the next time you boot Linux.
 - c. Add a command to `/etc/xinetd.conf` to start the `ybind` daemon.
 - d. Run the Red Hat Authentication Configuration Utility and enable NIS.

Basic Host Security

3. 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

4. What are the four areas associated with Pluggable Authentication Modules?
 - 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
5. 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.

System Logging Configuration

6. Assume you normally work from a user account called sysadm. How might you configure your Red Hat Enterprise Linux 3 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)

7. You are using the xinetd program to start services. How could you limit access to 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:

```
telnetd : 192.168.170.0/255.255.255.0
```

- c. Edit /etc/hosts.deny and add this line:

```
telnetd : 192.168.170.0/255.255.255.0
```

- d. Edit /etc/hosts.deny and add this line:

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

Firewall Policies

8. 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 400 MHz PC with 256MB of RAM 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 alleviate your concerns?

- a. Nothing, you're not advertising the systems on your LAN via DNS, so no one will ever find them.
- b. Install RHEL 3 on the spare PC and use **iptables** to set it up as a firewall.
- c. Install RHEL 3 on the spare PC and use **tcp_wrappers** to set it up as a firewall.
- d. Install Linux on all systems on your network.

9. Consider the following command:

```
# 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

8. 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?
- Assign the official IP address to a Linux system and create accounts on that system for all of the office personnel.
 - Install Linux and configure it for IP forwarding.
 - Install a Linux router.
 - Use the Linux system to connect to the Internet; then use **iptables** to set up IP masquerading.

LAB QUESTIONS

Lab 1

You have a growing network of Linux computers, and have to maintain users and passwords on each of these computers on a daily basis. You're having to update administrative files such as `/etc/passwd` on a number of computers. What can you do to simplify your task?

Lab 2

You want to set up a RHEL 3 computer as a secure Web server. To keep that system secure, you'll want to configure an appropriate firewall, and disable any services that you don't need. What should you do?

Lab 3

You want to make sure even the root user has to enter the root password when opening Red Hat administrative tools. You can do this by modifying the appropriate file in the `/etc/pam.d` directory. Try this out with the Red Hat Security Configuration tool.

Lab 4

In this lab, you'll see how you can limit access to specific users through the PAM listfile module. In this lab, you'll limit access to the Secure Shell that's covered in Chapter 11. Assume that you have four users on your system: michael, donna, randy, and nancy, and want to limit access to randy and nancy. What do you need to do to make this happen?

Lab 5

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?

Lab 6

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?

SELF TEST ANSWERS

Configuring NIS Clients

1. ☒ **B**. This is an ideal situation for NIS, where you can set up a single database of usernames and passwords.
☒ **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 Red Hat Authentication Configuration utility.
☒ **A** is incorrect because this is invalid syntax. **B** is incorrect because you need to do more than start ypbind; you also need to configure the NIS domain. **C** is incorrect because ypbind is an independent service that you need to start from the `/etc/rc.d/init.d` directory.

Basic Host Security

3. ☒ **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

4. ☒ **A**. PAM breaks the authentication process into these four areas.
☒ **B**, **C**, and **D** are not the four areas associated with PAM.
5. ☒ **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 can 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.

System Logging Configuration

6. ☒ **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. **D** is also incorrect, as explained earlier.

The Extended Internet Services Daemon (xinetd)

7. ☒ **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.

Firewall Policies

8. ☒ **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 DMZ 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** is a good option in principle, it won't necessarily make your network more secure.
9. ☒ **C.** Because the target is set to **REJECT**, the client will receive an error message. If the target was set to **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

10. ☒ **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 ANSWERS

Lab 1

You can set up an NIS server to maintain a common database of usernames and passwords. This should include at least the basic password database files, such as `/etc/passwd`, `/etc/group`, `/etc/shadow`, and `/etc/gshadow`, as defined in the `/var/yp/Makefile` configuration file.

Before you can set up an NIS server, you need to make sure you have the packages that you need, specifically the `ypserv` and `yp-tools` RPM packages. You can check and install these packages using the `rpm` command, as described throughout the book. Once installed, you'll want to start the `ypserv` service in the `/etc/rc.d/init.d` directory. You'll also want to use the `chkconfig` command to make sure this service starts the next time you boot this computer.

You'll need to set up an NIS domain name with the `domainname` command. You can then configure the NIS master server with the following command:

```
# /var/lib/yp/ypinit -m
```

This command assumes the local computer should also be configured as an NIS client on the given network. You're then prompted to enter the hostnames of other computers that you want to add to the NIS domain.

On a larger network, it can be helpful to have a backup for the NIS master server. If the NIS master server hostname is `NISmaster`, you can set this up with the following command:

```
# /var/lib/yp/ypinit -s NISmaster
```

You can then set up clients by configuring the `ypbind` service on each computer on the NIS domain. Make sure that the `ypbind` service starts the next time each computer restarts with a command such as:

```
# chkconfig --level 35 ypbind on
```

Lab 2

If you want to set up a RHEL 3 computer as a secure Web server, it's a straightforward process. You'll want to set up a firewall to block all but the most essential ports. This should include TCP/IP ports 80 and 443, which allow outside computers to access your regular and secure Web services.

The easiest way to set this up is with the text-mode Red Hat Security Level configuration tool, which you can start with the **lokkit no change to this in RHEL?** or **redhat-config-securitylevel-tui**

commands (**lokkit** is now a “front-end” to **redhat-config-securitylevel-tui**). Once you’re in the Red Hat tool, take the following steps:

1. Enable the firewall. This configures a basic set of firewall rules that prohibits access except for requests from come from inside the firewall.
2. Click Customize. This opens the Firewall Configuration – Customize window. In the Allow Incoming section, activate the WWW (HTTP) option. This allows outside access to your regular Web site.
3. In the Other Ports text box, type the following, which opens the ports associated with a standard secure Web service:

```
443:tcp,443,udp
```

4. Click OK. Click OK again to exit from the Firewall Configuration tool.
5. Enter the following command to check your resulting firewall.

```
# iptables -L
```

6. Once you’ve configured a Web service as described in Chapter 7, you’ll be able to access both the regular and secure Web servers from remote computers.

Lab 3

To make lab work, you’ll need to modify the Security Level Configuration tool using the **redhat-config-securitylevel** file in the **/etc/pam.d** directory. Open this file in the text editor of your choice. The first two commands allow users to start this tool automatically:

```
auth    sufficient    pam_rootok.so
auth    sufficient    pam_timestamp.so
```

The first command checks if you’re the root user. The second command checks to see if you’ve opened the given tool recently, based on the conditions of the **pam_timestamp** module. If you deleted (or commented out) these commands, all users, including the root user, will have to enter the root password when opening this tool. To do so, take the following steps:

1. Open the Red Hat Security Level Configuration tool in a command line in your GUI. Make sure it opens normally. When it does, close it without making any changes to your current firewall.
2. Back up the current PAM module for the Security Level Configuration tool to your home directory:

```
# cp /etc/pam.d/redhat-config-securitylevel ~
```

3. Open the file in **/etc/pam.d** in the text editor of your choice. Comment out the first two commands in this file.

4. Save the file. If you're not already logged into the GUI as the root user, log out of the GUI. Log back into the GUI as root.
5. Try opening the Red Hat Security Level Configuration tool. Click Main Menu | System Settings | Security Level. What happens?

Lab 4

To limit access to a PAM configured tool to specific users, you need a bit of help from the PAM listfile.so module, /etc/security/pam_listfile.so. With the following steps, I'm assuming that you need to configure the four specified users; you can configure existing users of your choice.

1. Create the users michael, donna, randy, and nancy. Set up passwords for these users. There are a number of ways to do so; I believe the simplest is with the following commands. Naturally, you'll need to enter an appropriate password for each user when prompted.

```
# useradd michael; passwd michael
# useradd donna; passwd donna
# useradd nancy; passwd nancy
# useradd randy; passwd randy
```

2. Navigate to the /etc/pam.d directory. Look to see if the sshd file exists; if not, you'll have to install it from the openssh-server RPM. While the technique is described in Chapter 11, it is no different from the technique used to install other services throughout this book. It should already be installed by default on a RHEL 3 server computer.
3. Open the /etc/pam.d/sshd file in the text editor of your choice. By default, the first two commands should look like:

```
auth required    pam_stack.so service=system-auth
auth required    pam_nologin.so
```

4. After the final **auth** command, enter the **auth** command that you need to point to a file with allowed users. (You could certainly use the **sense=deny** switch and configure a file with disallowed users.)

```
auth required pam_listfile.so onerr=succeed item=user \
sense=allow file=/etc/special
```

5. Since you want users nancy and randy to have access to the Secure Shell service, you'll want to add their usernames to the /etc/special configuration file, one username per line.
6. Try logging into the Secure Shell service with a prohibited username such as michael:

```
# ssh michael@localhost
```

7. If you haven't logged into the Secure Shell service before from this computer, you'll be prompted to verify the authenticity of the unknown computer. Since it's the local computer, you can assume it's reasonably safe; type **yes** and press ENTER.
8. Enter user michael's password. What happens? Can you get access?
9. Press CTRL-C to exit from the Secure Shell login process. Try again with donna's username and password. What happens? Is this what you expected?
10. Try again with the allowed users.
11. What happens if you return to the `/etc/pam.d/sshd` file and change **sense=allow** to **sense=deny**?

Lab 5

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 `krb5-telnet` RPM is installed.
2. Activate Telnet. Use the **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. (If you have another computer on your network with a private IP address, substitute accordingly in all steps in this lab.)
4. Save the configuration file and reload the xinetd service script with the **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 **service xinetd reload** command.
7. Edit `/etc/hosts.deny`. Add the **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 existing **iptables** chains. Back up `/etc/sysconfig/iptables`, if that file currently exists to `~/bak.iptables`.

11. Flush current firewall rules with the **iptables -F** command.
12. Block the Telnet port, 23, for all IP addresses except 192.168.1.33 with the **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 **iptables -F** command.
15. Restore any previous firewall rules with the **iptables-restore < ~/bak.iptables** command.
16. Bonus Lab: Repeat these commands for other services and networks.

Lab 6

Scenario 1: Cost is not an object. This means you can build a DMZ 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.