13 Electronic Mail Servers

CERTIFICATION OBJECTIVES

13.01	A Variety of E-Mail Agents	✓	Two-Minute Drill
13.02	The Configuration of Postfix	Q&A	Self Test
13.03	The Other SMTP Service: sendmail		

Linux offers a number of alternative methods for handling incoming and outgoing e-mail. RHEL 6 includes sendmail and Postfix for this purpose. Yes, it includes Dovecot, Fetchmail, and Procmail as well, but since the RHCE objectives focus on services associated with the Simple Mail Transfer Protocol (SMTP), this chapter focuses on sendmail and Postfix, as they are the two supported services associated with SMTP.

The default RHEL 6 SMTP service is now Postfix. That service was first developed in the late 1990s as an alternative to sendmail. It was designed to be easier to configure. It's feasible for most administrators to directly edit the associated configuration files. Now that Red Hat has changed default SMTP services from sendmail to Postfix, they've reached another milestone.

Despite Red Hat's move toward Postfix, sendmail is perhaps still the most common server for SMTP services. It was the default service through RHEL 5. Red Hat still supports it for RHEL 6. Once it is installed and configured, sendmail can be configured as an e-mail server for anything from an enterprise to a smart host for a personal system, subject to the limitations of an ISP. RHEL includes the open-source version of sendmail; the commercial version is the Sentrion Message Processing Engine from the company known as Sendmail (with the capital S).

For the purpose of this chapter, both Postfix and sendmail were installed on the physical host system. Smart host versions of each server were installed on the server1.example.com system. Access tests were performed from the VMs configured in Chapters 1 and 2, representing different external networks.

on the job

A number of alternatives to Postfix and sendmail are not covered in this book; they include procmail, mail.local, exim, Cyrus IMAP, and uucp.

INSIDE THE EXAM

The objectives related to e-mail services on the RHCE exam are relatively simple. First, as the focus is on SMTP services, the focus is on services directly associated with SMTP. The major SMTP services available for RHEL 6 are Postfix and sendmail. While Postfix is installed by default, you're certainly free to install and configure sendmail as an alternative to meet the noted requirements, specifically to

- Configure a mail transfer agent (MTA) to accept inbound e-mail from other systems

- Configure an MTA to forward (relay) e-mail through a smart host

A related clue comes from the Red Hat prep course for the RHCE, which includes an objective to "configure an SMTP server for basic operation (null client, receiving mail, smarthost relay)."

A null client is a system that can only send mail. A system that can receive mail is normally limited to the local network; however, such systems can be configured to receive e-mail from other networks as well. Mail can also be sent to remote systems. If a firewall or perhaps an ISP requires e-mail to be sent through their servers, you can configure SMTP services as smart hosts, which forwards information to such services.

In addition, you also need to meet the basic RHCE objectives that apply to all network services, as discussed in Chapter 11.

CERTIFICATION OBJECTIVE 13.01

A Variety of E-Mail Agents

With either Postfix or sendmail comes serious configuration files that may seem cryptic to administrators who are newer to e-mail administration. Do not let the size of the configuration files intimidate you. Just a few changes are required to meet the requirements associated with the RHCE objectives. In this section, you'll explore where SMTP services fit in the hierarchy of e-mail services.

Definitions and Protocols

A mail server has four major components, as described in Table 13-1. On any Linux computer, you can configure a mail transfer agent (MTA) such as Postfix or sendmail for various outbound services, such as forwarding, relaying, smart host communication with other MTAs, aliases, and spooling directories. Other MTAs, such as Dovecot, are designed to handle only incoming e-mail services, based on the protocols it serves, POP3 (Post Office Protocol, version 3) and IMAP4 (Internet Message Access Protocol, version 4).

TABLE 13-1

Mail Server Components

Abbreviation	Meaning	Examples
MTA	Mail transfer agent	Postfix, sendmail, Dovecot
MUA	Mail user agent	mutt, Evolution, mail, Thunderbird
MDA	Mail delivery agent	procmail
MSA	Mail submission agent	Postfix, sendmail

E-mail systems are heavily dependent on name resolution. While you could handle name resolution through /etc/hosts on a small network, any mail system that requires Internet access needs access to a fully functional DNS server. For spam protection and more, it's important to make sure that the system that intends to send an e-mail is actually transmitting with the assigned IP address.

But that is only one component of how e-mail works, from transmission to delivery. E-mail messages start with a mail user agent (MUA), a client system for sending and receiving e-mail such as mutt, Evolution, or Thunderbird. With the help of a Mail Submission Agent (MSA), such mail is normally sent to an MTA such as Postfix or sendmail. A Mail Delivery Agent (MDA) such as Procmail works locally to transfer e-mail from a server to an inbox folder. Procmail can also be used to filter e-mail. Red Hat also supports additional MTA services such as Dovecot to enable POP3 and/or IMAP (or the secure cousins, POP3s and IMAPs) to receive e-mail.

SMTP, the Simple Mail Transfer Protocol, has become one of the most important service protocols of the modern era. Much of the Internet-connected world lives and dies by e-mail and relies on SMTP to deliver it. Like POP3 and IMAP, SMTP is a *protocol*, a set of rules for transferring data used by various mail transfer agents.

Relevant Mail Server Packages

The packages associated with sendmail and Postfix are both part of the "E-mail server" package group. Key packages are listed in Table 13-2. You can install them with the **rpm** or **yum** command. Just remember that you do not need to install everything in this table.

When installed, the default E-mail server package group includes packages for the Postfix and Dovecot servers, along with the Spamassassin filter. Since you may not need all of these packages, it may be faster to install just Postfix or sendmail with the **rpm** or **yum** command, especially if you're configuring your Linux computer from the text console. It takes time to start the GUI.

TABLE 13-2

Mail Server Packages

RPM Package	Description
cyrus-imapd*	Installs the Cyrus IMAP enterprise e-mail system (several packages); may require perl-Cyrus.
cyrus-sasl	Adds the Cyrus implementation of the Simple Authentication and Security Layer (SASL).
dovecot	Supports both the IMAP and the POP incoming e-mail protocols.
dovecot-mysql, dovecot-pgsql, dovecot-pigeonhole	Includes database back ends and related plugins for Dovecot.
mailman	Supports e-mail discussion lists.
postfix	Includes an alternative to sendmail.
sendmail	Installs the most popular open-source mail server of the same name.
sendmail-cf	Adds a number of templates that you can use to generate your sendmail configuration file; required to process many sendmail configuration files.
spamassassin	Includes the spam fighting package of the same name.

exam watch

If you choose to work with sendmail, you should also install the sendmail-cf package to support the use of sendmail macro files.

Use alternatives to Select an E-Mail System

The **alternatives** command, with the **--config** switch, supports choices between different services such as Postfix and sendmail. Before using **alternatives**, you should stop the currently running SMTP service with the appropriate one of the following commands:

```
# /etc/init.d/postfix stop
# /etc/init.d/sendmail stop
```

Now run the following **alternatives** command, with the **--config** switch, to select the preferred MTA:

```
# alternatives --config mta
```

The command leads to the following output, which allows you to choose from installed SMTP e-mail servers. Other SMTP services, if installed, would be included in the list that follows:

```
There are 2 programs which provide 'mta'.

  Selection    Command
-----------------------------------------------
*+ 1           /usr/sbin/sendmail.sendmail
   2           /usr/sbin/sendmail.postfix

Enter to keep the current selection[+], or type selection number:
```

When making a selection, **alternatives** changes the appropriate runlevel scripts for each service, which can be confirmed with the following commands:

```
# chkconfig --list sendmail
# chkconfig --list postfix
```

In fact, the **chkconfig** command, when used to list the runlevels associated with an inactive service, may return an error message (with a proposed solution) similar to the following:

```
service sendmail supports chkconfig, but is not referenced in any runlevel (run
'chkconfig --add sendmail')
```

The **alternatives** command does not by itself stop or start a service. If you did not stop the original service earlier, the daemon will still be running. In that case, you'd have to use the **kill** command described in Chapter 9 to kill the undesired SMTP service. And it's important to have only one SMTP service running on a system. Interactions between sendmail and Postfix would lead to errors.

In addition, you'd have to use the appropriate script in the /etc/init.d directory (**sendmail** or **postfix**) to start the desired SMTP service.

General User Security

By default, all users are allowed to use locally configured SMTP services, without passwords. You'll see how this can be changed for both Postfix and sendmail in appropriate sections later in this chapter. This section assumes appropriate limitations have been configured.

In some cases, you may want to set up local users just so they have access to such services. If you don't want such users to log in to the server with regular accounts, one option is to make sure that such users don't have a login shell. For example, the following command can set up a user named tempworker on a local system without a login shell:

```
# useradd tempworker -s /sbin/nologin
```

That tempworker user can then set up his own e-mail manager such as Evolution, Thunderbird, or even Outlook Express to connect to networked Postfix or sendmail SMTP services. Any attempts by that user to log in directly to the server are rejected.

Of course, access is limited to configured users, whether or not their accounts are configured with a login shell. That's configured courtesy of the Simple Authentication and Security Layer (SASL). As implemented for RHEL 6, it's based on the cyrus-sasl package, as configured in the /etc/sasl2 directory. While that directory may include different configuration files for Postfix (smtpd.conf) and sendmail (Sendmail.conf, yes, that's an uppercase S), both configuration files refer back to the same authentication scheme with the following directive:

```
pwcheck_method:saslauthd
```

The /etc/sysconfig/saslauthd configuration file confirms the standard mechanism for password checks with the following directive:

```
MECH=pam
```

That's a reference to the Pluggable Authentication Modules (PAM) described in Chapter 10. In other words, users who are configured on the local system are controlled by associated files in the /etc/pam.d directory, namely smtp.postfix and smtp.sendmail. However, you'll need to make a few changes to Postfix to actually make it read the authentication database.

Mail Logging

Most log messages associated with SMTP services can be found in the /var/log/maillog file. Messages that you might expect to see in this file relate to

- Restarts of both sendmail and Postfix
- Successful and failed user connections
- Sent and rejected e-mail messages

Common Security Issues

By default, the SMTP service uses port 25. If you open port 25 on the firewall, outside users may have access to that server. If you need to know how to open that port, see Chapter 10. One option for **iptables**-based firewalls is based on source IP addresses. As both Postfix and sendmail are SMTP services, both use port 25.

To create a source address option with the Firewall Configuration tool, you'll need to use the Custom Rules option. As shown in Figure 13-1, you can see custom files added to the firewall configuration, from the /usr/share/netcf directory.

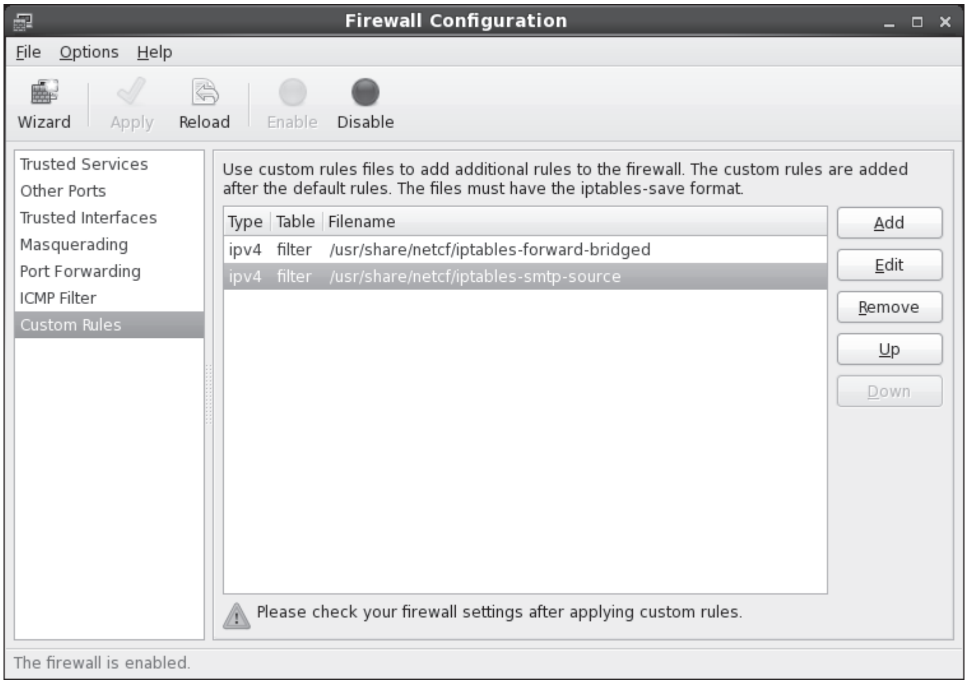
To create a custom rule that supports access only from systems on the 192.168.122.0 network, I've included the following entry in that file:

```
-A INPUT -m state --state NEW -m tcp -p tcp -s 192.168.122.0/24 --dport 25 -j
ACCEPT
```

The entry is in the same format as the commands in the /etc/sysconfig/iptables file. The **iptables** service, when started, reads the contents of the /etc/sysconfig/iptables file along with files cited in the Custom Rules section. Just remember, when you add a file as a custom rule, to make sure the Firewall Table option shown in Figure 13-2 refers to a filter, consistent with standard **iptables**-based firewall rules.

FIGURE 13-1

Custom Rules



Use custom rules files to add additional rules to the firewall. The custom rules are added after the default rules. The files must have the iptables-save format.

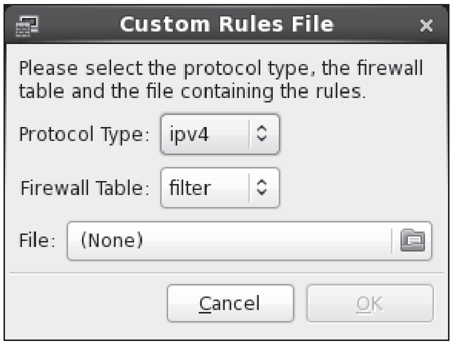
Type	Table	Filename
ipv4	filter	/usr/share/netcf/iptables-forward-bridged
ipv4	filter	/usr/share/netcf/iptables-smtp-source

Please check your firewall settings after applying custom rules.

The firewall is enabled.

FIGURE 13-2

Custom Rules file



Please select the protocol type, the firewall table and the file containing the rules.

Protocol Type: ipv4

Firewall Table: filter

File: (None)

Alternatively, you could edit the /etc/sysconfig/iptables file directly. But any future use of the Firewall Configuration tool would overwrite such custom rules.

In general, SELinux is not an issue for SMTP services. Only one SELinux boolean applies to the Postfix service, allow_postfix_local_write_mail_spool. It's active by default. As suggested by the name, it allows the Postfix service to write e-mail files to user spools in the /var/spool/postfix directory.

Testing an E-Mail Server

Besides the **telnet** command described later in this chapter, the appropriate way to test an e-mail server is with an e-mail client. Of course, it would be convenient to have a GUI e-mail client available. But as discussed in Chapter 2, only text clients like **mutt** may be available.

EXERCISE 13-1

Create Users Just for E-Mail

In this exercise, you will create three users on the local system, just so they can access the local SMTP server. It is understood that additional configuration is required to set up access or limits for these users on the Postfix or sendmail SMTP services. The users are mailer1, mailer2, and mailer3.

1. Review the **useradd** command. Identify the switch associated with the default login shell.

2. Review the contents of the /etc/passwd file. Find a shell not associated with logins. It should be

   ```
   /sbin/nologin
   ```

3. Run commands like **useradd mailer1 -s /sbin/nologin** to add a new user. Make sure to assign that user a password.
4. Review the result in /etc/passwd.
5. Repeat Step 3 for the other noted users.
6. Try logging in to one of the new accounts as a regular user. It should fail. Review associated messages in the /var/log/secure file.
7. Keep the new users.

CERTIFICATION OBJECTIVE 13.02

The Configuration of Postfix

The Postfix mail server is one way to manage the flow of e-mail on a system and for a network. Standard configuration files are stored in the /etc/postfix directory. The **postconf** command can be used to test the configuration. As installed, Postfix accepts e-mail from only the local system. The configuration changes required to set up Postfix to accept incoming e-mail, and to forward e-mail through a smart host are relatively simple.

The details of Postfix configuration files include options for user- and host-based security. If you already know how to configure Postfix for basic operation and just want to know what's required to meet the SMTP objectives for Postfix, jump ahead to the sections associated with /etc/postfix/access, accepting incoming e-mail and smart hosts.

Configuration Files

The configuration files are stored in the /etc/postfix directory. The main configuration file, main.cf, is somewhat simpler than the sendmail alternative, sendmail.cf. It's still complex, as it includes nearly 700 lines.

Except for the .cf files, any changes must first be processed into a database with the **postmap** command. For example, if you've added limits to the access file, it can be processed into a binary access.db file with the following command:

```
# postmap access
```

In many cases, the contents of files in the /etc/postfix directory is a commented version of the associated man page. The following sections do not cover the main.cf or the master.cf files, as those are covered later. It also does not cover the header_checks file, as that's more of a message filter.

After any changes are made to Postfix configuration files, it's normally best to reload them into the daemon with the following command:

```
# /etc/init.d/postfix reload
```

It's best to reload most services, as that avoids kicking off currently connected users. And that can avoid users who complain about lost e-mails. But watch the output. It should include the following:

```
Reloading postfix:    [ OK ]
```

Without that output, there may be a different kind of problem with Postfix. Sometimes that problem can be addressed by restarting the service with the following command:

```
# /etc/init.d/postfix restart
```

The Postfix access File

The access file may be configured with limits on users, hosts, and more. It includes a commented copy of the associated man page, which can also be called with the **man 5 access** command. When limits are included in that file, they're configured in the following pattern:

```
pattern action
```

Patterns can be set up in a number of ways. As suggested by the man page, you can limit users with patterns such as

```
username@example.com
```

Patterns can be configured with individual IP addresses, IP address networks, and domains, such as with the following examples. Pay attention to the syntax, specifically the lack of a dot at the end of the 192.168.100 and the beginning of the example.org expressions. These expressions still are inclusive of all systems on the 192.168.100.0/24 network and the *.example.org domain.

```
192.168.122.50
server1.example.com
192.168.100
example.org
```

Of course, such patterns have no meaning without an action. Typical actions include **REJECT** and **OK**. The following examples of active lines in the /etc/postfix/access file follow the pattern action format:

```
192.168.122.50 OK
server1.example.com OK
192.168.100 REJECT
example.org REJECT
```

exam watch

One way to configure host- and user-based security for Postfix is through the access file in the /etc/postfix directory. Another way to configure host-based security is with iptables command–based firewalls described in Chapter 10. While there are more complex methods to configure user-based security, the RHCE objectives suggest that you "configure the service for basic operation."

The Postfix canonical and generic Files

The files named canonical and generic in the /etc/postfix directory works like an alias file. In other words, when users move from place to place, or if a company moves from one domain to another, the canonical file can ease that transition. The difference is while the canonical file applies to incoming e-mail from other systems, the generic file applies to e-mail being sent to other systems.

Similar to the access file, options in these files follow a pattern:

```
pattern result
```

The simplest iteration is the following, which forwards e-mail sent to a local user to a regular e-mail address:

```
michael michael@example.com
```

For companies that use different domains, the following line would forward e-mail directed to michael@example.org to michael@example.com. It would forward other example.org e-mail addresses in a similar fashion.

```
@example.org @example.com
```

Don't forget to process the resulting files into databases with the **postmap canonical** and **postmap generic** commands. If you modify the relocated, transport, or the virtual files in the /etc/postfix directory, apply the **postmap** command to those files as well.

The Postfix relocated File

The /etc/postfix/relocated file is designed to contain information for users who are now on external networks, such as users who have left a current organization. The format is similar to the aforementioned canonical and generic files in the same directory. For example, the following entry might reflect forwarding from a local corporate network to a personal e-mail address:

```
john.doe@example.com  john.doe@example.net
```

The Postfix transport File

The /etc/postfix/transport file may be useful in some situations where mail is forwarded, such as from a smart host. For example, the following entry forwards e-mail directed to the example.com network to an SMTP server such as Postfix on the server1.example.com system:

```
example.com  smtp:server1.example.com
```

The Postfix virtual File

The /etc/postfix/virtual file can forward e-mail addressed in a normal fashion, such as to elizabeth@example.com, to the user account on a local system. For example, if user elizabeth is actually the administrator on a system, the following entry forwards mail sent to the noted e-mail address to the root administrative user:

```
elizabeth@example.com root
```

The main.cf Configuration File

Back up this file and open it in a text editor. There are several things that you should configure in this file to get it working. Properly configured, the changes should limit access to the local system and network. This section also describes the function of other active directives, based on the default version of the file.

First, Postfix queues, which include e-mail that has yet to be sent, or e-mail that has been received, can be found in the queue_directory:

```
queue_directory = /var/spool/postfix
```

The following directory is a standard. It describes the location of most Postfix commands.

```
command_directory = /usr/sbin
```

Postfix includes a substantial number of executable files, for configuration in the master.cf file. The daemon_directory directive specifies their location:

```
daemon_directory = /usr/libexec/postfix
```

Postfix includes writable data files in the following directory; it normally includes a master.lock file with the PID of the Postfix daemon:

```
data_directory = /var/lib/postfix
```

As defined in the comments of the main.cf file, some files and directories should be owned by the root administrative user; others should be owned by the specified mail_owner. In the /etc/groups file, you can confirm that there's a dedicated group named postfix, which is also part of the group named mail.

```
mail_owner = postfix
```

While Postfix works for the local system "out of the box," more has to be done to get it working for a network. To that end, you'll need to activate and modify the following **myhostname** directive to point to the name of the local system. For example, you might change the entry

```
#myhostname = host.domain.tld
```

to an alternative like

```
myhostname = server1.example.com
```

on the job

An authoritative DNS server may be configured to point to an SMTP e-mail server in its database.

While an SMTP server is located on a specific system, normally such SMTP servers are configured for an entire network. That's configured with the **mydomain** directive. To that end, you should change the following comment:

```
#mydomain = domain.tld
```

to reflect the domain name or IP network address of the local network:

```
mydomain = example.com
```

Normally, you'd just uncomment the following **myorigin** directive, to label e-mail addresses coming from this Postfix server with an origination domain. In this case, the origination domain is example.com:

```
myorigin = $mydomain
```

By default, the following active directive limits the scope of the Postfix service to the local system.

```
#inet_interfaces = all
inet_interfaces = localhost
```

In most cases, you'd change the active directive so that Postfix listens on all active network cards:

```
inet_interfaces = all
#inet_interfaces = localhost
```

Normally, Postfix listens on both IPv4 and IPv6 networks, based on the following inet_protocols directive:

```
inet_protocols = all
```

The mydestination directive specifies the systems served by this Postfix server. Based on the previous settings, the following default directive means that accepted mail may be sent to the local system's FQDN (server1.example.com), the localhost address on the example.com network, and the localhost system:

```
mydestination = $myhostname, localhost.$mydomain, localhost
```

For a Postfix server configured for the local network, you should add the name of the local domain, already assigned to the mydomain directive:

```
mydestination = $mydomain, $myhostname, localhost.$mydomain, localhost
```

In addition, you'll want to set up the **mynetworks** directive to point to the IP network address to be covered by this Postfix server. The default commented directive does not point to the example.com network defined for this book:

```
#mynetworks = 168.100.189.0/28, 127.0.0.0/8
```

So for systems like server1.example.com, this directive should be changed to

```
mynetworks = 192.168.122.0/24, 127.0.0.0/8
```

exam watch

Host-based security limits in Postfix are configured through the mynetworks directive in the /etc/postfix/main.cf file.

Once changes are made to the main.cf file (and any other files in the /etc/postfix directory) are complete and saved, you may want to review current Postfix parameters. To do so, run the following command:

```
# postconf
```

Of course, most of these parameters are defaults. To review the parameters changed by the main.cf file, run the following command:

```
# postconf -n
```

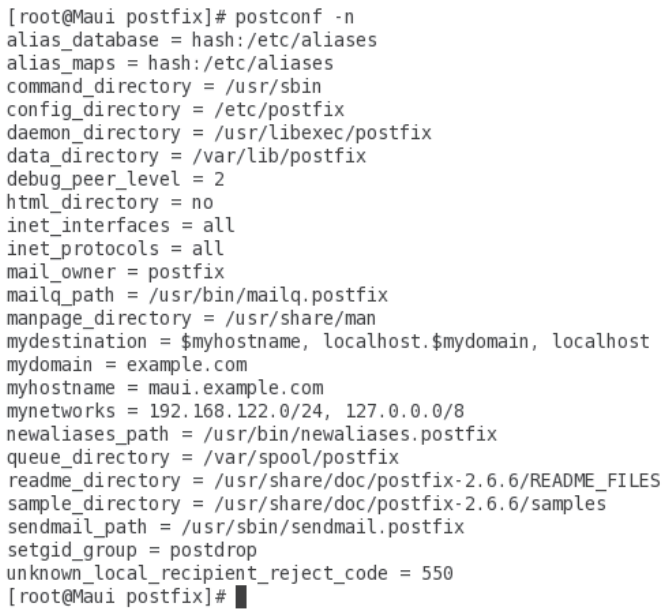
The output is shown in Figure 13-3.

One setting from the **postconf -n** output is important to authentication. Specifically, when the following directive is added to the main.cf file, Postfix will require authorized usernames and passwords for access:

```
smtpd_sender_restrictions = permit_sasl_authenticated, reject
```

In addition, Postfix includes a syntax checker in the basic daemon. Run the following command to see if there are any fatal errors in the main.cf file:

```
# postfix check
```



```
[root@Maui postfix]# postconf -n
alias_database = hash:/etc/aliases
alias_maps = hash:/etc/aliases
command_directory = /usr/sbin
config_directory = /etc/postfix
daemon_directory = /usr/libexec/postfix
data_directory = /var/lib/postfix
debug_peer_level = 2
html_directory = no
inet_interfaces = all
inet_protocols = all
mail_owner = postfix
mailq_path = /usr/bin/mailq.postfix
manpage_directory = /usr/share/man
mydestination = $myhostname, localhost.$mydomain, localhost
mydomain = example.com
myhostname = maui.example.com
mynetworks = 192.168.122.0/24, 127.0.0.0/8
newaliases_path = /usr/bin/newaliases.postfix
queue_directory = /var/spool/postfix
readme_directory = /usr/share/doc/postfix-2.6.6/README_FILES
sample_directory = /usr/share/doc/postfix-2.6.6/samples
sendmail_path = /usr/sbin/sendmail.postfix
setgid_group = postdrop
unknown_local_recipient_reject_code = 550
[root@Maui postfix]# 
```

FIGURE 13-3 Custom Postfix settings, based on /etc/postfix/main.cf

The /etc/aliases Configuration File

Another directive from the /etc/postfix/main.cf file includes the database hash from the /etc/aliases file, which is processed into the /etc/aliases.db file when the Postfix system is restarted.

```
alias_maps = hash:/etc/aliases
```

The /etc/aliases file is normally configured to redirect e-mail sent to system accounts to the root administrative user. As you might see at the end of that file, e-mail messages sent to root can be redirected to a regular user account:

```
# root    marc
```

While there are a number of additional directives available in this file, they're beyond the basic configuration associated with the RHCE objectives. When changes are complete, you can and should process this into an appropriate database with the **newaliases** command. As the /etc/aliases file works for both Postfix and sendmail, the **newaliases** command can process the /etc/aliases file for both MTAs.

The master.cf Configuration File

Generally, you should not have to make changes to the master.cf file. It's configured to set up Postfix for regular SMTP services. As shown in the first page of the file, it does include options for the submission protocol on port 587, which is required for a smart host relay to some ISP's e-mail servers. It also supports the configuration of secure SMTP.

Test the Current Postfix Configuration

As noted in previous chapters, the **telnet** command is an excellent way to review the current status of a service on a local system. Based on the default configuration of Postfix, an active version of this service should be listening on port 25. In that case, a **telnet localhost 25** command should return messages similar to the following:

```
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 Maui.example.com ESMTP Postfix
```

If IPv6 networking is enabled on the local system, the IPv4 loopback address (127.0.0.1) would be replaced by the regular IPv6 loopback address (::1). The **quit** command can be used to exit from this connection. But don't quit yet. The **EHLO localhost** command is important; the EHLO is the enhanced HELO command, which introduces the basic parameters of an SMTP server.

```
EHLO localhost
250-maui.example.com
250-PIPELINING
250-SIZE 10240000
250-VRFY
250-ETRN
250-ENHANCEDSTATUSCODES
250-8BITMIME
250 DSN
```

For our purposes, the most important information is what's missing. No authentication is required on this server. When authentication is properly configured on Postfix, you'll also see the following line in the output:

```
250-AUTH GSSAPI
```

Configure Postfix Authentication

When authentication is configured in Postfix, user limits can apply. But as there are no hints in the standard main.cf configuration file, you'll have to refer to Postfix documentation for clues. As suggested in Chapter 3, most packages include some level of documentation in the /usr/share/doc directory. Fortunately, Postfix documentation in that directory is rather extensive. For RHEL 6, you'll be able to find that documentation in the postfix-2.6.6/ subdirectory.

The directives that you need to add to the main.cf file to set up authentication are shown in the README-Postfix-SASL-RedHat.txt file in that directory. The key excerpt is shown in Figure 13-4.

For the first step listed, it's sufficient to copy the four directives listed to the end of the main.cf file, first to enable SASL authentication for Postfix connections:

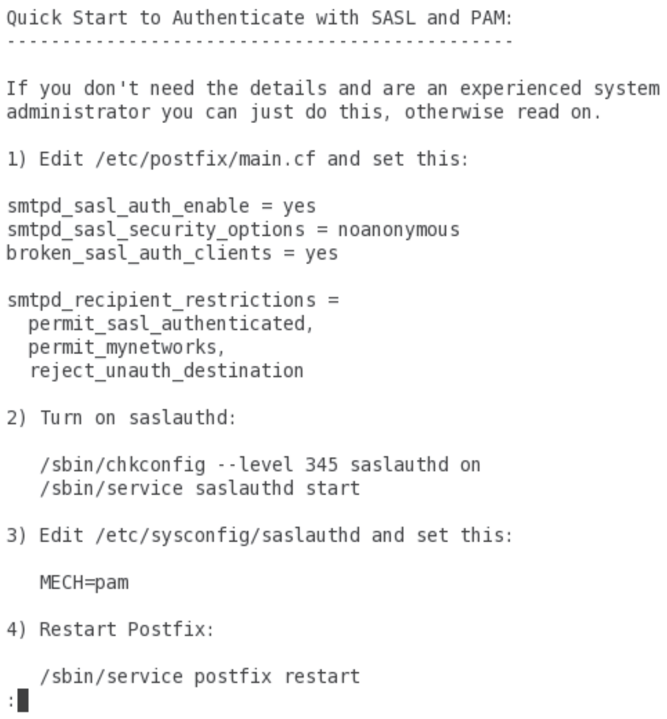
```
smtpd_sasl_auth_enable = yes
```

Next, this disables anonymous authentication:

```
smtpd_sasl_security_options = noanonymous
```

FIGURE 13-4

Directions to set up Postfix authentication



```
Quick Start to Authenticate with SASL and PAM:
----------------------------------------------

If you don't need the details and are an experienced system
administrator you can just do this, otherwise read on.

1) Edit /etc/postfix/main.cf and set this:

smtpd_sasl_auth_enable = yes
smtpd_sasl_security_options = noanonymous
broken_sasl_auth_clients = yes

smtpd_recipient_restrictions =
  permit_sasl_authenticated,
  permit_mynetworks,
  reject_unauth_destination

2) Turn on saslauthd:

   /sbin/chkconfig --level 345 saslauthd on
   /sbin/service saslauthd start

3) Edit /etc/sysconfig/saslauthd and set this:

   MECH=pam

4) Restart Postfix:

   /sbin/service postfix restart
:
```

The directive that follows allows authentication from nonstandard clients such as Microsoft Outlook Express:

```
broken_sasl_auth_clients = yes
```

This allows authenticated users, allows access from networks configured with the **mynetworks** directive, and rejects destinations other than the Postfix server:

```
smtpd_recipient_restrictions = permit_sasl_authenticated,
  permit_mynetworks, reject_unauth_destination
```

Configure Incoming E-Mail

The directives required to set up Postfix to accept incoming e-mail from other system have been previously described in the description of the main.cf file. But that was a more comprehensive description of that file. This section just covers the minimum requirements to configure Postfix, in the words of the RHCE objectives, "to accept inbound e-mail from other systems." Given a Postfix server configured on the server1.example.com system, on the 192.168.122.0/24 network, you'd make the following changes to the main.cf file in the /etc/postfix directory:

```
myhostname = server1.example.com
mydomain = example.com
myorigin = mydomain
inet_interfaces = all
mynetworks = 192.168.122.0/24, 127.0.0.0/8
```

Each of these options replaces either a comment or an active directive in the default /etc/postfix/main.cf file. For example, you should at least comment out the following directive:

```
#inet_interfaces = localhost
```

Configure a Relay Through a Smart Host

A smart host has all of the functionality of a regular SMTP server, except for the forwarding of all e-mail through a second SMTP server. The location of the smart host can be specified with the **relayhost** directive. For example, if the remote smart host is outsider1.example.org, you'd add the following directive to the /etc/postfix/main.cf file:

```
relayhost = outsider1.example.org
```

For smart hosts to work, you'll need to make sure that e-mail messages intended for users on the local system are properly forwarded. And that's possible through the aforementioned /etc/aliases file. At the least, you should configure e-mail intended for the root administrative user as forwarded to a regular local user, with a line such as

```
root   michael
```

EXERCISE 13-2

Switch Services

This exercise presumes you've installed and want to test the sendmail SMTP service. If you're set on Postfix, there is no need to run this exercise. It assumes that both the sendmail and sendmail-cf packages are installed.

1. Deactivate the Postfix service with the **/etc/init.d/postfix stop** command.
2. Run the **alternatives --config mta** command. From the menu that appears, select the sendmail SMTP service.
3. Start the sendmail service with the **/etc/init.d/sendmail start** command.
4. Review currently running SMTP processes with the **ps aux | grep postfix** and the **ps aux | grep sendmail** commands.
5. Stop the sendmail service with the **/etc/init.d/sendmail stop** command.
6. Run the **alternatives --config mta** command. From the menu that appears, select the Postfix SMTP service.
7. Restart the Postfix service with the **/etc/init.d/postfix start** command.

CERTIFICATION OBJECTIVE 13.03

The Other SMTP Service: sendmail

The sendmail e-mail server may still be the most popular SMTP service on the Internet. It is the older Red Hat way to manage the flow of e-mail on a system and for a network. It was the default SMTP service through RHEL 5. Standard configuration files are stored in the /etc/mail directory. As sendmail is quite complex, configuration is normally done with the help of macros.

As with Postfix, the sendmail software as installed accepts e-mail only from the local system. Based on an understanding of associated macro files, the configuration changes required to modify sendmail to accept incoming e-mail from a network and to forward e-mail through a smart host are relatively simple.

If you already know sendmail, and just want to know what's required to meet the RHCE objectives for that SMTP service, jump ahead to the sections associated with the /etc/mail/access, file, accepting incoming e-mail, and smart hosts.

The Basics of sendmail

When sendmail starts, it reads the /etc/mail/sendmail.cf and /etc/mail/submit.cf files. The sendmail.cf file is a long (around 1800 lines) file that may seem difficult to decipher but includes a wealth of helpful comments. The submit.cf file is nearly as long. This file provides detailed rules (organized into rulesets) on how sendmail should process e-mail addresses, filter spam, talk to other mail servers, and more.

This file is extremely complex and uses cryptic syntax. Fortunately, most of the directives included in this file are standards that you don't need to change. Many are required by various Internet agreements relating to e-mail address, mail transfer agents, and so on.

Red Hat simplifies this process with a smaller file, /etc/mail/sendmail.mc, which contains only the most relevant configuration directives. It is composed entirely of macros that define key sendmail.cf settings. Once appropriate changes are made, you can run the **make** command to compile a new, custom sendmail.cf file. However, the default RHEL version of this file is still around 200 lines long. In most cases, you might have to change two or three of those lines.

Of course, once files are created or revised and compiled in the /etc/mail directory, you'll want to make the sendmail service reread these configuration files with the following command:

```
# /etc/init.d/sendmail reload
```

If successful, you'll see the following output (the second "reloading" is in lowercase):

```
Reloading sendmail:  [ OK ]
reloading sm-client: [ OK ]
```

Without this output, you should assume that sendmail did not re-read the configuration files, and other measures are required, such as a restart.

Configuration Files

The following is a brief description of the standard configuration files in the /etc/mail directory. Additional files with a .db extension are database files processed from some of these listed here:

- **access** Supports access control. The default version of this file supports access from the local computer. You can add hostnames or networks to this list, with a message to **REJECT** with an error message, **DISCARD** without an error message, or **RELAY** to accept and send the e-mail. It is one way to configure host- and even user-based security.
- **aliasesdb-stamp** Supports date checks of existing database files.
- **domaintable** Allows mapping different domains. For example, if a company is moving its users from mheducation.com to mcgraw-hill.com, people might still send e-mails to addresses such as michael@mheducation.com. The following line would forward that e-mail to michael@mcgraw-hill.com.

  ```
  mheducation.com   mcgraw-hill.com
  ```

- **helpfile** Supports help commands from the sendmail prompt, accessible with the **telnet localhost 25** command.
- **local-host-names** Allows added hostnames or aliases for a sendmail server. Enter one alias per line.
- **mailertable** Information added to this file may be used as a substitute for DNS searches.
- **makefile** Supports compiling the sendmail.mc file.
- **sendmail.cf** Specifies the main sendmail configuration file.
- **sendmail.mc** Name of a macro file that can be used to generate a new sendmail.cf file.
- **spamassassin/** A directory that includes configuration files to help minimize spam. The following line in /etc/procmailrc helps it work with Procmail for locally received e-mail:

  ```
  INCLUDERC=/etc/mail/spamassassin/spamassassin-default.rc
  ```

on the job

If you forget how spamassassin is used with sendmail, run the* rpm -qi spamassassin *command. You'll see it in the description.

- **statistic** Collects statistics on sendmail usage in binary format. You can read it with the **mailstats** command. Does not exist until the sendmail service starts processing mail.
- **submit.cf** The main outgoing sendmail configuration file.
- **submit.mc** A macro that you can edit and then generate a new submit.cf file.
- **trusted-users** Lists special users that can send e-mail without warnings.
- **virtusertable** Supports e-mail forwarding; if some users from outside local networks use the local sendmail server, this file supports e-mail forwarding from those domains.

Some of these files require the other sendmail package, sendmail-cf. Use the **rpm -qa | grep sendmail** command to confirm whether these packages are installed.

Many of these files are cited in the sendmail.mc file. For example, the following directive incorporates /etc/mail/virtusertable in the default sendmail.mc configuration file:

```
FEATURE(`virtusertable',`hash -o /etc/mail/virtusertable.db')dnl
```

You may notice several versions of these files with .db extensions. These are the database files used by sendmail. When you edit files in the /etc/mail directory, the **/etc/mail/make** command, described shortly, processes these files into the .db databases.

There's one more important file, /etc/aliases, already described earlier in this chapter. It has the same functionality for both Postfix and sendmail. In other words, it includes a list of forwarders on a local system, from system addresses to the root account, or from one regular user account to another. The **newalises** command processes this file for sendmail as well.

The sendmail.mc Macro File

Even the main sendmail macro file, sendmail.mc, can seem intimidating. But very few changes are required to actually get sendmail working. Nevertheless, in case you encounter a slightly different question on an exam, or a somewhat nonstandard question on the job, it's important to understand the contents of the sendmail.mc file.

Macro files for sendmail start with the following **divert** directive:

```
divert(-1)dnl
```

Each line in the file either starts or ends with the **dnl**, which is the functional equivalent of the comment character. All information after the **dnl** is ignored by the sendmail macro processor. If coupled with a **divert(0)dnl**, all information between the two **divert** directives is ignored.

The comments that follow provide important directions; the **make** command in the /etc/mail directory in fact processes all files in the /etc/mail directory:

```
dnl # This is the sendmail macro config file for m4. If you make changes to
dnl # /etc/mail/sendmail.mc, you will need to regenerate the
dnl # /etc/mail/sendmail.cf file by confirming that the sendmail-cf
dnl # package is installed and then performing a
dnl #
dnl #     /etc/mail/make
```

The command that follows includes the noted c4.m4 macro processor, from the sendmail-cf package:

```
include(`/usr/share/sendmail-cf/m4/cf.m4')dnl
```

The **include** directive instructs the **make** command to read the contents of the named file and insert it at the current location in the output. The quotes in the sendmail.mc file do not conform to standard English usage. This is how additional standard configuration information is left out of the main sendmail.mc macro file.

Incidentally, quoted parameters start with a back quote mark (`) and end with a single quote mark (‘).

The **VERSIONID** that follows provides a label for the current configuration:

```
VERSIONID(`setup for linux')dnl
```

The **OSTYPE** directive that follows specifies the configured operating system:

```
OSTYPE(`linux')dnl
```

The **define** directive sets files or enables possibly desirable features. Some examples in sendmail.mc support a list of e-mail aliases in the **ALIAS_FILE** (/etc/aliases), identify where procmail lives (**PROCMAIL_MAILER_PATH**), and set basic authentication options (**confAUTH_OPTIONS**).

One useful option is to avoid advertising the version of sendmail in use, which would otherwise be shown in e-mail message headers. If you activate the following **define** feature, others don't have to know that you've configured sendmail version 8.14.4.

```
dnl define(`confSMTP_LOGIN_MSG', `$j Sendmail; $b')dnl
```

The commented **define** directive for a SMART_HOST that follows would be the simplest way to set up forwarding to a smart host, as discussed later in this chapter.

```
dnl define(`SMART_HOST', `smtp.your.provider')dnl
```

Most of the active **define** options that follow relate to the performance of the sendmail service. The exceptions, shown next, support the use of Procmail for filtering, enable /etc/aliases for substitute e-mail addresses, and require authentication to receive e-mail:

```
define(`PROCMAIL_MAILER_PATH', `/usr/bin/procmail')dnl
define(`ALIAS_FILE', `/etc/aliases')dnl
define(`confAUTH_OPTIONS', 'A')dnl
```

The problem with the last of these **define** directives is that it allows plain text authentication. Like the Telnet service, it allows people to send their usernames and passwords over the network in clear text, where anyone who is listening can read that authentication information. Clear text may be appropriate for initial tests and generally conforms to the "basic operation" concept associated with the RHCE objectives. However, for a sendmail service that requires some form of encryption, you could substitute the following directive, which is commented out by default:

```
dnl define(`confAUTH_OPTIONS', `A p')dnl
```

That authentication mechanism, if active, should be coupled with authentication options. The commented directives that follow provide some suggested solutions.

While there's no evidence from the RHCE objectives of an authentication requirement for SMTP services, it is listed in the public Red Hat RH254 course outline, which suggests that you do need to understand the use of Secure Sockets Layer (SSL) certificates, and its successor, Transport Layer Security (TLS). While it's likely unnecessary for SMTP services, it may be helpful to become familiar with some of these authentication mechanisms, as a preview of Chapter 14. To that end, the following commented directives support a variety of authentication mechanisms:

```
dnl TRUST_AUTH_MECH(`EXTERNAL DIGEST-MD5 CRAM-MD5 LOGIN PLAIN')dnl
dnl define(`confAUTH_MECHANISMS', `EXTERNAL GSSAPI DIGEST-MD5 CRAM-MD5 LOGIN
PLAIN')dnl
```

To support encryption, the following commented section describes the commands required to create a SSL/TLS certificate for sendmail, and its location in the directory tree:

```
dnl # Rudimentary information on creating certificates for sendmail TLS:
dnl #     cd /etc/pki/tls/certs; make sendmail.pem
dnl # Complete usage:
dnl #     make -C /etc/pki/tls/certs usage
dnl #
dnl define(`confCACERT_PATH', `/etc/pki/tls/certs')dnl
dnl define(`confCACERT', `/etc/pki/tls/certs/ca-bundle.crt')dnl
dnl define(`confSERVER_CERT', `/etc/pki/tls/certs/sendmail.pem')dnl
dnl define(`confSERVER_KEY', `/etc/pki/tls/certs/sendmail.pem')dnl
```

The **define** directive that follows, if activated, would support the use of LDAP authentication:

```
dnl define(`confDONT_BLAME_SENDMAIL', `groupreadablekeyfile')dnl
```

The **define** directives that follow relate to sendmail behavior for message delays and do not affect basic configuration. The only active directive shown disables timeouts when the server waits for an identification (IDENT) query.

```
dnl define(`confTO_QUEUEWARN', `4h')dnl
dnl define(`confTO_QUEUERETURN', `5d')dnl
dnl define(`confQUEUE_LA', `12')dnl
dnl define(`confREFUSE_LA', `18')dnl
define(`confTO_IDENT', `0')dnl
dnl FEATURE(delay_checks)dnl
```

The **FEATURE** directives enable specific features. Some administrators configure sendmail to use the submission protocol, in place of SMTP. In that case, you'd want to disable the following directive, which keeps sendmail from listening on port 587:

```
FEATURE(`no_default_msa', 'dnl')dnl
```

Other **FEATURE** directives relate to specific executable and configuration files. The following specifies the locations of the sendmail shell (smrsh), along with the aforementioned mailertable and virtualusertable databases.

```
FEATURE(`smrsh', `/usr/sbin/smrsh')dnl
FEATURE(`mailertable`hash -o /etc/mail/mailertable.db')dnl
FEATURE(`virtusertable', `hash -o /etc/mail/virtusertable.db')dnl
```

The **FEATURE** directives that follow support redirection to other e-mail addresses, add the local domain name to an e-mail address if one is not listed, and use files like trusted-users and local-host-names:

```
FEATURE(redirect)dnl
FEATURE(always_add_domain)dnl
FEATURE(use_cw_file)dnl
FEATURE(use_ct_file)dnl
```

While this feature supports repeated attempts for e-mail delivery:

```
FEATURE(local_procmail, `', 'procmail -t -Y -a $h -d $u')dnl
```

The following options are prerequisites for host-based security, as they look to the /etc/mail/access file for (allowed and) blacklisted users, systems, and even networks:

```
FEATURE(`access_db', `hash -T<TMPF> -o /etc/mail/access.db')dnl
FEATURE(`blacklist_recipients')dnl
```

If the root user tries to send e-mail through this SMTP server, the following option requires that user's full e-mail address:

```
EXPOSED_USER(`root')dnl
```

By default, the following **DAEMON_OPTIONS** directive limits the sendmail service to the local system. For basic operation, it's simplest to comment out this directive:

```
DAEMON_OPTIONS(`Port=smtp,Addr=127.0.0.1, Name=MTA')dnl
```

No additional directives are required in the sendmail.mc file to support access from other systems on a network. However, security also depends on the /etc/mail/access file, which also by default limits access to the localhost system.

To configure sendmail on the aforementioned submission port 587, you'd activate the following directive:

```
dnl DAEMON_OPTIONS(`Port=submission, Name=MSA, M=Ea')dnl
```

If you've compiled the previously described TLS certificates, you can activate the following directive to listen for secure SMTP connections on port 465:

```
dnl DAEMON_OPTIONS(`Port=smtps, Name=TLSMTA, M=s')dnl
```

For IPv6 networking, the following directive would listen to only the localhost system:

```
dnl DAEMON_OPTIONS(`port=smtp,Addr=::1, Name=MTA-v6, Family=inet6')dnl
```

Alternatively, the following directive listens for both IPv4 and IPv6 traffic:

```
dnl DAEMON_OPTIONS(`Name=MTA-v4, Family=inet, Name=MTA-v6, Family=inet6')
```

The following **FEATURE** directive allows sendmail to **accept_unresolvable_domains**. This allows sendmail to accept mail even if it can't figure out the domain of the user who sent the e-mail. Specifically, a domain is regarded as unresolvable when a reverse IP address search does not find the associated domain name. However, if reliable DNS service is available, deactivating this option can reduce spam.

```
FEATURE(`accept_unresolvable_domains')dnl
```

If active, the following directive accepts the use of MX records from DNS servers for the locations of remote e-mail servers:

```
dnl FEATURE(`relay_based_on_MX')dnl
```

The following directive is needed to make sure the sendmail service accepts e-mail from local users:

```
LOCAL_DOMAIN(`localhost.localdomain')dnl
```

If you want to substitute a different domain for e-mail addresses, the following directives, if active, specify a substitute:

```
dnl MASQUERADE_AS(`mydomain.com')dnl
dnl FEATURE(masquerade_envelope)dnl
dnl FEATURE(masquerade_entire_domain)dnl
```

The masquerading can be extended. The following directives, if active, would get sendmail to substitute the **MASQUERADE_AS** domain for the localhost, localhost.localdomain, mydomainalias.com, and mydomain.lan domain names.

```
dnl MASQUERADE_DOMAIN(localhost)dnl
dnl MASQUERADE_DOMAIN(localhost.localdomain)dnl
dnl MASQUERADE_DOMAIN(mydomainalias.com)dnl
dnl MASQUERADE_DOMAIN(mydomain.lan)dnl
```

Finally, the following **MAILER** directives specifies the servers in use:

```
MAILER(smtp)dnl
MAILER(procmail)dnl
dnl MAILER(cyrusv2)dnl
```

The submit.mc Macro File

In most cases, no changes are needed to the submit.mc file. If changed, it can also be processed by the same **make** command in the /etc/mail directory. In general, if the network is configured to use the Network Information Service (NIS) for an authentication database, you would comment out the following directive:

```
define(`confDONT_INIT_GROUPS', `True')dnl
```

But NIS is not secure. It just so happens that for RHEL 6, Red Hat has removed NIS from its exam objectives.

If the local network is set up to use only IPv6 addressing, you would change the last line in this file from

```
FEATURE(`msp', `[127.0.0.1]')dnl
```

to

```
FEATURE(`msp', `[IPv6:::1]')dnl
```

Configure sendmail to Accept E-Mail from Other Systems

This section satisfies the RHCE objective to set up sendmail for basic operation. Navigate to the /etc/mail directory. Back up the sendmail.mc macro file. Open that file in a text editor. Review the following directive, which limits sendmail access to the local computer:

```
DAEMON_OPTIONS(`Port=smtp,Addr=127.0.0.1, Name=MTA')dnl
```

You can allow other computers to use your sendmail server by commenting out this line. As described earlier, this requires a **dnl** directive in front, as shown:

```
dnl DAEMON_OPTIONS(`Port=smtp,Addr=127.0.0.1, Name=MTA')dnl
```

Next, if reliable DNS access is available, comment out the **FEATURE** directive that allows the sendmail service to **accept_unresolvable_domains**. This action, which requires verification of sender e-mail addresses, can help block spammers. A spammer may fake his domain. Users from “unresolvable domains” aren’t allowed access to this service—unless the **accept_unresolvable_domains** option is active—who use just an IP address, or spammers who fake their domain name to hide themselves:

```
dnl FEATURE(`accept_unresolvable_domains')dnl
```

But that’s not enough. To allow remote system access to the local sendmail server, you’ll need to add their names or IP addresses to the /etc/mail/access file. For example, to allow access to the 192.168.122.0 domain, you’d add the following line to that file:

```
Connect:192.168.122             RELAY
```

Watch the notation; unlike with other services, there is no dot (.) at the end of the address. It covers all computers on the 192.168.122.0 network. Alternatively, you could designate the example.com domain or a specific computer name or IP address.

Back up the current sendmail.cf file. Then you can generate a new sendmail.cf file, process the other files in /etc/mail, and restart sendmail services with the following commands:

```
# /etc/mail/make
# /etc/init.d/sendmail restart
```

Now you can reconfigure e-mail clients such as Mozilla Thunderbird, Novell Evolution, or even Microsoft Outlook Express to send outgoing e-mail through the newly configured sendmail server. You'll need to set the sendmail computer domain name or IP address as the SMTP outgoing mail server.

Configure sendmail to Relay E-Mail to a Smart Host

In addition to the options described for the sendmail.mc file, it's easy to configure sendmail to relay e-mail to a smart host. The default sendmail.mc file provides a hint with the following commented directive:

```
dnl define(`SMART_HOST', `smtp.your.provider')dnl
```

If the remote smart host is smtp.example.org, you'd activate the directive as follows:

```
define(`SMART_HOST', `smtp.example.org')dnl
```

Configure User- and Host-Based sendmail Security

As suggested in the discussion of the /etc/mail/access file, host-based security can be configured there. For example, as long as the limitation to the 127.0.0.1 system is removed from or commented out of the sendmail.mc macro, host-based security is controlled through the /etc/mail/access file. For example, the following entry rejects all users named michael:

```
michael@ REJECT
```

In addition, unlike Postfix, sendmail can be protected with TCP Wrappers, as discussed in Chapter 10. In other words, you could use the hosts.allow and hosts.deny files in the /etc directory to limit access to certain users and hosts.

Test the Current sendmail Configuration

As with Postfix, the **telnet** command can be used to review the current status of the sendmail service. Based on the default configuration of Postfix, an active version of this service should be listening on port 25. In that case, a **telnet localhost 25** command should return something similar to the following messages:

```
Trying ::1...
telnet: connect to address ::1: Connection refused
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 server1.example.com ESMTP Sendmail 8.14.4/8.14.4; Wed, 23 Feb 2011 16:58:20
-0800
```

The first connection refused message confirms that the sendmail configuration file does not normally listen to IPv6 networking. To set up sendmail.mc for both IPv4 and IPv6 beyond the localhost system, deactivate the following directive (by adding the **dnl** in front):

```
dnl DAEMON_OPTIONS(`Port=smtp,Addr=127.0.0.1, Name=MTA')dnl
```

And activate the following directive:

```
DAEMON_OPTIONS(`Name=MTA-v4, Family=inet, Name=MTA-v6, Family=inet6')
```

But that's required only if IPv6 networking is required, which is not always the case. If only IPv4 networking is required, comment out this directive as well.

SCENARIO & SOLUTION

You're told to configure an SMTP server for the 192.168.0.0/24 network.	Use the default Postfix server; modify the **myhostname**, **mydomain**, **myorigin**, **inet_interfaces**, and **mynetworks** directives in /etc/postfix/main.cf. Remember to process the non-cf files into databases with the **postmap** command.
You're told to allow access just to the SMTP server for user1, user2, and user3.	Create the noted users with a /sbin/nologin default shell.
You're told to set up sendmail on RHEL 6.	Stop the Postfix service, and use **alternatives** to change the default MTA to sendmail.
You're told to configure sendmail to allow access to all systems.	In the sendmail.mc file, comment out the **DAEMON_OPTIONS** directive associated with the loopback address.

CERTIFICATION SUMMARY

Red Hat includes two servers associated with the SMTP protocol: Postfix and sendmail. Red Hat has switched default SMTP services between RHEL 5 and RHEL 6; it is now Postfix. Whichever SMTP service you select, it's just one part of the hierarchy of services for e-mail. Both Postfix and sendmail are part of the "E-mail server" package group. If you want to switch between SMTP services, the **alternatives --config mta** command can help. Mail log information can be found in the /var/log/maillog file. You can test the current status of both SMTP services from the local system with the **telnet localhost 25** command.

Postfix is somewhat easier to configure than sendmail. Different Postfix configuration files can be found in the /etc/postfix directory. User and host limits can be configured in the access file. Several other files relate to redirected or renamed e-mail accounts or domains. You'll need to modify several Postfix configuration directives in the /etc/postfix/main.cf file, including **myhostname**, **mydomain**, **myorigin**, **inet_interfaces**, and **mynetworks**. The **relayhost** directive can help configure forwarding to a smart host. If you need to configure Postfix authentication, refer to additional directives in the /usr/share/doc/postfix-2.6.6 directory, in the README-Postfix-SASL-RedHat.txt file.

The sendmail service may still be the most popular SMTP server on the Internet. It has configuration files in the /etc/mail directory. The two main configuration files are: sendmail.cf and submit.cf. You can configure these files through macros configured in the sendmail.mc and submit.mc files. You can configure user- and host-based security with the help of the /etc/mail/access file. The sendmail.mc file includes a commented example directive to help you configure a connection to a remote smart host.

TWO-MINUTE DRILL

Here are some of the key points from the certification objectives in Chapter 13.

A Variety of E-Mail Agents

- ❑ RHEL 6 allows you to select between Postfix and sendmail. Both are MTAs. Don't activate both.
- ❑ You can use the **alternatives --config mta** command to switch between Postfix and sendmail.
- ❑ Mail server information is logged in the /var/log/maillog file.

The Configuration of Postfix

- ❑ The Postfix server is easier to configure through configuration files in the /etc/postfix directory. In fact, you can configure the main.cf file directly.
- ❑ You can configure Postfix user and host security Postfix in /etc/aliases.
- ❑ You can set up various kinds of e-mail forwarding in files like canonical, generic, and relocated, all in the /etc/postfix directory.
- ❑ The /usr/share/doc/postfix-2.6.6 directory includes information on user authentication options in the README-Postfix-SASL-RedHat.txt file.
- ❑ The **relayhost** command can be used to set up a connection to a smart host.
- ❑ You can test a standard Postfix configuration from the local system with the **telnet localhost 25** command. (The same command works for a standard sendmail configuration as well.)

The Other SMTP Service: sendmail

- ❑ The main sendmail configuration file is /etc/mail/sendmail.cf. It's easier to configure sendmail through its macro file, /etc/mail/sendmail.mc.
- ❑ You can open up access to all systems in the sendmail.mc file by commenting out the **DAEMON_OPTIONS** directive.
- ❑ The sendmail.mc file includes a commented suggestion to configure a connection to a smart host.
- ❑ You can customize the computers allowed to access a sendmail server through the access and virtusertable files in the /etc/mail directory.
- ❑ The **/etc/mail/make** command processes all files in the **/etc/mail** directory.

SELF TEST

The following questions will help measure your understanding of the material presented in this chapter. As no multiple-choice questions appear on the Red Hat exams, no multiple-choice questions appear in this book. These questions exclusively test your understanding of the chapter. It is okay if you have another way of performing a task. Getting results, not memorizing trivia, is what counts on the Red Hat exams. There may be more than one answer to many of the questions.

A Variety of E-Mail Agents

1. List two examples of an MTA supported on RHEL 6.

   ____

2. What command can be used to switch between installed Postfix and sendmail services?

   ____

The Configuration of Postfix

3. How would you change the following directive in /etc/postfix/main.cf to open Postfix to all systems?

   ```
   inet_interfaces = localhost
   ```

   ____

4. If you use /etc/aliases for forwarding e-mail, what command processes these files into an appropriate database file for Postfix?

   ____

5. What file supports limits on hosts that can connect to Postfix?

   ____

6. What directive in the main.cf file is used to specify the domain served by the Postfix server?

   ____

7. What directive in the main.cf file is used to specify the IP address network served by the Postfix server?

   ____

8. In what directory can you find documentation associated with the Postfix server?

____

The Other SMTP Service: sendmail

9. In what file would you store forwarding e-mail addresses?

____

10. Why would you want to comment out the following directive in sendmail.mc?

```
DAEMON_OPTIONS(`Port=smtp,Addr=127.0.0.1, Name=MTA')dnl
```

____

11. What service is no longer needed if you comment out the following directive in sendmail.mc?

```
FEATURE(`accept_unresolvable_domains')dnl
```

____

12. What command processes all files in /etc/mail?

____

LAB QUESTIONS

Several of these labs involve configuration exercises. You should do these exercises on test machines only. It's assumed that you're running these exercises on virtual machines such as KVM. For this chapter, it's also assumed that you may be changing the configuration of a physical host system for such virtual machines.

Red Hat presents its exams electronically. For that reason, the labs in this and future chapters are available from the CD that accompanies the book, in the Chapter13/ subdirectory. In case you haven't yet set up RHEL 6 on a system, refer to Chapter 1 for installation instructions.

The answers for each lab follow the Self Test answers for the fill-in-the-blank questions.

SELF TEST ANSWERS

A Variety of E-Mail Agents

1. Three examples of MTAs supported on RHEL 6 are Postfix, sendmail, and Dovecot.
2. The command that can be used to help switch between the Postfix and sendmail MTAs is **alternatives --config mta**.

The Configuration of Postfix

3. The simplest solution is to change the directive to

   ```
   inet_interfaces = all
   ```

4. Forwarding e-mail addresses for both sendmail and Postfix are normally stored in /etc/aliases. Make sure to process these files into appropriate databases; for /etc/aliases, the database is updated with the **newaliases** command.
5. The file that supports limits on hosts that can connect to Postfix is /etc/postfix/access.
6. The directive in the main.cf file that is used to specify the domain served by the Postfix server is **mydomain**.
7. The directive in the main.cf file is used to specify the IP address network served by the Postfix server is **mynetworks**.
8. You find documentation associated with the Postfix server in the /usr/share/doc/postfix-2.6.6 directory.

The Other SMTP Service: sendmail

9. Forwarding e-mail addresses for both sendmail and Postfix are normally stored in /etc/aliases. If you're forwarding e-mail for entire domains, the appropriate file is /etc/mail/domaintable. Make sure to process these files into appropriate databases; for /etc/aliases, the database is updated with the **newaliases** command. For /etc/mail/domaintable, the database is updated with the **/etc/mail/make** command.
10. If you comment out the noted directive, access is supported by all systems allowed to connect.
11. If you comment out the noted directive in sendmail.mc, a reliable DNS server is not required.
12. The command that processes all files in the /etc/mail directory is **/etc/mail/make**.

LAB ANSWERS

For most of these labs, you may be using an e-mail client like **mutt**. To send an e-mail to user michael@localhost, take the following steps:

1. Run the **mutt michael@localhost** command. The **To: michael@localhost** message should appear.
2. Press ENTER. At the **Subject:** prompt, enter an appropriate test subject name and press ENTER.
3. You're taken to a blank screen in the vi editor. Use commands appropriate to that editor to a screen similar to that shown in Figure 13-5.
4. From the screen shown in Figure 13-5, press **y** to send the noted message.

In addition, you may be verifying e-mail receipt in a username file in the /var/spool/mail directory. Normally, such e-mail can be reviewed from within a user account with the **mail** command. In addition, you may be verifying access to a running SMTP server with the **telnet *ip_address* 25** command, where *ip_address* is the IP address of the SMTP server.

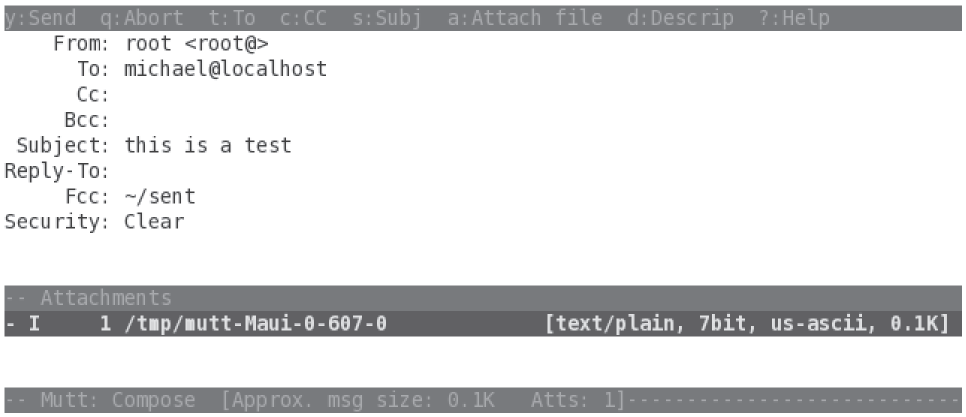
After making a configuration change, don't forget to process the file appropriately. For Postfix, the **postmap *filename*** command processes the file. For sendmail, the /etc/mail/make script processes files in the /etc/mail directory. For the common /etc/aliases file, the **newaliases** command processes the file. And don't forget to make sure the service actually re-reads the new configuration files.

Lab 1

In Postfix, to disable local-only access in the /etc/postfix/main.cf file, change the **inet_interfaces** directive to accept **all** connections:

```
inet_interfaces = all
```

FIGURE 13-5 The mutt e-mail client



```
y:Send  q:Abort  t:To  c:CC  s:Subj  a:Attach file  d:Descrip  ?:Help
    From: root <root@>
      To: michael@localhost
      Cc:
     Bcc:
 Subject: this is a test
Reply-To:
     Fcc: ~/sent
Security: Clear


-- Attachments
- I     1 /tmp/mutt-Maui-0-607-0            [text/plain, 7bit, us-ascii, 0.1K]


-- Mutt: Compose  [Approx. msg size: 0.1K   Atts: 1]---------------------------
```

But to meet the requirements of the lab, you'll want to retain the default value of that directive:

```
inet_interfaces = localhost
```

Make sure Postfix is active (and any alternative mail servers such as sendmail are not).

In general, to verify authentication on an SMTP server, connect from the local system with the **telnet localhost 25** command. When you see a message similar to

```
220 maui.example.com ESMTP Postfix
```

type in the following command:

```
EHLO localhost
```

Depending on the configuration, you should see messages similar to the following:

```
250-AUTH LOGIN PLAIN

250-AUTH=LOGIN PLAIN
```

To verify receipt of e-mail in a user account, log in to that account, or at least verify the time stamp associated with the username in the /var/mail directory. To make sure e-mail directed to the root user is redirected to a regular user account, you'd add a line like the following to the /etc/aliases file:

```
root:  michael
```

Given the wording for the question, any standard user account would be acceptable. Of course, to implement this change, you'll have to run the **newaliases** command, which processes this file into the /etc/aliases.db file.

To make sure this works, you'll want to use a command line client such as **mutt** or even **mail** as defined in Chapter 2. For example, if you send a test e-mail to the local root user, the message should be received by user michael (or whomever is configured in the /etc/aliases file to receive e-mail forwarded from the root user).

Lab 2

To enable access from more than just the localhost, you'll need to modify the inet_interfaces directive in /etc/postfix/main.cf to

```
inet_interfaces = all
```

The next job is to limit access to a specific network, in this case, example.com. While there are options in /etc/postfix files, perhaps the most efficient way to limit access to a specific network is with an appropriate **iptables**-based firewall rule. For example, the following custom rule would limit access to TCP port 25 to systems on the given IP address network. The network shown is based on the originally defined configuration for example.com, the 192.168.122.0/24 network:

```
-A INPUT -m state --state NEW -m tcp -p tcp -s 192.168.122.0/24 --dport 25 -j
ACCEPT
```

In addition, you'll need to set up this network in the /etc/postfix/access file with a rule like the following:

```
192.168.122 OK
```

Once Postfix is running, you should be able to confirm the result with an appropriate **telnet** command from a remote system. For example, if Postfix is configured on a system with a 192.168.122.50 IP address, the command would be

```
# telnet 192.168.122.50 25
```

The configuration of a smart host in Postfix is based on the **relayhost** directive. For the parameters given in the Lab, if the physical host is located on system maui.example.com, the directive in the main.cf file would be

```
relayhost = maui.example.com
```

If Postfix on the server1.example.com system is properly configured as a smart host, e-mails to the forwarded host should be reliably delivered, and even logged into the appropriate /var/log/maillog file.

Lab 3

With the **iptables** rule shown in Lab 2, access should already be prohibited from other networks. However, if you did not include the 192.168.122.0/24 network address in the **iptables** rule shown in the answer to Lab 2, a different approach is available. You can use options available in the /etc/postfix/access file, such as the following to reject messages from the example.org network. If your example.org network uses a different IP address, revise accordingly.

```
192.168.100 REJECT
```

It should be easy to verify connections to the SMTP server from a remote system. From prohibited networks, when you run a command like: **telnet 192.168.122.50 25** from a host on the 192.168.100.0/24 network, you should see the following output:

```
Trying 192.168.122.50...

telnet: connect to address 192.168.122.50: No route to host
```

Lab 4

This lab should be straightforward. As long as sendmail and sendmail-cf are installed on the local system, the basic steps to make the move from Postfix to sendmail are as follows:

1. Stop the Postfix service with a command like **/etc/init.d/postfix stop**.
2. Run the **alternatives --config mta** script, and select sendmail.

Lab 5

This lab implicitly assumes both IPv4 and IPv6 networking is in operation. With that in mind, open the sendmail.mc file in the /etc/mail directory. To configure sendmail to accept connections from both IPv4 and IPv6 addresses, first disable the following directive. The added **dnl** in front turns it into a comment.

```
dnl DAEMON_OPTIONS(`Port=smtp,Addr=127.0.0.1, Name=MTA')dnl
```

Next, to let sendmail listen for both IPv4 and IPv6 traffic, activate the following directive:

```
DAEMON_OPTIONS(`Name=MTA-v4, Family=inet, Name=MTA-v6, Family=inet6')
```

If successful, you should be able to connect to the system with either of the following commands:

```
# telnet ::1 25
# telnet 127.0.0.1 25
```

Forwarding from the root account can be enabled through the /etc/aliases file, as explained in the answer to Lab 1.

Lab 6

In sendmail, to disable local-only access in the /etc/mail/sendmail.mc file, comment out the following line. Unlike in most Linux configuration files, the comment code is a **dnl** at the start of this line:

```
DNL DAEMON_OPTIONS(`Port=smtp,Addr=127.0.0.1, Name=MTA')dnl
```

The **dnl** at the end of the line does not affect the command to its left.

Next, you'll want to enable support through /etc/mail/access. To allow access to the example.com network as discussed in this book, add the following line to that file:

```
192.168.122              RELAY
```

For this lab, assume the sendmail system is on server1.example.com, with IP address 192.168.122.50, and the physical system is on maui.example.com on IP address 192.168.122.1. Return to the sendmail.mc file, and then look at this directive:

```
dnl define(`SMART_HOST', `smtp.your.provider')dnl
```

Based on the conditions given, you'd change that directive to

```
define(`SMART_HOST', `maui.example.com')dnl
```

Lab 7

This lab is quite similar to Lab 2. While there are options in the sendmail configuration files, the most efficient way to limit access to a specific network is still with an appropriate **iptables**-based firewall rule. The rule discussed in the answer to Lab 2 would also work in this case. Alternatively, you can configure access limits in the /etc/mail/access file, such as

```
Connect: 192.168.122 OK
Connect: 192.168.100 REJECT
```