**Chapter 23. Email**

[23.1. Email Protocols](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-email.html#s1-email-protocols)

[23.2. Email Program Classifications](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-email-types.html)

[23.3. Mail Transport Agents](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-email-mta.html)

[23.4. Mail Transport Agent (MTA) Configuration](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-email-switchmail.html)

[23.5. Mail Delivery Agents](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-email-mda.html)

[23.6. Mail User Agents](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-email-mua.html)

[23.7. Additional Resources](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-email-additional-resources.html)

The birth of electronic mail (*email*) occurred in the early 1960s. The mailbox was a file in a user's home directory that was readable only by that user. Primitive mail applications appended new text messages to the bottom of the file, making the user wade through the constantly growing file to find any particular message. This system was only capable of sending messages to users on the same system.

The first network transfer of an electronic mail message file took place in 1971 when a computer engineer named Ray Tomlinson sent a test message between two machines via ARPANET — the precursor to the Internet. Communication via email soon became very popular, comprising 75 percent of ARPANET's traffic in less than two years.

Today, email systems based on standardized network protocols have evolved into some of the most widely used services on the Internet. Red Hat Enterprise Linux offers many advanced applications to serve and access email.

This chapter reviews modern email protocols in use today and some of the programs designed to send and receive email.

**23.1. Email Protocols**

Today, email is delivered using a client/server architecture. An email message is created using a mail client program. This program then sends the message to a server. The server then forwards the message to the recipient's email server, where the message is then supplied to the recipient's email client.

To enable this process, a variety of standard network protocols allow different machines, often running different operating systems and using different email programs, to send and receive email.

The following protocols discussed are the most commonly used in the transfer of email.

**23.1.1. Mail Transport Protocols**

Mail delivery from a client application to the server, and from an originating server to the destination server, is handled by the *Simple Mail Transfer Protocol* (*SMTP*).

**23.1.1.1. SMTP**

The primary purpose of SMTP is to transfer email between mail servers. However, it is critical for email clients as well. To send email, the client sends the message to an outgoing mail server, which in turn contacts the destination mail server for delivery. For this reason, it is necessary to specify an SMTP server when configuring an email client.

Under Red Hat Enterprise Linux, a user can configure an SMTP server on the local machine to handle mail delivery. However, it is also possible to configure remote SMTP servers for outgoing mail.

One important point to make about the SMTP protocol is that it does not require authentication. This allows anyone on the Internet to send email to anyone else or even to large groups of people. It is this characteristic of SMTP that makes junk email or *spam* possible. Imposing relay restrictions limits random users on the Internet from sending email through your SMTP server, to other servers on the internet. Servers that do not impose such restrictions are called *open relay* servers.

By default, Sendmail (/usr/sbin/sendmail) is the default SMTP program under Red Hat Enterprise Linux. However, a simpler mail server application called Postfix (/usr/sbin/postfix) is also available.

**23.1.2. Mail Access Protocols**

There are two primary protocols used by email client applications to retrieve email from mail servers: the *Post Office Protocol* (*POP*) and the *Internet Message Access Protocol* (*IMAP*).

**23.1.2.1. POP**

The default POP server under Red Hat Enterprise Linux is /usr/lib/cyrus-imapd/pop3d and is provided by the cyrus-imapd package. When using a POP server, email messages are downloaded by email client applications. By default, most POP email clients are automatically configured to delete the message on the email server after it has been successfully transferred, however this setting usually can be changed.

POP is fully compatible with important Internet messaging standards, such as *Multipurpose Internet Mail Extensions* (*MIME*), which allow for email attachments.

POP works best for users who have one system on which to read email. It also works well for users who do not have a persistent connection to the Internet or the network containing the mail server. Unfortunately for those with slow network connections, POP requires client programs upon authentication to download the entire content of each message. This can take a long time if any messages have large attachments.

The most current version of the standard POP protocol is POP3.

There are, however, a variety of lesser-used POP protocol variants:

* *APOP* — POP3 with MDS authentication. An encoded hash of the user's password is sent from the email client to the server rather then sending an unencrypted password.
* *KPOP* — POP3 with Kerberos authentication. Refer to [Section 42.6, “Kerberos”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-kerberos.html) for more information.
* *RPOP* — POP3 with RPOP authentication. This uses a per-user ID, similar to a password, to authenticate POP requests. However, this ID is not encrypted, so RPOP is no more secure than standard POP.

For added security, it is possible to use *Secure Socket Layer* (*SSL*) encryption for client authentication and data transfer sessions. This can be enabled by using the ipop3s service or by using the /usr/sbin/stunnel program. Refer to [Section 23.6.1, “Securing Communication”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-email-mua.html#s2-email-security) for more information.

**23.1.2.2. IMAP**

The default IMAP server under Red Hat Enterprise Linux is /usr/lib/cyrus-imapd/imapd and is provided by the cyrus-imapd package. When using an IMAP mail server, email messages remain on the server where users can read or delete them. IMAP also allows client applications to create, rename, or delete mail directories on the server to organize and store email.

IMAP is particularly useful for those who access their email using multiple machines. The protocol is also convenient for users connecting to the mail server via a slow connection, because only the email header information is downloaded for messages until opened, saving bandwidth. The user also has the ability to delete messages without viewing or downloading them.

For convenience, IMAP client applications are capable of caching copies of messages locally, so the user can browse previously read messages when not directly connected to the IMAP server.

IMAP, like POP, is fully compatible with important Internet messaging standards, such as MIME, which allow for email attachments.

For added security, it is possible to use *SSL* encryption for client authentication and data transfer sessions. This can be enabled by using the imaps service, or by using the /usr/sbin/stunnel program. Refer to [Section 23.6.1, “Securing Communication”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-email-mua.html#s2-email-security) for more information.

Other free, as well as commercial, IMAP clients and servers are available, many of which extend the IMAP protocol and provide additional functionality. A comprehensive list can be found online at <http://www.imap.org/products/longlist.htm>.

**23.1.2.3. Dovecot**

The imap-login and pop3-login daemons which implement the IMAP and POP3 protocols are included in the dovecot package. The use of IMAP and POP is configured through dovecot; by default dovecot runs only IMAP. To configure dovecot to use POP:

1. Edit /etc/dovecot.conf to have the line:

protocols = imap imaps pop3 pop3s

1. Make that change operational for the current session by running the command:

/sbin/service dovecot restart

1. Make that change operational after the next reboot by running the command:

chkconfig dovecot on

Please note that dovecot only reports that it started the IMAP server, but also starts the POP3 server.

Unlike SMTP, both of these protocols require connecting clients to authenticate using a username and password. By default, passwords for both protocols are passed over the network unencrypted.

To configure SSL on dovecot:

* Edit the dovecot configuration file /etc/pki/dovecot/dovecot-openssl.conf as you prefer. However in a typical installation, this file does not require modification.
* Rename, move or delete the files /etc/pki/dovecot/certs/dovecot.pem and /etc/pki/dovecot/private/dovecot.pem.
* Execute the /usr/share/doc/dovecot-1.0/examples/mkcert.sh script which creates the dovecot self signed certificates. The certificates are copied in the /etc/pki/dovecot/certs and /etc/pki/dovecot/private directories. To implement the changes, restart dovecot (/sbin/service dovecot restart).

## 23.2. Email Program Classifications

In general, all email applications fall into at least one of three classifications. Each classification plays a specific role in the process of moving and managing email messages. While most users are only aware of the specific email program they use to receive and send messages, each one is important for ensuring that email arrives at the correct destination.

### 23.2.1. Mail Transport Agent

A *Mail Transport Agent* (*MTA*) transports email messages between hosts using SMTP. A message may involve several MTAs as it moves to its intended destination.

While the delivery of messages between machines may seem rather straightforward, the entire process of deciding if a particular MTA can or should accept a message for delivery is quite complicated. In addition, due to problems from spam, use of a particular MTA is usually restricted by the MTA's configuration or the access configuration for the network on which the MTA resides.

Many modern email client programs can act as an MTA when sending email. However, this action should not be confused with the role of a true MTA. The sole reason email client programs are capable of sending email like an MTA is because the host running the application does not have its own MTA. This is particularly true for email client programs on non-UNIX-based operating systems. However, these client programs only send outbound messages to an MTA they are authorized to use and do not directly deliver the message to the intended recipient's email server.

Since Red Hat Enterprise Linux installs two MTAs, Sendmail and Postfix, email client programs are often not required to act as an MTA. Red Hat Enterprise Linux also includes a special purpose MTA called Fetchmail.

For more information on Sendmail, Postfix, and Fetchmail, refer to [Section 23.3, “Mail Transport Agents”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-email-mta.html).

### 23.2.2. Mail Delivery Agent

A *Mail Delivery Agent* (*MDA*) is invoked by the MTA to file incoming email in the proper user's mailbox. In many cases, the MDA is actually a *Local Delivery Agent* (*LDA*), such as mail or Procmail.

Any program that actually handles a message for delivery to the point where it can be read by an email client application can be considered an MDA. For this reason, some MTAs (such as Sendmail and Postfix) can fill the role of an MDA when they append new email messages to a local user's mail spool file. In general, MDAs do not transport messages between systems nor do they provide a user interface; MDAs distribute and sort messages on the local machine for an email client application to access.

### 23.2.3. Mail User Agent

A *Mail User Agent* (*MUA*) is synonymous with an email client application. An MUA is a program that, at the very least, allows a user to read and compose email messages. Many MUAs are capable of retrieving messages via the POP or IMAP protocols, setting up mailboxes to store messages, and sending outbound messages to an MTA.

MUAs may be graphical, such as Evolution, or have a very simple, text-based interface, such as mutt.

## 23.3. Mail Transport Agents

Red Hat Enterprise Linux includes two primary MTAs, Sendmail and Postfix. Sendmail is configured as the default MTA, although it is easy to switch the default MTA to Postfix.

### 23.3.1. Sendmail

Sendmail's core purpose, like other MTAs, is to safely transfer email among hosts, usually using the SMTP protocol. However, Sendmail is highly configurable, allowing control over almost every aspect of how email is handled, including the protocol used. Many system administrators elect to use Sendmail as their MTA due to its power and scalability.

#### 23.3.1.1. Purpose and Limitations

It is important to be aware of what Sendmail is and what it can do, as opposed to what it is not. In these days of monolithic applications that fulfill multiple roles, Sendmail may seem like the only application needed to run an email server within an organization. Technically, this is true, as Sendmail can spool mail to each users' directory and deliver outbound mail for users. However, most users actually require much more than simple email delivery. Users usually want to interact with their email using an MUA, that uses POP or IMAP, to download their messages to their local machine. Or, they may prefer a Web interface to gain access to their mailbox. These other applications can work in conjunction with Sendmail, but they actually exist for different reasons and can operate separately from one another.

It is beyond the scope of this section to go into all that Sendmail should or could be configured to do. With literally hundreds of different options and rule sets, entire volumes have been dedicated to helping explain everything that can be done and how to fix things that go wrong. Refer to the [Section 23.7, “Additional Resources”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-email-additional-resources.html) for a list of Sendmail resources.

This section reviews the files installed with Sendmail by default and reviews basic configuration changes, including how to stop unwanted email (spam) and how to extend Sendmail with the *Lightweight Directory Access Protocol (LDAP)*.

#### 23.3.1.2. The Default Sendmail Installation

The Sendmail executable is /usr/sbin/sendmail.

Sendmail's lengthy and detailed configuration file is /etc/mail/sendmail.cf. Avoid editing the sendmail.cf file directly. To make configuration changes to Sendmail, edit the /etc/mail/sendmail.mc file, back up the original /etc/mail/sendmail.cf, and use the following alternatives to generate a new configuration file:

* Use the included makefile in /etc/mail (make all -C /etc/mail) to create a new /etc/mail/sendmail.cf configuration file. All other generated files in /etc/mail (db files) will be regenerated if needed. The old makemap commands are still usable. The make command will automatically be used by service sendmail start | restart | reload if the make package is installed.
* Alternatively you may use the included m4 macro processor to create a new /etc/mail/sendmail.cf.

More information on configuring Sendmail can be found in [Section 23.3.1.3, “Common Sendmail Configuration Changes”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-email-mta.html#s3-email-mta-sendmail-changes).

Various Sendmail configuration files are installed in the /etc/mail/ directory including:

* access — Specifies which systems can use Sendmail for outbound email.
* domaintable — Specifies domain name mapping.
* local-host-names — Specifies aliases for the host.
* mailertable — Specifies instructions that override routing for particular domains.
* virtusertable — Specifies a domain-specific form of aliasing, allowing multiple virtual domains to be hosted on one machine.

Several of the configuration files in /etc/mail/, such as access, domaintable, mailertable and virtusertable, must actually store their information in database files before Sendmail can use any configuration changes. To include any changes made to these configurations in their database files, run the command

makemap hash /etc/mail/*<name>* < /etc/mail/*<name>*

where *<name>* is replaced with the name of the configuration file to convert.

For example, to have all emails addressed to the example.com domain delivered to <[bob@other-example.com](mailto:bob@other-example.com)>, add the following line to the virtusertable file:

@example.com bob@other-example.com

To finalize the change, the virtusertable.db file must be updated using the following command as root:

makemap hash /etc/mail/virtusertable < /etc/mail/virtusertable

This creates an updated virtusertable.db file containing the new configuration.

#### 23.3.1.3. Common Sendmail Configuration Changes

When altering the Sendmail configuration file, it is best not to edit an existing file, but to generate an entirely new /etc/mail/sendmail.cf file.

## Caution

Before changing the sendmail.cf file, it is a good idea to create a backup copy.

To add the desired functionality to Sendmail, edit the /etc/mail/sendmail.mc file as the root user. When finished, use the m4 macro processor to generate a new sendmail.cf by executing the following command:

m4 /etc/mail/sendmail.mc > /etc/mail/sendmail.cf

By default, the m4 macro processor is installed with Sendmail but is part of the m4 package.

After creating a new /etc/mail/sendmail.cf file, restart Sendmail for the changes to take effect. The easiest way to do this is to type the following command:

/sbin/service sendmail restart

## Important

The default sendmail.cf file does not allow Sendmail to accept network connections from any host other than the local computer. To configure Sendmail as a server for other clients, edit the /etc/mail/sendmail.mc file, and either change the address specified in the Addr= option of the DAEMON\_OPTIONS directive from 127.0.0.1 to the IP address of an active network device or comment out the DAEMON\_OPTIONS directive all together by placing dnl at the beginning of the line. When finished, regenerate /etc/mail/sendmail.cf by executing the following command:

m4 /etc/mail/sendmail.mc > /etc/mail/sendmail.cf

The default configuration which ships with Red Hat Enterprise Linux works for most SMTP-only sites. However, it does not work for UUCP (UNIX to UNIX Copy) sites. If using UUCP mail transfers, the /etc/mail/sendmail.mc file must be reconfigured and a new /etc/mail/sendmail.cf must be generated.

Consult the /usr/share/sendmail-cf/README file before editing any files in the directories under the /usr/share/sendmail-cf directory, as they can affect the future configuration of /etc/mail/sendmail.cf files.

#### 23.3.1.4. Masquerading

One common Sendmail configuration is to have a single machine act as a mail gateway for all machines on the network. For instance, a company may want to have a machine called mail.example.com that handles all of their email and assigns a consistent return address to all outgoing mail.

In this situation, the Sendmail server must masquerade the machine names on the company network so that their return address is user@example.com instead of user@host.example.com.

To do this, add the following lines to /etc/mail/sendmail.mc:

FEATURE(always\_add\_domain)dnl

FEATURE(`masquerade\_entire\_domain') dnl

FEATURE(`masquerade\_envelope') dnl

FEATURE(`allmasquerade') dnl

MASQUERADE\_AS(`bigcorp.com.') dnl

MASQUERADE\_DOMAIN(`bigcorp.com.') dnl

MASQUERADE\_AS(bigcorp.com) dnl

After generating a new sendmail.cf using m4, this configuration makes all mail from inside the network appear as if it were sent from bigcorp.com.

#### 23.3.1.5. Stopping Spam

Email spam can be defined as unnecessary and unwanted email received by a user who never requested the communication. It is a disruptive, costly, and widespread abuse of Internet communication standards.

Sendmail makes it relatively easy to block new spamming techniques being employed to send junk email. It even blocks many of the more usual spamming methods by default. Main anti-spam features available in sendmail are *header checks, relaying denial (default from version 8.9), access database and sender information checks.*

For example, forwarding of SMTP messages, also called relaying, has been disabled by default since Sendmail version 8.9. Before this change occurred, Sendmail directed the mail host (x.edu) to accept messages from one party (y.com) and sent them to a different party (z.net). Now, however, Sendmail must be configured to permit any domain to relay mail through the server. To configure relay domains, edit the /etc/mail/relay-domains file and restart Sendmail.

However, many times users are bombarded with spam from other servers throughout the Internet. In these instances, Sendmail's access control features available through the /etc/mail/access file can be used to prevent connections from unwanted hosts. The following example illustrates how this file can be used to both block and specifically allow access to the Sendmail server:

badspammer.com ERROR:550 "Go away and do not spam us anymore"

tux.badspammer.com OK

10.0 RELAY

This example shows that any email sent from badspammer.com is blocked with a 550 RFC-821 compliant error code, with a message sent back to the spammer. Email sent from the tux.badspammer.com sub-domain, is accepted. The last line shows that any email sent from the 10.0.\*.\* network can be relayed through the mail server.

Because /etc/mail/access.db is a database, use makemap to activate any changes. Do this using the following command as root:

makemap hash /etc/mail/access < /etc/mail/access

Message header analysis allows you to reject mail based on header contents. SMTP servers store information about an emails journey in the message header. As the message travels from one MTA to another, each puts in a "Received" header above all the other Received headers. It is however important to note that this information may be altered by spammers.

The above examples only represent a small part of what Sendmail can do in terms of allowing or blocking access. Refer to the /usr/share/sendmail-cf/README for more information and examples.

Since Sendmail calls the Procmail MDA when delivering mail, it is also possible to use a spam filtering program, such as SpamAssassin, to identify and file spam for users. Refer to [Section 23.5.2.6, “Spam Filters”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-email-mda.html#s3-email-mda-spam) for more about using SpamAssassin.

#### 23.3.1.6. Using Sendmail with LDAP

Using the *Lightweight Directory Access Protocol (LDAP)* is a very quick and powerful way to find specific information about a particular user from a much larger group. For example, an LDAP server can be used to look up a particular email address from a common corporate directory by the user's last name. In this kind of implementation, LDAP is largely separate from Sendmail, with LDAP storing the hierarchical user information and Sendmail only being given the result of LDAP queries in pre-addressed email messages.

However, Sendmail supports a much greater integration with LDAP, where it uses LDAP to replace separately maintained files, such as aliases and virtusertables, on different mail servers that work together to support a medium- to enterprise-level organization. In short, LDAP abstracts the mail routing level from Sendmail and its separate configuration files to a powerful LDAP cluster that can be leveraged by many different applications.

The current version of Sendmail contains support for LDAP. To extend the Sendmail server using LDAP, first get an LDAP server, such as **OpenLDAP**, running and properly configured. Then edit the /etc/mail/sendmail.mc to include the following:

LDAPROUTE\_DOMAIN('*yourdomain.com*')dnl

FEATURE('ldap\_routing')dnl

## Note

This is only for a very basic configuration of Sendmail with LDAP. The configuration can differ greatly from this depending on the implementation of LDAP, especially when configuring several Sendmail machines to use a common LDAP server.

Consult /usr/share/sendmail-cf/README for detailed LDAP routing configuration instructions and examples.

Next, recreate the /etc/mail/sendmail.cf file by running m4 and restarting Sendmail. Refer to [Section 23.3.1.3, “Common Sendmail Configuration Changes”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-email-mta.html#s3-email-mta-sendmail-changes) for instructions.

For more information on LDAP, refer to [Chapter 24, *Lightweight Directory Access Protocol (LDAP)*](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-ldap.html).

### 23.3.2. Postfix

Originally developed at IBM by security expert and programmer Wietse Venema, Postfix is a Sendmail-compatible MTA that is designed to be secure, fast, and easy to configure.

To improve security, Postfix uses a modular design, where small processes with limited privileges are launched by a *master* daemon. The smaller, less privileged processes perform very specific tasks related to the various stages of mail delivery and run in a change rooted environment to limit the effects of attacks.

Configuring Postfix to accept network connections from hosts other than the local computer takes only a few minor changes in its configuration file. Yet for those with more complex needs, Postfix provides a variety of configuration options, as well as third party add ons that make it a very versatile and full-featured MTA.

The configuration files for Postfix are human readable and support upward of 250 directives. Unlike Sendmail, no macro processing is required for changes to take effect and the majority of the most commonly used options are described in the heavily commented files.

## Important

Before using Postfix, the default MTA must be switched from Sendmail to Postfix.

#### 23.3.2.1. The Default Postfix Installation

The Postfix executable is /usr/sbin/postfix. This daemon launches all related processes needed to handle mail delivery.

Postfix stores its configuration files in the /etc/postfix/ directory. The following is a list of the more commonly used files:

* access — Used for access control, this file specifies which hosts are allowed to connect to Postfix.
* aliases — A configurable list required by the mail protocol.
* main.cf — The global Postfix configuration file. The majority of configuration options are specified in this file.
* master.cf — Specifies how Postfix interacts with various processes to accomplish mail delivery.
* transport — Maps email addresses to relay hosts.

## Important

The default /etc/postfix/main.cf file does not allow Postfix to accept network connections from a host other than the local computer. For instructions on configuring Postfix as a server for other clients, refer to [Section 23.3.2.2, “Basic Postfix Configuration”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-email-mta.html#s3-email-mta-postfix-conf).

When changing some options within files in the /etc/postfix/ directory, it may be necessary to restart the postfix service for the changes to take effect. The easiest way to do this is to type the following command:

/sbin/service postfix restart

#### 23.3.2.2. Basic Postfix Configuration

By default, Postfix does not accept network connections from any host other than the local host. Perform the following steps as root to enable mail delivery for other hosts on the network:

* Edit the /etc/postfix/main.cf file with a text editor, such as vi.
* Uncomment the mydomain line by removing the hash mark (#), and replace *domain.tld* with the domain the mail server is servicing, such as example.com.
* Uncomment the myorigin = $mydomain line.
* Uncomment the myhostname line, and replace *host.domain.tld* with the hostname for the machine.
* Uncomment the mydestination = $myhostname, localhost.$mydomain line.
* Uncomment the mynetworks line, and replace *168.100.189.0/28* with a valid network setting for hosts that can connect to the server.
* Uncomment the inet\_interfaces = all line.
* Restart the postfix service.

Once these steps are complete, the host accepts outside emails for delivery.

Postfix has a large assortment of configuration options. One of the best ways to learn how to configure Postfix is to read the comments within /etc/postfix/main.cf. Additional resources including information about LDAP and SpamAssassin integration are available online at <http://www.postfix.org/>.

### 23.3.3. Fetchmail

Fetchmail is an MTA which retrieves email from remote servers and delivers it to the local MTA. Many users appreciate the ability to separate the process of downloading their messages located on a remote server from the process of reading and organizing their email in an MUA. Designed with the needs of dial-up users in mind, Fetchmail connects and quickly downloads all of the email messages to the mail spool file using any number of protocols, including POP3 and IMAP. It can even forward email messages to an SMTP server, if necessary.

Fetchmail is configured for each user through the use of a .fetchmailrc file in the user's home directory.

Using preferences in the .fetchmailrc file, Fetchmail checks for email on a remote server and downloads it. It then delivers it to port 25 on the local machine, using the local MTA to place the email in the correct user's spool file. If Procmail is available, it is launched to filter the email and place it in a mailbox so that it can be read by an MUA.

#### 23.3.3.1. Fetchmail Configuration Options

Although it is possible to pass all necessary options on the command line to check for email on a remote server when executing Fetchmail, using a .fetchmailrc file is much easier. Place any desired configuration options in the .fetchmailrc file for those options to be used each time the fetchmail command is issued. It is possible to override these at the time Fetchmail is run by specifying that option on the command line.

A user's .fetchmailrc file contains three classes of configuration options:

* global options — Gives Fetchmail instructions that control the operation of the program or provide settings for every connection that checks for email.
* server options — Specifies necessary information about the server being polled, such as the hostname, as well as preferences for specific email servers, such as the port to check or number of seconds to wait before timing out. These options affect every user using that server.
* user options — Contains information, such as username and password, necessary to authenticate and check for email using a specified email server.

Global options appear at the top of the .fetchmailrc file, followed by one or more server options, each of which designate a different email server that Fetchmail should check. User options follow server options for each user account checking that email server. Like server options, multiple user options may be specified for use with a particular server as well as to check multiple email accounts on the same server.

Server options are called into service in the .fetchmailrc file by the use of a special option verb, poll or skip, that precedes any of the server information. The poll action tells Fetchmail to use this server option when it is run, which checks for email using the specified user options. Any server options after a skip action, however, are not checked unless this server's hostname is specified when Fetchmail is invoked. The skip option is useful when testing configurations in .fetchmailrc because it only checks skipped servers when specifically invoked, and does not affect any currently working configurations.

A sample .fetchmailrc file looks similar to the following example:

set postmaster "user1"

set bouncemail

poll pop.domain.com proto pop3

user 'user1' there with password 'secret' is user1 here

poll mail.domain2.com

user 'user5' there with password 'secret2' is user1 here

user 'user7' there with password 'secret3' is user1 here

In this example, the global options specify that the user is sent email as a last resort (postmaster option) and all email errors are sent to the postmaster instead of the sender (bouncemail option). The set action tells Fetchmail that this line contains a global option. Then, two email servers are specified, one set to check using POP3, the other for trying various protocols to find one that works. Two users are checked using the second server option, but all email found for any user is sent to user1's mail spool. This allows multiple mailboxes to be checked on multiple servers, while appearing in a single MUA inbox. Each user's specific information begins with the user action.

## Note

Users are not required to place their password in the .fetchmailrc file. Omitting the with password '<password>' section causes Fetchmail to ask for a password when it is launched.

Fetchmail has numerous global, server, and local options. Many of these options are rarely used or only apply to very specific situations. The fetchmail man page explains each option in detail, but the most common ones are listed here.

#### 23.3.3.2. Global Options

Each global option should be placed on a single line after a set action.

* daemon *<seconds>* — Specifies daemon-mode, where Fetchmail stays in the background. Replace *<seconds>* with the number of seconds Fetchmail is to wait before polling the server.
* postmaster — Specifies a local user to send mail to in case of delivery problems.
* syslog — Specifies the log file for errors and status messages. By default, this is /var/log/maillog.

#### 23.3.3.3. Server Options

Server options must be placed on their own line in .fetchmailrc after a poll or skip action.

* auth *<auth-type>* — Replace *<auth-type>* with the type of authentication to be used. By default, password authentication is used, but some protocols support other types of authentication, including kerberos\_v5, kerberos\_v4, and ssh. If the any authentication type is used, Fetchmail first tries methods that do not require a password, then methods that mask the password, and finally attempts to send the password unencrypted to authenticate to the server.
* interval *<number>* — Polls the specified server every *<number>* of times that it checks for email on all configured servers. This option is generally used for email servers where the user rarely receives messages.
* port *<port-number>* — Replace *<port-number>* with the port number. This value overrides the default port number for the specified protocol.
* proto *<protocol>* — Replace *<protocol>* with the protocol, such as pop3 or imap, to use when checking for messages on the server.
* timeout *<seconds>* — Replace *<seconds>* with the number of seconds of server inactivity after which Fetchmail gives up on a connection attempt. If this value is not set, a default of 300 seconds is assumed.

#### 23.3.3.4. User Options

User options may be placed on their own lines beneath a server option or on the same line as the server option. In either case, the defined options must follow the user option (defined below).

* fetchall — Orders Fetchmail to download all messages in the queue, including messages that have already been viewed. By default, Fetchmail only pulls down new messages.
* fetchlimit *<number>* — Replace *<number>* with the number of messages to be retrieved before stopping.
* flush — Deletes all previously viewed messages in the queue before retrieving new messages.
* limit *<max-number-bytes>* — Replace *<max-number-bytes>* with the maximum size in bytes that messages are allowed to be when retrieved by Fetchmail. This option is useful with slow network links, when a large message takes too long to download.
* password '*<password>*' — Replace *<password>* with the user's password.
* preconnect "*<command>*" — Replace *<command>* with a command to be executed before retrieving messages for the user.
* postconnect "*<command>*" — Replace *<command>* with a command to be executed after retrieving messages for the user.
* ssl — Activates SSL encryption.
* user "*<username>*" — Replace *<username>* with the username used by Fetchmail to retrieve messages. This option must precede all other user options.

#### 23.3.3.5. Fetchmail Command Options

Most Fetchmail options used on the command line when executing the fetchmail command mirror the .fetchmailrc configuration options. In this way, Fetchmail may be used with or without a configuration file. These options are not used on the command line by most users because it is easier to leave them in the .fetchmailrc file.

There may be times when it is desirable to run the fetchmail command with other options for a particular purpose. It is possible to issue command options to temporarily override a .fetchmailrc setting that is causing an error, as any options specified at the command line override configuration file options.

#### 23.3.3.6. Informational or Debugging Options

Certain options used after the fetchmail command can supply important information.

* --configdump — Displays every possible option based on information from .fetchmailrc and Fetchmail defaults. No email is retrieved for any users when using this option.
* -s — Executes Fetchmail in silent mode, preventing any messages, other than errors, from appearing after the fetchmail command.
* -v — Executes Fetchmail in verbose mode, displaying every communication between Fetchmail and remote email servers.
* -V — Displays detailed version information, lists its global options, and shows settings to be used with each user, including the email protocol and authentication method. No email is retrieved for any users when using this option.

#### 23.3.3.7. Special Options

These options are occasionally useful for overriding defaults often found in the .fetchmailrc file.

* -a — Fetchmail downloads all messages from the remote email server, whether new or previously viewed. By default, Fetchmail only downloads new messages.
* -k — Fetchmail leaves the messages on the remote email server after downloading them. This option overrides the default behavior of deleting messages after downloading them.
* -l *<max-number-bytes>* — Fetchmail does not download any messages over a particular size and leaves them on the remote email server.
* --quit — Quits the Fetchmail daemon process.

More commands and .fetchmailrc options can be found in the fetchmail man page.

**23.4. Mail Transport Agent (MTA) Configuration**

A *Mail Transport Agent* (MTA) is essential for sending email. A *Mail User Agent* (MUA) such as **Evolution**, **Thunderbird**, and **Mutt**, is used to read and compose email. When a user sends an email from an MUA, the message is handed off to the MTA, which sends the message through a series of MTAs until it reaches its destination.

Even if a user does not plan to send email from the system, some automated tasks or system programs might use the /bin/mail command to send email containing log messages to the root user of the local system.

Red Hat Enterprise Linux 5 provides three MTAs: Sendmail, Postfix, and Exim. If all three are installed, sendmail is the default MTA. The **Mail Transport Agent Switcher** allows for the selection of either sendmail, postfix, or exim as the default MTA for the system.

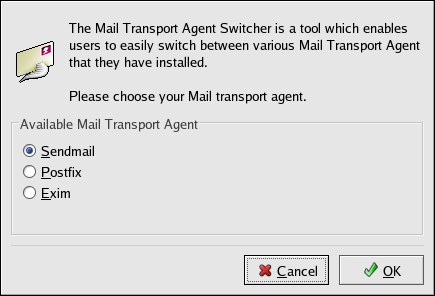
The system-switch-mail RPM package must be installed to use the text-based version of the **Mail Transport Agent Switcher** program. If you want to use the graphical version, the system-switch-mail-gnome package must also be installed.

**Note**

For more information on installing RPM packages, refer to [Part II, “Package Management”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/pt-pkg-management.html).

To start the **Mail Transport Agent Switcher**, select **System** (the main menu on the panel) => **Administration** => **Mail Transport Agent Switcher**, or type the command system-switch-mail at a shell prompt (for example, in an XTerm or GNOME terminal).

The program automatically detects if the X Window System is running. If it is running, the program starts in graphical mode as shown in [Figure 23.1, “Mail Transport Agent Switcher”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-email-switchmail.html#fig-switchmail). If X is not detected, it starts in text-mode. To force **Mail Transport Agent Switcher** to run in text-mode, use the command system-switch-mail-nox.



**Figure 23.1. Mail Transport Agent Switcher**

If you select **OK** to change the MTA, the selected mail daemon is enabled to start at boot time, and the unselected mail daemons are disabled so that they do not start at boot time. The selected mail daemon is started, and any other mail daemon is stopped; thus making the changes take place immediately.

## 23.5. Mail Delivery Agents

Red Hat Enterprise Linux includes two primary MDAs, Procmail and mail. Both of the applications are considered LDAs and both move email from the MTA's spool file into the user's mailbox. However, Procmail provides a robust filtering system.

This section details only Procmail. For information on the mail command, consult its man page.

Procmail delivers and filters email as it is placed in the mail spool file of the localhost. It is powerful, gentle on system resources, and widely used. Procmail can play a critical role in delivering email to be read by email client applications.

Procmail can be invoked in several different ways. Whenever an MTA places an email into the mail spool file, Procmail is launched. Procmail then filters and files the email for the MUA and quits. Alternatively, the MUA can be configured to execute Procmail any time a message is received so that messages are moved into their correct mailboxes. By default, the presence of /etc/procmailrc or of a .procmailrc file (also called an *rc* file) in the user's home directory invokes Procmail whenever an MTA receives a new message.

Whether Procmail acts upon an email message depends upon whether the message matches a specified set of conditions or *recipes* in the rc file. If a message matches a recipe, then the email is placed in a specified file, is deleted, or is otherwise processed.

When Procmail starts, it reads the email message and separates the body from the header information. Next, Procmail looks for /etc/procmailrc and rc files in the /etc/procmailrcs directory for default, system-wide, Procmail environmental variables and recipes. Procmail then searches for a .procmailrc file in the user's home directory. Many users also create additional rc files for Procmail that are referred to within the .procmailrc file in their home directory.

By default, no system-wide rc files exist in the /etc/ directory and no .procmailrc files exist in any user's home directory. Therefore, to use Procmail, each user must construct a .procmailrc file with specific environment variables and rules.

### 23.5.1. Procmail Configuration

The Procmail configuration file contains important environmental variables. These variables specify things such as which messages to sort and what to do with the messages that do not match any recipes.

These environmental variables usually appear at the beginning of .procmailrc in the following format:

*<env-variable>*="*<value>*"

In this example, *<env-variable>* is the name of the variable and *<value>* defines the variable.

There are many environment variables not used by most Procmail users and many of the more important environment variables are already defined by a default value. Most of the time, the following variables are used:

* DEFAULT — Sets the default mailbox where messages that do not match any recipes are placed.

The default DEFAULT value is the same as $ORGMAIL.

* INCLUDERC — Specifies additional rc files containing more recipes for messages to be checked against. This breaks up the Procmail recipe lists into individual files that fulfill different roles, such as blocking spam and managing email lists, that can then be turned off or on by using comment characters in the user's .procmailrc file.

For example, lines in a user's .procmailrc file may look like this:

MAILDIR=$HOME/Msgs

INCLUDERC=$MAILDIR/lists.rc

INCLUDERC=$MAILDIR/spam.rc

If the user wants to turn off Procmail filtering of their email lists but leave spam control in place, they would comment out the first INCLUDERC line with a hash mark character (#).

* LOCKSLEEP — Sets the amount of time, in seconds, between attempts by Procmail to use a particular lockfile. The default is eight seconds.
* LOCKTIMEOUT — Sets the amount of time, in seconds, that must pass after a lockfile was last modified before Procmail assumes that the lockfile is old and can be deleted. The default is 1024 seconds.
* LOGFILE — The file to which any Procmail information or error messages are written.
* MAILDIR — Sets the current working directory for Procmail. If set, all other Procmail paths are relative to this directory.
* ORGMAIL — Specifies the original mailbox, or another place to put the messages if they cannot be placed in the default or recipe-required location.

By default, a value of /var/spool/mail/$LOGNAME is used.

* SUSPEND — Sets the amount of time, in seconds, that Procmail pauses if a necessary resource, such as swap space, is not available.
* SWITCHRC — Allows a user to specify an external file containing additional Procmail recipes, much like the INCLUDERC option, except that recipe checking is actually stopped on the referring configuration file and only the recipes on the SWITCHRC-specified file are used.
* VERBOSE — Causes Procmail to log more information. This option is useful for debugging.

Other important environmental variables are pulled from the shell, such as LOGNAME, which is the login name; HOME, which is the location of the home directory; and SHELL, which is the default shell.

A comprehensive explanation of all environments variables, as well as their default values, is available in the procmailrc man page.

### 23.5.2. Procmail Recipes

New users often find the construction of recipes the most difficult part of learning to use Procmail. To some extent, this is understandable, as recipes do their message matching using *regular expressions*, which is a particular format used to specify qualifications for a matching string. However, regular expressions are not very difficult to construct and even less difficult to understand when read. Additionally, the consistency of the way Procmail recipes are written, regardless of regular expressions, makes it easy to learn by example. To see example Procmail recipes, refer to [Section 23.5.2.5, “Recipe Examples”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-email-mda.html#s3-email-procmail-recipes-examples).

Procmail recipes take the following form:

:0*<flags>: <lockfile-name>*

\* *<special-condition-character>* *<condition-1>*

\* *<special-condition-character>* *<condition-2>*

\* *<special-condition-character>* *<condition-N>*

*<special-action-character><action-to-perform>*

The first two characters in a Procmail recipe are a colon and a zero. Various flags can be placed after the zero to control how Procmail processes the recipe. A colon after the *<flags>* section specifies that a lockfile is created for this message. If a lockfile is created, the name can be specified by replacing *<lockfile-name>*.

A recipe can contain several conditions to match against the message. If it has no conditions, every message matches the recipe. Regular expressions are placed in some conditions to facilitate message matching. If multiple conditions are used, they must all match for the action to be performed. Conditions are checked based on the flags set in the recipe's first line. Optional special characters placed after the \* character can further control the condition.

The *<action-to-perform>* specifies the action taken when the message matches one of the conditions. There can only be one action per recipe. In many cases, the name of a mailbox is used here to direct matching messages into that file, effectively sorting the email. Special action characters may also be used before the action is specified. Refer to [Section 23.5.2.4, “Special Conditions and Actions”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-email-mda.html#s3-email-procmail-recipes-special) for more information.

#### 23.5.2.1. Delivering vs. Non-Delivering Recipes

The action used if the recipe matches a particular message determines whether it is considered a *delivering* or *non-delivering* recipe. A delivering recipe contains an action that writes the message to a file, sends the message to another program, or forwards the message to another email address. A non-delivering recipe covers any other actions, such as a *nesting block*. A nesting block is a set of actions, contained in braces { }, that are performed on messages which match the recipe's conditions. Nesting blocks can be nested inside one another, providing greater control for identifying and performing actions on messages.

When messages match a delivering recipe, Procmail performs the specified action and stops comparing the message against any other recipes. Messages that match non-delivering recipes continue to be compared against other recipes.

#### 23.5.2.2. Flags

Flags are essential to determine how or if a recipe's conditions are compared to a message. The following flags are commonly used:

* A — Specifies that this recipe is only used if the previous recipe without an A or a flag also matched this message.
* a — Specifies that this recipe is only used if the previous recipe with an A or a flag also matched this message and was successfully completed.
* B — Parses the body of the message and looks for matching conditions.
* b — Uses the body in any resulting action, such as writing the message to a file or forwarding it. This is the default behavior.
* c — Generates a carbon copy of the email. This is useful with delivering recipes, since the required action can be performed on the message and a copy of the message can continue being processed in the rc files.
* D — Makes the egrep comparison case-sensitive. By default, the comparison process is not case-sensitive.
* E — While similar to the A flag, the conditions in the recipe are only compared to the message if the immediately preceding the recipe without an E flag did not match. This is comparable to an *else* action.
* e — The recipe is compared to the message only if the action specified in the immediately preceding recipe fails.
* f — Uses the pipe as a filter.
* H — Parses the header of the message and looks for matching conditions. This occurs by default.
* h — Uses the header in a resulting action. This is the default behavior.
* w — Tells Procmail to wait for the specified filter or program to finish, and reports whether or not it was successful before considering the message filtered.
* W — Is identical to w except that "Program failure" messages are suppressed.

For a detailed list of additional flags, refer to the procmailrc man page.

#### 23.5.2.3. Specifying a Local Lockfile

Lockfiles are very useful with Procmail to ensure that more than one process does not try to alter a message simultaneously. Specify a local lockfile by placing a colon (:) after any flags on a recipe's first line. This creates a local lockfile based on the destination file name plus whatever has been set in the LOCKEXT global environment variable.

Alternatively, specify the name of the local lockfile to be used with this recipe after the colon.

#### 23.5.2.4. Special Conditions and Actions

Special characters used before Procmail recipe conditions and actions change the way they are interpreted.

The following characters may be used after the \* character at the beginning of a recipe's condition line:

* ! — In the condition line, this character inverts the condition, causing a match to occur only if the condition does not match the message.
* < — Checks if the message is under a specified number of bytes.
* > — Checks if the message is over a specified number of bytes.

The following characters are used to perform special actions:

* ! — In the action line, this character tells Procmail to forward the message to the specified email addresses.
* $ — Refers to a variable set earlier in the rc file. This is often used to set a common mailbox that is referred to by various recipes.
* | — Starts a specified program to process the message.
* { and } — Constructs a nesting block, used to contain additional recipes to apply to matching messages.

If no special character is used at the beginning of the action line, Procmail assumes that the action line is specifying the mailbox in which to write the message.

#### 23.5.2.5. Recipe Examples

Procmail is an extremely flexible program, but as a result of this flexibility, composing Procmail recipes from scratch can be difficult for new users.

The best way to develop the skills to build Procmail recipe conditions stems from a strong understanding of regular expressions combined with looking at many examples built by others. A thorough explanation of regular expressions is beyond the scope of this section. The structure of Procmail recipes and useful sample Procmail recipes can be found at various places on the Internet (such as <http://www.iki.fi/era/procmail/links.html>). The proper use and adaptation of regular expressions can be derived by viewing these recipe examples. In addition, introductory information about basic regular expression rules can be found in the grep man page.

The following simple examples demonstrate the basic structure of Procmail recipes and can provide the foundation for more intricate constructions.

A basic recipe may not even contain conditions, as is illustrated in the following example:

:0:

new-mail.spool

The first line specifies that a local lockfile is to be created but does not specify a name, so Procmail uses the destination file name and appends the value specified in the LOCKEXT environment variable. No condition is specified, so every message matches this recipe and is placed in the single spool file called new-mail.spool, located within the directory specified by the MAILDIR environment variable. An MUA can then view messages in this file.

A basic recipe, such as this, can be placed at the end of all rc files to direct messages to a default location.

The following example matched messages from a specific email address and throws them away.

:0

\* ^From: spammer@domain.com

/dev/null

With this example, any messages sent by spammer@domain.com are sent to the /dev/null device, deleting them.

## Caution

Be certain that rules are working as intended before sending messages to /dev/null for permanent deletion. If a recipe inadvertently catches unintended messages, and those messages disappear, it becomes difficult to troubleshoot the rule.

A better solution is to point the recipe's action to a special mailbox, which can be checked from time to time to look for false positives. Once satisfied that no messages are accidentally being matched, delete the mailbox and direct the action to send the messages to /dev/null.

The following recipe grabs email sent from a particular mailing list and places it in a specified folder.

:0:

\* ^(From|CC|To).\*tux-lug

tuxlug

Any messages sent from the tux-lug@domain.com mailing list are placed in the tuxlug mailbox automatically for the MUA. Note that the condition in this example matches the message if it has the mailing list's email address on the From, CC, or To lines.

Consult the many Procmail online resources available in [Section 23.7, “Additional Resources”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-email-additional-resources.html) for more detailed and powerful recipes.

#### 23.5.2.6. Spam Filters

Because it is called by Sendmail, Postfix, and Fetchmail upon receiving new emails, Procmail can be used as a powerful tool for combating spam.

This is particularly true when Procmail is used in conjunction with SpamAssassin. When used together, these two applications can quickly identify spam emails, and sort or destroy them.

SpamAssassin uses header analysis, text analysis, blacklists, a spam-tracking database, and self-learning Bayesian spam analysis to quickly and accurately identify and tag spam.

The easiest way for a local user to use SpamAssassin is to place the following line near the top of the ~/.procmailrc file:

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

The /etc/mail/spamassassin/spamassassin-default.rc contains a simple Procmail rule that activates SpamAssassin for all incoming email. If an email is determined to be spam, it is tagged in the header as such and the title is prepended with the following pattern:

\*\*\*\*\*SPAM\*\*\*\*\*

The message body of the email is also prepended with a running tally of what elements caused it to be diagnosed as spam.

To file email tagged as spam, a rule similar to the following can be used:

:0 Hw

\* ^X-Spam-Status: Yes

spam

This rule files all email tagged in the header as spam into a mailbox called spam.

Since SpamAssassin is a Perl script, it may be necessary on busy servers to use the binary SpamAssassin daemon (spamd) and client application (spamc). Configuring SpamAssassin this way, however, requires root access to the host.

To start the spamd daemon, type the following command as root:

/sbin/service spamassassin start

To start the SpamAssassin daemon when the system is booted, use an initscript utility, such as the **Services Configuration Tool** (system-config-services), to turn on the spamassassin service. Refer to for more information about initscript utilities.

To configure Procmail to use the SpamAssassin client application instead of the Perl script, place the following line near the top of the ~/.procmailrc file. For a system-wide configuration, place it in /etc/procmailrc:

INCLUDERC=/etc/mail/spamassassin/spamassassin-spamc.rc

## 23.6. Mail User Agents

There are scores of mail programs available under Red Hat Enterprise Linux. There are full-featured, graphical email client programs, such as **Ximian Evolution**, as well as text-based email programs such as mutt.

The remainder of this section focuses on securing communication between the client and server.

### 23.6.1. Securing Communication

Popular MUAs included with Red Hat Enterprise Linux, such as **Ximian Evolution** and mutt offer SSL-encrypted email sessions.

Like any other service that flows over a network unencrypted, important email information, such as usernames, passwords, and entire messages, may be intercepted and viewed by users on the network. Additionally, since the standard POP and IMAP protocols pass authentication information unencrypted, it is possible for an attacker to gain access to user accounts by collecting usernames and passwords as they are passed over the network.

#### 23.6.1.1. Secure Email Clients

Most Linux MUAs designed to check email on remote servers support SSL encryption. To use SSL when retrieving email, it must be enabled on both the email client and server.

SSL is easy to enable on the client-side, often done with the click of a button in the MUA's configuration window or via an option in the MUA's configuration file. Secure IMAP and POP have known port numbers (993 and 995, respectively) that the MUA uses to authenticate and download messages.

#### 23.6.1.2. Securing Email Client Communications

Offering SSL encryption to IMAP and POP users on the email server is a simple matter.

First, create an SSL certificate. This can be done two ways: by applying to a *Certificate Authority* (*CA*) for an SSL certificate or by creating a self-signed certificate.

## Caution

Self-signed certificates should be used for testing purposes only. Any server used in a production environment should use an SSL certificate granted by a CA.

To create a self-signed SSL certificate for IMAP, change to the /etc/pki/tls/certs/ directory and type the following commands as root:

rm -f cyrus-imapd.pem make cyrus-imapd.pem

Answer all of the questions to complete the process.

To create a self-signed SSL certificate for POP, change to the /etc/pki/tls/certs/ directory, and type the following commands as root:

rm -f ipop3d.pem make ipop3d.pem

Again, answer all of the questions to complete the process.

## Important

Please be sure to remove the default imapd.pem and ipop3d.pem files before issuing each make command.

Once finished, execute the /sbin/service xinetd restart command to restart the xinetd daemon which controls imapd and ipop3d.

Alternatively, the stunnel command can be used as an SSL encryption wrapper around the standard, non-secure daemons, imapd or pop3d.

The stunnel program uses external OpenSSL libraries included with Red Hat Enterprise Linux to provide strong cryptography and protect the connections. It is best to apply to a CA to obtain an SSL certificate, but it is also possible to create a self-signed certificate.

To create a self-signed SSL certificate, change to the /etc/pki/tls/certs/ directory, and type the following command:

make stunnel.pem

Again, answer all of the questions to complete the process.

Once the certificate is generated, it is possible to use the stunnel command to start the imapd mail daemon using the following command:

/usr/sbin/stunnel -d 993 -l /usr/sbin/imapd imapd

Once this command is issued, it is possible to open an IMAP email client and connect to the email server using SSL encryption.

To start the pop3d using the stunnel command, type the following command:

/usr/sbin/stunnel -d 995 -l /usr/sbin/pop3d pop3d

For more information about how to use stunnel, read the stunnel man page or refer to the documents in the /usr/share/doc/stunnel-*<version-number>*/ directory, where *<version-number>* is the version number for stunnel.

**Chapter 16. Berkeley Internet Name Domain (BIND)**

[16.1. Introduction to DNS](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-bind.html#s1-bind-introduction)

[16.2. /etc/named.conf](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-bind-namedconf.html)

[16.3. Zone Files](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-bind-zone.html)

[16.4. Using rndc](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-bind-rndc.html)

[16.5. Advanced Features of BIND](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-bind-features.html)

[16.6. Common Mistakes to Avoid](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-bind-mistakes.html)

[16.7. Additional Resources](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-bind-additional-resources.html)

On most modern networks, including the Internet, users locate other computers by name. This frees users from the daunting task of remembering the numerical network address of network resources. The most effective way to configure a network to allow such name-based connections is to set up a *Domain Name Service* (*DNS*) or a *nameserver*, which resolves hostnames on the network to numerical addresses and vice versa.

This chapter reviews the nameserver included in Red Hat Enterprise Linux and the *Berkeley Internet Name Domain* (*BIND*) DNS server, with an emphasis on the structure of its configuration files and how it may be administered both locally and remotely.

**Note**

BIND is also known as the service named in Red Hat Enterprise Linux. You can manage it via the Services Configuration Tool (system-config-service).

**16.1. Introduction to DNS**

DNS associates hostnames with their respective IP addresses, so that when users want to connect to other machines on the network, they can refer to them by name, without having to remember IP addresses.

Use of DNS and FQDNs also has advantages for system administrators, allowing the flexibility to change the IP address for a host without affecting name-based queries to the machine. Conversely, administrators can shuffle which machines handle a name-based query.

DNS is normally implemented using centralized servers that are authoritative for some domains and refer to other DNS servers for other domains.

When a client host requests information from a nameserver, it usually connects to port 53. The nameserver then attempts to resolve the FQDN based on its resolver library, which may contain authoritative information about the host requested or cached data from an earlier query. If the nameserver does not already have the answer in its resolver library, it queries other nameservers, called *root nameservers*, to determine which nameservers are authoritative for the FQDN in question. Then, with that information, it queries the authoritative nameservers to determine the IP address of the requested host. If a reverse lookup is performed, the same procedure is used, except that the query is made with an unknown IP address rather than a name.

**16.1.1. Nameserver Zones**

On the Internet, the FQDN of a host can be broken down into different sections. These sections are organized into a hierarchy (much like a tree), with a main trunk, primary branches, secondary branches, and so forth. Consider the following FQDN:

bob.sales.example.com

When looking at how an FQDN is resolved to find the IP address that relates to a particular system, read the name from right to left, with each level of the hierarchy divided by periods (.). In this example, com defines the *top level domain* for this FQDN. The name example is a sub-domain under com, while sales is a sub-domain under example. The name furthest to the left, bob, identifies a specific machine hostname.

Except for the hostname, each section is called a *zone*, which defines a specific *namespace*. A namespace controls the naming of the sub-domains to its left. While this example only contains two sub-domains, an FQDN must contain at least one sub-domain but may include many more, depending upon how the namespace is organized.

Zones are defined on authoritative nameservers through the use of *zone files* (which describe the namespace of that zone), the mail servers to be used for a particular domain or sub-domain, and more. Zone files are stored on *primary nameservers* (also called *master nameservers*), which are truly authoritative and where changes are made to the files, and *secondary nameservers* (also called *slave nameservers*), which receive their zone files from the primary nameservers. Any nameserver can be a primary and secondary nameserver for different zones at the same time, and they may also be considered authoritative for multiple zones. It all depends on how the nameserver is configured.

**16.1.2. Nameserver Types**

There are four primary nameserver configuration types:

master

Stores original and authoritative zone records for a namespace, and answers queries about the namespace from other nameservers.

slave

Answers queries from other nameservers concerning namespaces for which it is considered an authority. However, slave nameservers get their namespace information from master nameservers.

caching-only

Offers name-to-IP resolution services, but is not authoritative for any zones. Answers for all resolutions are cached in memory for a fixed period of time, which is specified by the retrieved zone record.

forwarding

Forwards requests to a specific list of nameservers for name resolution. If none of the specified nameservers can perform the resolution, the resolution fails.

A nameserver may be one or more of these types. For example, a nameserver can be a master for some zones, a slave for others, and only offer forwarding resolutions for others.

**16.1.3. BIND as a Nameserver**

BIND performs name resolution services through the /usr/sbin/named daemon. BIND also includes an administration utility called /usr/sbin/rndc. More information about rndc can be found in [Section 16.4, “Using rndc”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-bind-rndc.html).

BIND stores its configuration files in the following locations:

/etc/named.conf

The configuration file for the named daemon

/var/named/ directory

The named working directory which stores zone, statistic, and cache files

**Note**

If you have installed the bind-chroot package, the BIND service will run in the /var/named/chroot environment. All configuration files will be moved there. As such, named.conf will be located in /var/named/chroot/etc/named.conf, and so on.

**Tip**

If you have installed the caching-nameserver package, the default configuration file is /etc/named.caching-nameserver.conf. To override this default configuration, you can create your own custom configuration file in /etc/named.conf. BIND will use the /etc/named.conf custom file instead of the default configuration file after you restart.

**16.2. /etc/named.conf**

The named.conf file is a collection of statements using nested options surrounded by opening and closing ellipse characters, { }. Administrators must be careful when editing named.conf to avoid syntax errors as many seemingly minor errors prevent the named service from starting.

A typical named.conf file is organized similar to the following example:

*<statement-1>* ["*<statement-1-name>*"] [*<statement-1-class>*] {

*<option-1>*;

*<option-2>*;

*<option-N>*; };

*<statement-2>* ["*<statement-2-name>*"] [*<statement-2-class>*] {

*<option-1>*;

*<option-2>*;

*<option-N>*; };

*<statement-N>* ["*<statement-N-name>*"] [*<statement-N-class>*] {

*<option-1>*;

*<option-2>*;

*<option-N>*;

};

**16.2.1. Common Statement Types**

The following types of statements are commonly used in /etc/named.conf:

**16.2.1.1. acl Statement**

The acl statement (or access control statement) defines groups of hosts which can then be permitted or denied access to the nameserver.

An acl statement takes the following form:

acl *<acl-name>* {

*<match-element>*;

[*<match-element>*; ...]

};

In this statement, replace *<acl-name>* with the name of the access control list and replace *<match-element>* with a semi-colon separated list of IP addresses. Most of the time, an individual IP address or IP network notation (such as 10.0.1.0/24) is used to identify the IP addresses within the acl statement.

The following access control lists are already defined as keywords to simplify configuration:

* any — Matches every IP address
* localhost — Matches any IP address in use by the local system
* localnets — Matches any IP address on any network to which the local system is connected
* none — Matches no IP addresses

When used in conjunction with other statements (such as the options statement), acl statements can be very useful in preventing the misuse of a BIND nameserver.

The following example defines two access control lists and uses an options statement to define how they are treated by the nameserver:

acl black-hats {

10.0.2.0/24; 192.168.0.0/24; };

acl red-hats { 10.0.1.0/24; };

options {

blackhole { black-hats; };

allow-query { red-hats; };

allow-recursion { red-hats; };

}

This example contains two access control lists, black-hats and red-hats. Hosts in the black-hats list are denied access to the nameserver, while hosts in the red-hats list are given normal access.

**16.2.1.2. include Statement**

The include statement allows files to be included in a named.conf file. In this way, sensitive configuration data (such as keys) can be placed in a separate file with restrictive permissions.

An include statement takes the following form:

include "*<file-name>*"

In this statement, *<file-name>* is replaced with an absolute path to a file.

**16.2.1.3. options Statement**

The options statement defines global server configuration options and sets defaults for other statements. It can be used to specify the location of the named working directory, the types of queries allowed, and much more.

The options statement takes the following form:

options {

*<option>*;

[*<option>*; ...]

};

In this statement, the *<option>* directives are replaced with a valid option.

The following are commonly used options:

allow-query

Specifies which hosts are allowed to query this nameserver. By default, all hosts are allowed to query. An access control list, or collection of IP addresses or networks, may be used here to allow only particular hosts to query the nameserver.

allow-recursion

Similar to allow-query, this option applies to recursive queries. By default, all hosts are allowed to perform recursive queries on the nameserver.

blackhole

Specifies which hosts are not allowed to query the server.

directory

Specifies the named working directory if different from the default value, /var/named/.

forwarders

Specifies a list of valid IP addresses for nameservers where requests should be forwarded for resolution.

forward

Specifies the forwarding behavior of a forwarders directive.

The following options are accepted:

* first — Specifies that the nameservers listed in the forwarders directive be queried before named attempts to resolve the name itself.
* only — Specifies that named does not attempt name resolution itself in the event that queries to nameservers specified in the forwarders directive fail.

listen-on

Specifies the network interface on which named listens for queries. By default, all interfaces are used.

Using this directive on a DNS server which also acts a gateway, BIND can be configured to only answer queries that originate from one of the networks.

The following is an example of a listen-on directive:

options {

listen-on { 10.0.1.1; };

};

In this example, only requests that arrive from the network interface serving the private network (10.0.1.1) are accepted.

notify

Controls whether named notifies the slave servers when a zone is updated. It accepts the following options:

* yes — Notifies slave servers.
* no — Does not notify slave servers.
* explicit — Only notifies slave servers specified in an also-notify list within a zone statement.

pid-file

Specifies the location of the process ID file created by named.

root-delegation-only

Turns on the enforcement of delegation properties in top-level domains (TLDs) and root zones with an optional exclude list. *Delegation* is the process of dividing a single zone into multiple subzones. In order to create a delegated zone, items known as *NS records* are used. NameServer records (delegation records) announce the authoritative nameservers for a particular zone.

The following root-delegation-only example specifies an exclude list of TLDs from whom undelegated responses are expected and trusted:

options {

root-delegation-only exclude { "ad"; "ar"; "biz"; "cr"; "cu"; "de"; "dm"; "id";

"lu"; "lv"; "md"; "ms"; "museum"; "name"; "no"; "pa";

"pf"; "se"; "sr"; "to"; "tw"; "us"; "uy"; };

};

statistics-file

Specifies an alternate location for statistics files. By default, named statistics are saved to the /var/named/named.stats file.

There are several other options also available, many of which rely upon one another to work properly. Refer to the *BIND 9 Administrator Reference Manual* referenced in [Section 16.7.1, “Installed Documentation”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-bind-additional-resources.html#s2-bind-installed-docs) and the bind.conf man page for more details.

**16.2.1.4. zone Statement**

A zone statement defines the characteristics of a zone, such as the location of its configuration file and zone-specific options. This statement can be used to override the global options statements.

A zone statement takes the following form:

zone *<zone-name>* *<zone-class>* {

*<zone-options>*;

[*<zone-options>*; ...]

};

In this statement, *<zone-name>* is the name of the zone, *<zone-class>* is the optional class of the zone, and *<zone-options>* is a list of options characterizing the zone.

The *<zone-name>* attribute for the zone statement is particularly important. It is the default value assigned for the $ORIGIN directive used within the corresponding zone file located in the /var/named/ directory. The named daemon appends the name of the zone to any non-fully qualified domain name listed in the zone file.

**Note**

If you have installed the caching-nameserver package, the default configuration file will be in /etc/named.rfc1912.zones.

For example, if a zone statement defines the namespace for example.com, use example.com as the *<zone-name>* so it is placed at the end of hostnames within the example.com zone file.

For more information about zone files, refer to [Section 16.3, “Zone Files”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-bind-zone.html).

The most common zone statement options include the following:

allow-query

Specifies the clients that are allowed to request information about this zone. The default is to allow all query requests.

allow-transfer

Specifies the slave servers that are allowed to request a transfer of the zone's information. The default is to allow all transfer requests.

allow-update

Specifies the hosts that are allowed to dynamically update information in their zone. The default is to deny all dynamic update requests.

Be careful when allowing hosts to update information about their zone. Do not enable this option unless the host specified is completely trusted. In general, it is better to have an administrator manually update the records for a zone and reload the named service.

file

Specifies the name of the file in the named working directory that contains the zone's configuration data.

masters

Specifies the IP addresses from which to request authoritative zone information and is used only if the zone is defined as type slave.

notify

Specifies whether or not named notifies the slave servers when a zone is updated. This directive accepts the following options:

* yes — Notifies slave servers.
* no — Does not notify slave servers.
* explicit — Only notifies slave servers specified in an also-notify list within a zone statement.

type

Defines the type of zone.

Below is a list of valid options:

* delegation-only — Enforces the delegation status of infrastructure zones such as COM, NET, or ORG. Any answer that is received without an explicit or implicit delegation is treated as NXDOMAIN. This option is only applicable in TLDs or root zone files used in recursive or caching implementations.
* forward — Forwards all requests for information about this zone to other nameservers.
* hint — A special type of zone used to point to the root nameservers which resolve queries when a zone is not otherwise known. No configuration beyond the default is necessary with a hint zone.
* master — Designates the nameserver as authoritative for this zone. A zone should be set as the master if the zone's configuration files reside on the system.
* slave — Designates the nameserver as a slave server for this zone. Also specifies the IP address of the master nameserver for the zone.

zone-statistics

Configures named to keep statistics concerning this zone, writing them to either the default location (/var/named/named.stats) or the file listed in the statistics-file option in the server statement. Refer to [Section 16.2.2, “Other Statement Types”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-bind-namedconf.html#s2-bind-namedconf-state-other) for more information about the server statement.

**16.2.1.5. Sample zone Statements**

Most changes to the /etc/named.conf file of a master or slave nameserver involves adding, modifying, or deleting zone statements. While these zone statements can contain many options, most nameservers require only a small subset to function efficiently. The following zone statements are very basic examples illustrating a master-slave nameserver relationship.

The following is an example of a zone statement for the primary nameserver hosting example.com (192.168.0.1):

zone "example.com" IN {

type master;

file "example.com.zone";

allow-update { none; };

};

In the statement, the zone is identified as example.com, the type is set to master, and the named service is instructed to read the /var/named/example.com.zone file. It also tells named not to allow any other hosts to update.

A slave server's zone statement for example.com is slightly different from the previous example. For a slave server, the type is set to slave and in place of the allow-update line is a directive telling named the IP address of the master server.

The following is an example slave server zone statement for example.com zone:

zone "example.com" {

type slave;

file "example.com.zone";

masters { 192.168.0.1; };

};

This zone statement configures named on the slave server to query the master server at the 192.168.0.1 IP address for information about the example.com zone. The information that the slave server receives from the master server is saved to the /var/named/example.com.zone file.

**16.2.2. Other Statement Types**

The following is a list of lesser used statement types available within named.conf:

controls

Configures various security requirements necessary to use the rndc command to administer the named service.

Refer to [Section 16.4.1, “Configuring /etc/named.conf”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-bind-rndc.html#s2-bind-rndc-configuration-namedconf) to learn more about how the controls statement is structured and what options are available.

key "*<key-name>*"

Defines a particular key by name. Keys are used to authenticate various actions, such as secure updates or the use of the rndc command. Two options are used with key:

* algorithm *<algorithm-name>* — The type of algorithm used, such as dsa or hmac-md5.
* secret "*<key-value>*" — The encrypted key.

Refer to [Section 16.4.2, “Configuring /etc/rndc.conf”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-bind-rndc.html#s2-bind-rndc-configuration-rndcconf) for instructions on how to write a key statement.

logging

Allows for the use of multiple types of logs, called *channels*. By using the channel option within the logging statement, a customized type of log can be constructed — with its own file name (file), size limit (size), versioning (version), and level of importance (severity). Once a customized channel is defined, a category option is used to categorize the channel and begin logging when named is restarted.

By default, named logs standard messages to the syslog daemon, which places them in /var/log/messages. This occurs because several standard channels are built into BIND with various severity levels, such as default\_syslog (which handles informational logging messages) and default\_debug (which specifically handles debugging messages). A default category, called default, uses the built-in channels to do normal logging without any special configuration.

Customizing the logging process can be a very detailed process and is beyond the scope of this chapter. For information on creating custom BIND logs, refer to the *BIND 9 Administrator Reference Manual* referenced in [Section 16.7.1, “Installed Documentation”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-bind-additional-resources.html#s2-bind-installed-docs).

server

Specifies options that affect how named should respond to remote nameservers, especially with regard to notifications and zone transfers.

The transfer-format option controls whether one resource record is sent with each message (one-answer) or multiple resource records are sent with each message (many-answers). While many-answers is more efficient, only newer BIND nameservers understand it.

trusted-keys

Contains assorted public keys used for secure DNS (DNSSEC). Refer to [Section 16.5.3, “Security”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-bind-features.html#s2-bind-features-security) for more information concerning BIND security.

view "*<view-name>*"

Creates special views depending upon which network the host querying the nameserver is on. This allows some hosts to receive one answer regarding a zone while other hosts receive totally different information. Alternatively, certain zones may only be made available to particular trusted hosts while non-trusted hosts can only make queries for other zones.

Multiple views may be used, but their names must be unique. The match-clients option specifies the IP addresses that apply to a particular view. Any options statement may also be used within a view, overriding the global options already configured for named. Most view statements contain multiple zone statements that apply to the match-clients list. The order in which view statements are listed is important, as the first view statement that matches a particular client's IP address is used.

Refer to [Section 16.5.2, “Multiple Views”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-bind-features.html#s2-bind-features-views) for more information about the view statement.

**16.2.3. Comment Tags**

The following is a list of valid comment tags used within named.conf:

* // — When placed at the beginning of a line, that line is ignored by named.
* # — When placed at the beginning of a line, that line is ignored by named.
* /\* and \*/ — When text is enclosed in these tags, the block of text is ignored by named.

**16.3. Zone Files**

*Zone files* contain information about a namespace and are stored in the named working directory (/var/named/) by default. Each zone file is named according to the file option data in the zone statement, usually in a way that relates to the domain in question and identifies the file as containing zone data, such as example.com.zone.

**Note**

If you have installed the bind-chroot package, the BIND service will run in the /var/named/chroot environment. All configuration files will be moved there. As such, you can find the zone files in /var/named/chroot/var/named.

Each zone file may contain *directives* and *resource records*. Directives tell the nameserver to perform tasks or apply special settings to the zone. Resource records define the parameters of the zone and assign identities to individual hosts. Directives are optional, but resource records are required to provide name service to a zone.

All directives and resource records should be entered on individual lines.

Comments can be placed after semicolon characters (;) in zone files.

**16.3.1. Zone File Directives**

Directives begin with the dollar sign character ($) followed by the name of the directive. They usually appear at the top of the zone file.

The following are commonly used directives:

$INCLUDE

Configures named to include another zone file in this zone file at the place where the directive appears. This allows additional zone settings to be stored apart from the main zone file.

$ORIGIN

Appends the domain name to unqualified records, such as those with the hostname and nothing more.

For example, a zone file may contain the following line:

$ORIGIN example.com.

Any names used in resource records that do not end in a trailing period (.) are appended with example.com.

**Note**

The use of the $ORIGIN directive is unnecessary if the zone is specified in /etc/named.conf because the zone name is used as the value for the $ORIGIN directive by default.

$TTL

Sets the default *Time to Live (TTL)* value for the zone. This is the length of time, in seconds, that a zone resource record is valid. Each resource record can contain its own TTL value, which overrides this directive.

Increasing this value allows remote nameservers to cache the zone information for a longer period of time, reducing the number of queries for the zone and lengthening the amount of time required to proliferate resource record changes.

**16.3.2. Zone File Resource Records**

The primary component of a zone file is its resource records.

There are many types of zone file resource records. The following are used most frequently:

A

This refers to the Address record, which specifies an IP address to assign to a name, as in this example:

*<host>* IN A *<IP-address>*

If the *<host>* value is omitted, then an A record points to a default IP address for the top of the namespace. This system is the target for all non-FQDN requests.

Consider the following A record examples for the example.com zone file:

server1 IN A 10.0.1.3

IN A 10.0.1.5

Requests for example.com are pointed to 10.0.1.3 or 10.0.1.5.

CNAME

This refers to the Canonical Name record, which maps one name to another. This type of record can also be referred to as an *alias record*.

The next example tells named that any requests sent to the *<alias-name>* should point to the host, *<real-name>*. CNAME records are most commonly used to point to services that use a common naming scheme, such as www for Web servers.

*<alias-name>* IN CNAME *<real-name>*

In the following example, an A record binds a hostname to an IP address, while a CNAME record points the commonly used www hostname to it.

server1 IN A 10.0.1.5

www IN CNAME server1

MX

This refers to the Mail eXchange record, which tells where mail sent to a particular namespace controlled by this zone should go.

IN MX *<preference-value>* *<email-server-name>*

Here, the *<preference-value>* allows numerical ranking of the email servers for a namespace, giving preference to some email systems over others. The MX resource record with the lowest *<preference-value>* is preferred over the others. However, multiple email servers can possess the same value to distribute email traffic evenly among them.

The *<email-server-name>* may be a hostname or FQDN.

IN MX 10 mail.example.com.

IN MX 20 mail2.example.com.

In this example, the first mail.example.com email server is preferred to the mail2.example.com email server when receiving email destined for the example.com domain.

NS

This refers to the NameServer record, which announces the authoritative nameservers for a particular zone.

The following illustrates the layout of an NS record:

IN NS *<nameserver-name>*

Here, *<nameserver-name>* should be an FQDN.

Next, two nameservers are listed as authoritative for the domain. It is not important whether these nameservers are slaves or if one is a master; they are both still considered authoritative.

IN NS dns1.example.com.

IN NS dns2.example.com.

PTR

This refers to the PoinTeR record, which is designed to point to another part of the namespace.

PTR records are primarily used for reverse name resolution, as they point IP addresses back to a particular name. Refer to [Section 16.3.4, “Reverse Name Resolution Zone Files”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-bind-zone.html#s2-bind-configuration-zone-reverse) for more examples of PTR records in use.

SOA

This refers to the Start Of Authority resource record, which proclaims important authoritative information about a namespace to the nameserver.

Located after the directives, an SOA resource record is the first resource record in a zone file.

The following shows the basic structure of an SOA resource record:

@ IN SOA *<primary-name-server>* *<hostmaster-email>* (

*<serial-number>*

*<time-to-refresh>*

*<time-to-retry>*

*<time-to-expire>*

*<minimum-TTL> )*

The @ symbol places the $ORIGIN directive (or the zone's name, if the $ORIGIN directive is not set) as the namespace being defined by this SOA resource record. The hostname of the primary nameserver that is authoritative for this domain is the *<primary-name-server>* directive, and the email of the person to contact about this namespace is the *<hostmaster-email>* directive.

The *<serial-number>* directive is a numerical value incremented every time the zone file is altered to indicate it is time for named to reload the zone. The *<time-to-refresh>* directive is the numerical value slave servers use to determine how long to wait before asking the master nameserver if any changes have been made to the zone. The *<serial-number>* directive is a numerical value used by the slave servers to determine if it is using outdated zone data and should therefore refresh it.

The *<time-to-retry>* directive is a numerical value used by slave servers to determine the length of time to wait before issuing a refresh request in the event that the master nameserver is not answering. If the master has not replied to a refresh request before the amount of time specified in the *<time-to-expire>* directive elapses, the slave servers stop responding as an authority for requests concerning that namespace.

The *<minimum-TTL>* directive is the amount of time other nameservers cache the zone's information.

When configuring BIND, all times are specified in seconds. However, it is possible to use abbreviations when specifying units of time other than seconds, such as minutes (M), hours (H), days (D), and weeks (W). The table in [Table 16.1, “Seconds compared to other time units”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-bind-zone.html#tb-bind-seconds) shows an amount of time in seconds and the equivalent time in another format.

| **Seconds** | **Other Time Units** |
| --- | --- |
| 60 | 1M |
| 1800 | 30M |
| 3600 | 1H |
| 10800 | 3H |
| 21600 | 6H |
| 43200 | 12H |
| 86400 | 1D |
| 259200 | 3D |
| 604800 | 1W |
| 31536000 | 365D |

**Table 16.1. Seconds compared to other time units**

The following example illustrates the form an SOA resource record might take when it is populated with real values.

@ IN SOA dns1.example.com. hostmaster.example.com. (

2001062501 ; serial

21600 ; refresh after 6 hours

3600 ; retry after 1 hour

604800 ; expire after 1 week

86400 ) ; minimum TTL of 1 day

**16.3.3. Example Zone File**

Seen individually, directives and resource records can be difficult to grasp. However, when placed together in a single file, they become easier to understand.

The following example shows a very basic zone file.

$ORIGIN example.com.

$TTL 86400

@ IN SOA dns1.example.com. hostmaster.example.com. (

2001062501 ; serial

21600 ; refresh after 6 hours

3600 ; retry after 1 hour

604800 ; expire after 1 week

86400 ) ; minimum TTL of 1 day

IN NS dns1.example.com.

IN NS dns2.example.com.

IN MX 10 mail.example.com.

IN MX 20 mail2.example.com.

dns1 IN A 10.0.1.1

dns2 IN A 10.0.1.2

server1 IN A 10.0.1.5

server2 IN A 10.0.1.6

ftp IN A 10.0.1.3

IN A 10.0.1.4

mail IN CNAME server1

mail2 IN CNAME server2

www IN CNAME server1

In this example, standard directives and SOA values are used. The authoritative nameservers are set as dns1.example.com and dns2.example.com, which have A records that tie them to 10.0.1.1 and 10.0.1.2, respectively.

The email servers configured with the MX records point to server1 and server2 via CNAME records. Since the server1 and server2 names do not end in a trailing period (.), the $ORIGIN domain is placed after them, expanding them to server1.example.com and server2.example.com. Through the related A resource records, their IP addresses can be determined.

FTP and Web services, available at the standard ftp.example.com and www.example.com names, are pointed at the appropriate servers using CNAME records.

This zone file would be called into service with a zone statement in the named.conf similar to the following:

zone "example.com" IN {

type master;

file "example.com.zone";

allow-update { none; };

};

**16.3.4. Reverse Name Resolution Zone Files**

A reverse name resolution zone file is used to translate an IP address in a particular namespace into an FQDN. It looks very similar to a standard zone file, except that PTR resource records are used to link the IP addresses to a fully qualified domain name.

The following illustrates the layout of a PTR record:

*<last-IP-digit>* IN PTR *<FQDN-of-system>*

The *<last-IP-digit>* is the last number in an IP address which points to a particular system's FQDN.

In the following example, IP addresses 10.0.1.1 through 10.0.1.6 are pointed to corresponding FQDNs. It can be located in /var/named/example.com.rr.zone.

$ORIGIN 1.0.10.in-addr.arpa.

$TTL 86400

@ IN SOA dns1.example.com. hostmaster.example.com. (

2001062501 ; serial

21600 ; refresh after 6 hours

3600 ; retry after 1 hour

604800 ; expire after 1 week

86400 ) ; minimum TTL of 1 day

1 IN PTR dns1.example.com.

2 IN PTR dns2.example.com.

5 IN PTR server1.example.com.

6 IN PTR server2.example.com.

3 IN PTR ftp.example.com.

4 IN PTR ftp.example.com.

This zone file would be called into service with a zone statement in the named.conf file similar to the following:

zone "1.0.10.in-addr.arpa" IN {

type master;

file "example.com.rr.zone";

allow-update { none; };

};

There is very little difference between this example and a standard zone statement, except for the zone name. Note that a reverse name resolution zone requires the first three blocks of the IP address reversed followed by .in-addr.arpa. This allows the single block of IP numbers used in the reverse name resolution zone file to be associated with the zone.

## 16.4. Using rndc

BIND includes a utility called rndc which allows command line administration of the named daemon from the localhost or a remote host.

In order to prevent unauthorized access to the named daemon, BIND uses a shared secret key authentication method to grant privileges to hosts. This means an identical key must be present in both /etc/named.conf and the rndc configuration file, /etc/rndc.conf.

## Note

If you have installed the bind-chroot package, the BIND service will run in the /var/named/chroot environment. All configuration files will be moved there. As such, the rndc.conf file is located in /var/named/chroot/etc/rndc.conf.

Note that since the rndc utility does not run in a chroot environment, /etc/rndc.conf is a symlink to /var/named/chroot/etc/rndc.conf.

### 16.4.1. Configuring /etc/named.conf

In order for rndc to connect to a named service, there must be a controls statement in the BIND server's /etc/named.conf file.

The controls statement, shown in the following example, allows rndc to connect from the localhost.

controls {

inet 127.0.0.1 allow { localhost; }

keys { *<key-name>*; };

};

This statement tells named to listen on the default TCP port 953 of the loopback address and allow rndc commands coming from the localhost, if the proper key is given. The *<key-name>* specifies a name in the key statement within the /etc/named.conf file. The next example illustrates a sample key statement.

key "*<key-name>*" {

algorithm hmac-md5;

secret "*<key-value>*";

};

In this case, the *<key-value>* uses the HMAC-MD5 algorithm. Use the following command to generate keys using the HMAC-MD5 algorithm:

dnssec-keygen -a hmac-md5 -b *<bit-length>* -n HOST *<key-file-name>*

A key with at least a 256-bit length is a good idea. The actual key that should be placed in the *<key-value>* area can be found in the *<key-file-name>* file generated by this command.

## Warning

Because /etc/named.conf is world-readable, it is advisable to place the key statement in a separate file, readable only by root, and then use an include statement to reference it. For example:

include "/etc/rndc.key";

### 16.4.2. Configuring /etc/rndc.conf

The key is the most important statement in /etc/rndc.conf.

key "*<key-name>*" {

algorithm hmac-md5;

secret "*<key-value>*";

};

The *<key-name>* and *<key-value>* should be exactly the same as their settings in /etc/named.conf.

To match the keys specified in the target server's /etc/named.conf, add the following lines to /etc/rndc.conf.

options {

default-server localhost;

default-key "*<key-name>*";

};

This directive sets a global default key. However, the rndc configuration file can also specify different keys for different servers, as in the following example:

server localhost {

key "*<key-name>*";

};

## Important

Make sure that only the root user can read or write to the /etc/rndc.conf file.

For more information about the /etc/rndc.conf file, refer to the rndc.conf man page.

### 16.4.3. Command Line Options

An rndc command takes the following form:

rndc *<options>* *<command>* *<command-options>*

When executing rndc on a properly configured localhost, the following commands are available:

* halt — Stops the named service immediately.
* querylog — Logs all queries made to this nameserver.
* refresh — Refreshes the nameserver's database.
* reload — Reloads the zone files but keeps all other previously cached responses. This command also allows changes to zone files without losing all stored name resolutions.

If changes made only affect a specific zone, reload only that specific zone by adding the name of the zone after the reload command.

* stats — Dumps the current named statistics to the /var/named/named.stats file.
* stop — Stops the server gracefully, saving any dynamic update and *Incremental Zone Transfers* (*IXFR*) data before exiting.

Occasionally, it may be necessary to override the default settings in the /etc/rndc.conf file. The following options are available:

* -c *<configuration-file>* — Specifies the alternate location of a configuration file.
* -p *<port-number>* — Specifies a port number to use for the rndc connection other than the default port 953.
* -s *<server>* — Specifies a server other than the default-server listed in /etc/rndc.conf.
* -y *<key-name>* — Specifies a key other than the default-key option in /etc/rndc.conf.

Additional information about these options can be found in the rndc man page.

**16.5. Advanced Features of BIND**

Most BIND implementations only use named to provide name resolution services or to act as an authority for a particular domain or sub-domain. However, BIND version 9 has a number of advanced features that allow for a more secure and efficient DNS service.

**Caution**

Some of these advanced features, such as DNSSEC, TSIG, and IXFR (which are defined in the following section), should only be used in network environments with nameservers that support the features. If the network environment includes non-BIND or older BIND nameservers, verify that each advanced feature is supported before attempting to use it.

All of the features mentioned are discussed in greater detail in the *BIND 9 Administrator Reference Manual* referenced in [Section 16.7.1, “Installed Documentation”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-bind-additional-resources.html#s2-bind-installed-docs).

**16.5.1. DNS Protocol Enhancements**

BIND supports Incremental Zone Transfers (IXFR), where a slave nameserver only downloads the updated portions of a zone modified on a master nameserver. The standard transfer process requires that the entire zone be transferred to each slave nameserver for even the smallest change. For very popular domains with very lengthy zone files and many slave nameservers, IXFR makes the notification and update process much less resource-intensive.

Note that IXFR is only available when using *dynamic updating* to make changes to master zone records. If manually editing zone files to make changes, Automatic Zone Transfer (AXFR) is used. More information on dynamic updating is available in the *BIND 9 Administrator Reference Manual* referenced in [Section 16.7.1, “Installed Documentation”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-bind-additional-resources.html#s2-bind-installed-docs).

**16.5.2. Multiple Views**

Through the use of the view statement in named.conf, BIND can present different information depending on which network a request originates from.

This is primarily used to deny sensitive DNS entries from clients outside of the local network, while allowing queries from clients inside the local network.

The view statement uses the match-clients option to match IP addresses or entire networks and give them special options and zone data.

**16.5.3. Security**

BIND supports a number of different methods to protect the updating and transfer of zones, on both master and slave nameservers:

*DNSSEC*

Short for *DNS SECurity*, this feature allows for zones to be cryptographically signed with a *zone key*.

In this way, the information about a specific zone can be verified as coming from a nameserver that has signed it with a particular private key, as long as the recipient has that nameserver's public key.

BIND version 9 also supports the SIG(0) public/private key method of message authentication.

*TSIG*

Short for *Transaction SIGnatures*, this feature allows a transfer from master to slave only after verifying that a shared secret key exists on both nameservers.

This feature strengthens the standard IP address-based method of transfer authorization. An attacker would not only need to have access to the IP address to transfer the zone, but they would also need to know the secret key.

BIND version 9 also supports *TKEY*, which is another shared secret key method of authorizing zone transfers.

**Chapter 17. OpenSSH**

[17.1. Features of SSH](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-openssh.html#s1-ssh-intro)

[17.2. SSH Protocol Versions](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ssh-version.html)

[17.3. Event Sequence of an SSH Connection](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ssh-conn.html)

[17.4. Configuring an OpenSSH Server](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-openssh-server-config.html)

[17.5. OpenSSH Configuration Files](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ssh-configfiles.html)

[17.6. Configuring an OpenSSH Client](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-openssh-client-config.html)

[17.7. More Than a Secure Shell](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ssh-beyondshell.html)

[17.8. Additional Resources](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-openssh-additional-resources.html)

SSH™ (or *S*ecure *SH*ell) is a protocol which facilitates secure communications between two systems using a client/server architecture and allows users to log into server host systems remotely. Unlike other remote communication protocols, such as FTP or Telnet, SSH encrypts the login session, rendering the connection difficult for intruders to collect unencrypted passwords.

SSH is designed to replace older, less secure terminal applications used to log into remote hosts, such as telnet or rsh. A related program called scp replaces older programs designed to copy files between hosts, such as rcp. Because these older applications do not encrypt passwords transmitted between the client and the server, avoid them whenever possible. Using secure methods to log into remote systems decreases the risks for both the client system and the remote host.

**17.1. Features of SSH**

The SSH protocol provides the following safeguards:

* After an initial connection, the client can verify that it is connecting to the same server it had connected to previously.
* The client transmits its authentication information to the server using strong, 128-bit encryption.
* All data sent and received during a session is transferred using 128-bit encryption, making intercepted transmissions extremely difficult to decrypt and read.
* The client can forward X11[[5](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-openssh.html#ftn.id2917427)] applications from the server. This technique, called *X11 forwarding*, provides a secure means to use graphical applications over a network.

Because the SSH protocol encrypts everything it sends and receives, it can be used to secure otherwise insecure protocols. Using a technique called *port forwarding*, an SSH server can become a conduit to securing otherwise insecure protocols, like POP, and increasing overall system and data security.

The OpenSSH server and client can also be configured to create a tunnel similar to a virtual private network for traffic between server and client machines.

Finally, OpenSSH servers and clients can be configured to authenticate using the GSSAPI implementation of the Kerberos network authentication protocol. For more information on configuring Kerberos authentication services, refer to [Section 42.6, “Kerberos”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-kerberos.html).

Red Hat Enterprise Linux includes the general OpenSSH package (openssh) as well as the OpenSSH server (openssh-server) and client (openssh-clients) packages. Note, the OpenSSH packages require the OpenSSL package (openssl) which installs several important cryptographic libraries, enabling OpenSSH to provide encrypted communications.

**17.1.1. Why Use SSH?**

Nefarious computer users have a variety of tools at their disposal enabling them to disrupt, intercept, and re-route network traffic in an effort to gain access to a system. In general terms, these threats can be categorized as follows:

* *Interception of communication between two systems* — In this scenario, the attacker can be somewhere on the network between the communicating parties, copying any information passed between them. The attacker may intercept and keep the information, or alter the information and send it on to the intended recipient.

This attack can be mounted through the use of a packet sniffer — a common network utility.

* *Impersonation of a particular host* — Using this strategy, an attacker's system is configured to pose as the intended recipient of a transmission. If this strategy works, the user's system remains unaware that it is communicating with the wrong host.

This attack can be mounted through techniques known as DNS poisoning[[6](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-openssh.html#ftn.id3055656)] or IP spoofing[[7](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-openssh.html#ftn.id3066457)].

Both techniques intercept potentially sensitive information and, if the interception is made for hostile reasons, the results can be disastrous.

If SSH is used for remote shell login and file copying, these security threats can be greatly diminished. This is because the SSH client and server use digital signatures to verify their identity. Additionally, all communication between the client and server systems is encrypted. Attempts to spoof the identity of either side of a communication does not work, since each packet is encrypted using a key known only by the local and remote systems.

## 17.2. SSH Protocol Versions

The SSH protocol allows any client and server programs built to the protocol's specifications to communicate securely and to be used interchangeably.

Two varieties of SSH (version 1 and version 2) currently exist. The OpenSSH suite under Red Hat Enterprise Linux uses SSH version 2 which has an enhanced key exchange algorithm not vulnerable to the exploit in version 1. However, the OpenSSH suite does support version 1 connections.

## Important

It is recommended that only SSH version 2-compatible servers and clients are used whenever possible.

## 17.3. Event Sequence of an SSH Connection

The following series of events help protect the integrity of SSH communication between two hosts.

1. A cryptographic handshake is made so that the client can verify that it is communicating with the correct server.
2. The transport layer of the connection between the client and remote host is encrypted using a symmetric cipher.
3. The client authenticates itself to the server.
4. The remote client interacts with the remote host over the encrypted connection.

### 17.3.1. Transport Layer

The primary role of the transport layer is to facilitate safe and secure communication between the two hosts at the time of authentication and during subsequent communication. The transport layer accomplishes this by handling the encryption and decryption of data, and by providing integrity protection of data packets as they are sent and received. The transport layer also provides compression, speeding the transfer of information.

Once an SSH client contacts a server, key information is exchanged so that the two systems can correctly construct the transport layer. The following steps occur during this exchange:

* Keys are exchanged
* The public key encryption algorithm is determined
* The symmetric encryption algorithm is determined
* The message authentication algorithm is determined
* The hash algorithm is determined

During the key exchange, the server identifies itself to the client with a unique *host key*. If the client has never communicated with this particular server before, the server's host key is unknown to the client and it does not connect. OpenSSH gets around this problem by accepting the server's host key. This is done after the user is notified and has both accepted and verified the new host key. In subsequent connections, the server's host key is checked against the saved version on the client, providing confidence that the client is indeed communicating with the intended server. If, in the future, the host key no longer matches, the user must remove the client's saved version before a connection can occur.

## Caution

It is possible for an attacker to masquerade as an SSH server during the initial contact since the local system does not know the difference between the intended server and a false one set up by an attacker. To help prevent this, verify the integrity of a new SSH server by contacting the server administrator before connecting for the first time or in the event of a host key mismatch.

SSH is designed to work with almost any kind of public key algorithm or encoding format. After an initial key exchange creates a hash value used for exchanges and a shared secret value, the two systems immediately begin calculating new keys and algorithms to protect authentication and future data sent over the connection.

After a certain amount of data has been transmitted using a given key and algorithm (the exact amount depends on the SSH implementation), another key exchange occurs, generating another set of hash values and a new shared secret value. Even if an attacker is able to determine the hash and shared secret value, this information is only useful for a limited period of time.

### 17.3.2. Authentication

Once the transport layer has constructed a secure tunnel to pass information between the two systems, the server tells the client the different authentication methods supported, such as using a private key-encoded signature or typing a password. The client then tries to authenticate itself to the server using one of these supported methods.

SSH servers and clients can be configured to allow different types of authentication, which gives each side the optimal amount of control. The server can decide which encryption methods it supports based on its security model, and the client can choose the order of authentication methods to attempt from the available options.

### 17.3.3. Channels

After a successful authentication over the SSH transport layer, multiple channels are opened via a technique called *multiplexing*[[8](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ssh-conn.html#ftn.id3065353)]. Each of these channels handles communication for different terminal sessions and for forwarded X11 sessions.

Both clients and servers can create a new channel. Each channel is then assigned a different number on each end of the connection. When the client attempts to open a new channel, the clients sends the channel number along with the request. This information is stored by the server and is used to direct communication to that channel. This is done so that different types of sessions do not affect one another and so that when a given session ends, its channel can be closed without disrupting the primary SSH connection.

Channels also support *flow-control*, which allows them to send and receive data in an orderly fashion. In this way, data is not sent over the channel until the client receives a message that the channel is open.

The client and server negotiate the characteristics of each channel automatically, depending on the type of service the client requests and the way the user is connected to the network. This allows great flexibility in handling different types of remote connections without having to change the basic infrastructure of the protocol.

## 17.4. Configuring an OpenSSH Server

To run an OpenSSH server, you must first make sure that you have the proper RPM packages installed. The openssh-server package is required and is dependent on the openssh package.

The OpenSSH daemon uses the configuration file /etc/ssh/sshd\_config. The default configuration file should be sufficient for most purposes. If you want to configure the daemon in ways not provided by the default sshd\_config, read the sshd man page for a list of the keywords that can be defined in the configuration file.

To start the OpenSSH service, use the command /sbin/service sshd start. To stop the OpenSSH server, use the command /sbin/service sshd stop. If you want the daemon to start automatically at boot time, refer to [Chapter 15, *Controlling Access to Services*](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-services.html) for information on how to manage services.

If you reinstall, the reinstalled system creates a new set of identification keys. Any clients who had connected to the system with any of the OpenSSH tools before the reinstall will see the following message:

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

@ WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED! @

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!

Someone could be eavesdropping on you right now (man-in-the-middle attack)!

It is also possible that the RSA host key has just been changed.

If you want to keep the host keys generated for the system, backup the /etc/ssh/ssh\_host\*key\* files and restore them after the reinstall. This process retains the system's identity, and when clients try to connect to the system after the reinstall, they will not receive the warning message.

### 17.4.1. Requiring SSH for Remote Connections

For SSH to be truly effective, using insecure connection protocols, such as Telnet and FTP, should be prohibited. Otherwise, a user's password may be protected using SSH for one session, only to be captured later while logging in using Telnet.

Some services to disable include:

* telnet
* rsh
* rlogin
* vsftpd

To disable insecure connection methods to the system, use the command line program chkconfig, the ncurses-based program **/usr/sbin/ntsysv**, or the **Services Configuration Tool** (system-config-services) graphical application. All of these tools require root level access.

## 17.5. OpenSSH Configuration Files

OpenSSH has two different sets of configuration files: one for client programs (ssh, scp, and sftp) and one for the server daemon (sshd).

System-wide SSH configuration information is stored in the /etc/ssh/ directory:

* moduli — Contains Diffie-Hellman groups used for the Diffie-Hellman key exchange which is critical for constructing a secure transport layer. When keys are exchanged at the beginning of an SSH session, a shared, secret value is created which cannot be determined by either party alone. This value is then used to provide host authentication.
* ssh\_config — The system-wide default SSH client configuration file. It is overridden if one is also present in the user's home directory (~/.ssh/config).
* sshd\_config — The configuration file for the sshd daemon.
* ssh\_host\_dsa\_key — The DSA private key used by the sshd daemon.
* ssh\_host\_dsa\_key.pub — The DSA public key used by the sshd daemon.
* ssh\_host\_key — The RSA private key used by the sshd daemon for version 1 of the SSH protocol.
* ssh\_host\_key.pub — The RSA public key used by the sshd daemon for version 1 of the SSH protocol.
* ssh\_host\_rsa\_key — The RSA private key used by the sshd daemon for version 2 of the SSH protocol.
* ssh\_host\_rsa\_key.pub — The RSA public key used by the sshd for version 2 of the SSH protocol.

User-specific SSH configuration information is stored in the user's home directory within the ~/.ssh/ directory:

* authorized\_keys — This file holds a list of authorized public keys for servers. When the client connects to a server, the server authenticates the client by checking its signed public key stored within this file.
* id\_dsa — Contains the DSA private key of the user.
* id\_dsa.pub — The DSA public key of the user.
* id\_rsa — The RSA private key used by ssh for version 2 of the SSH protocol.
* id\_rsa.pub — The RSA public key used by ssh for version 2 of the SSH protocol
* identity — The RSA private key used by ssh for version 1 of the SSH protocol.
* identity.pub — The RSA public key used by ssh for version 1 of the SSH protocol.
* known\_hosts — This file contains DSA host keys of SSH servers accessed by the user. This file is very important for ensuring that the SSH client is connecting the correct SSH server.

## Important

If an SSH server's host key has changed, the client notifies the user that the connection cannot proceed until the server's host key is deleted from the known\_hosts file using a text editor. Before doing this, however, contact the system administrator of the SSH server to verify the server is not compromised.

## 17.6. Configuring an OpenSSH Client

To connect to an OpenSSH server from a client machine, you must have the openssh-clients and openssh packages installed on the client machine.

### 17.6.1. Using the ssh Command

The ssh command is a secure replacement for the rlogin, rsh, and telnet commands. It allows you to log in to a remote machine as well as execute commands on a remote machine.

Logging in to a remote machine with ssh is similar to using telnet. To log in to a remote machine named penguin.example.net, type the following command at a shell prompt:

ssh penguin.example.net

The first time you ssh to a remote machine, you will see a message similar to the following:

The authenticity of host 'penguin.example.net' can't be established.

DSA key fingerprint is 94:68:3a:3a:bc:f3:9a:9b:01:5d:b3:07:38:e2:11:0c.

Are you sure you want to continue connecting (yes/no)?

Type **yes** to continue. This will add the server to your list of known hosts (~/.ssh/known\_hosts) as seen in the following message:

Warning: Permanently added 'penguin.example.net' (RSA) to the list of known hosts.

Next, you will see a prompt asking for your password for the remote machine. After entering your password, you will be at a shell prompt for the remote machine. If you do not specify a username the username that you are logged in as on the local client machine is passed to the remote machine. If you want to specify a different username, use the following command:

ssh *username*@penguin.example.net

You can also use the syntax ssh -l *username* penguin.example.net.

The ssh command can be used to execute a command on the remote machine without logging in to a shell prompt. The syntax is ssh *hostname* *command*. For example, if you want to execute the command ls /usr/share/doc on the remote machine penguin.example.net, type the following command at a shell prompt:

ssh penguin.example.net ls /usr/share/doc

After you enter the correct password, the contents of the remote directory /usr/share/doc will be displayed, and you will return to your local shell prompt.

### 17.6.2. Using the scp Command

The scp command can be used to transfer files between machines over a secure, encrypted connection. It is similar to rcp.

The general syntax to transfer a local file to a remote system is as follows:

scp *<localfile>* *username@tohostname:<remotefile>*

The *<localfile>* specifies the source including path to the file, such as /var/log/maillog. The *<remotefile>* specifies the destination, which can be a new filename such as /tmp/hostname-maillog. For the remote system, if you do not have a preceding /, the path will be relative to the home directory of *username*, typically /home/username/.

To transfer the local file shadowman to the home directory of your account on penguin.example.net, type the following at a shell prompt (replace *username* with your username):

scp shadowman *username*@penguin.example.net:shadowman

This will transfer the local file shadowman to /home/*username*/shadowman on penguin.example.net. Alternately, you can leave off the final shadowman in the scp command.

The general syntax to transfer a remote file to the local system is as follows:

scp *username@tohostname:<remotefile>* *<newlocalfile>*

The *<remotefile>* specifies the source including path, and *<newlocalfile>* specifies the destination including path.

Multiple files can be specified as the source files. For example, to transfer the contents of the directory downloads/ to an existing directory called uploads/ on the remote machine penguin.example.net, type the following at a shell prompt:

scp downloads/\* *username*@penguin.example.net:uploads/

### 17.6.3. Using the sftp Command

The sftp utility can be used to open a secure, interactive FTP session. It is similar to ftp except that it uses a secure, encrypted connection. The general syntax is sftp *username@hostname.com*. Once authenticated, you can use a set of commands similar to those used by FTP. Refer to the sftp man page for a list of these commands. To read the man page, execute the command man sftp at a shell prompt. The sftp utility is only available in OpenSSH version 2.5.0p1 and higher.

## 17.7. More Than a Secure Shell

A secure command line interface is just the beginning of the many ways SSH can be used. Given the proper amount of bandwidth, X11 sessions can be directed over an SSH channel. Or, by using TCP/IP forwarding, previously insecure port connections between systems can be mapped to specific SSH channels.

### 17.7.1. X11 Forwarding

Opening an X11 session over an SSH connection is as easy as connecting to the SSH server using the -Y option and running an X program on a local machine.

ssh -Y <user>@example.com

When an X program is run from the secure shell prompt, the SSH client and server create a new secure channel, and the X program data is sent over that channel to the client machine transparently.

X11 forwarding can be very useful. For example, X11 forwarding can be used to create a secure, interactive session of the **Printer Configuration Tool**. To do this, connect to the server using **ssh** and type:

system-config-printer &

After supplying the root password for the server, the **Printer Configuration Tool** appears and allows the remote user to safely configure printing on the remote system.

### 17.7.2. Port Forwarding

SSH can secure otherwise insecure TCP/IP protocols via port forwarding. When using this technique, the SSH server becomes an encrypted conduit to the SSH client.

Port forwarding works by mapping a local port on the client to a remote port on the server. SSH can map any port from the server to any port on the client; port numbers do not need to match for this technique to work.

To create a TCP/IP port forwarding channel which listens for connections on the localhost, use the following command:

ssh -L *local-port*:*remote-hostname*:*remote-port* *username*@*hostname*

## Note

Setting up port forwarding to listen on ports below 1024 requires root level access.

To check email on a server called mail.example.com using POP3 through an encrypted connection, use the following command:

ssh -L 1100:mail.example.com:110 mail.example.com

Once the port forwarding channel is in place between the client machine and the mail server, direct a POP3 mail client to use port 1100 on the localhost to check for new mail. Any requests sent to port 1100 on the client system are directed securely to the mail.example.com server.

If mail.example.com is not running an SSH server, but another machine on the same network is, SSH can still be used to secure part of the connection. However, a slightly different command is necessary:

ssh -L 1100:mail.example.com:110 other.example.com

In this example, POP3 requests from port 1100 on the client machine are forwarded through the SSH connection on port 22 to the SSH server, other.example.com. Then, other.example.com connects to port 110 on mail.example.com to check for new mail. Note, when using this technique only the connection between the client system and other.example.com SSH server is secure.

Port forwarding can also be used to get information securely through network firewalls. If the firewall is configured to allow SSH traffic via its standard port (22) but blocks access to other ports, a connection between two hosts using the blocked ports is still possible by redirecting their communication over an established SSH connection.

## Note

Using port forwarding to forward connections in this manner allows any user on the client system to connect to that service. If the client system becomes compromised, the attacker also has access to forwarded services.

System administrators concerned about port forwarding can disable this functionality on the server by specifying a No parameter for the AllowTcpForwarding line in /etc/ssh/sshd\_config and restarting the sshd service.

### 17.7.3. Generating Key Pairs

If you do not want to enter your password every time you use ssh, scp, or sftp to connect to a remote machine, you can generate an authorization key pair.

Keys must be generated for each user. To generate keys for a user, use the following steps as the user who wants to connect to remote machines. If you complete the steps as root, only root will be able to use the keys.

Starting with OpenSSH version 3.0, ~/.ssh/authorized\_keys2, ~/.ssh/known\_hosts2, and /etc/ssh\_known\_hosts2 are obsolete. SSH Protocol 1 and 2 share the ~/.ssh/authorized\_keys, ~/.ssh/known\_hosts, and /etc/ssh/ssh\_known\_hosts files.

Red Hat Enterprise Linux 5 uses SSH Protocol 2 and RSA keys by default.

## Tip

If you reinstall and want to save your generated key pair, backup the .ssh directory in your home directory. After reinstalling, copy this directory back to your home directory. This process can be done for all users on your system, including root.

#### 17.7.3.1. Generating an RSA Key Pair for Version 2

Use the following steps to generate an RSA key pair for version 2 of the SSH protocol. This is the default starting with OpenSSH 2.9.

1. To generate an RSA key pair to work with version 2 of the protocol, type the following command at a shell prompt:
2. ssh-keygen -t rsa

Accept the default file location of ~/.ssh/id\_rsa. Enter a passphrase different from your account password and confirm it by entering it again.

The public key is written to ~/.ssh/id\_rsa.pub. The private key is written to ~/.ssh/id\_rsa. Never distribute your private key to anyone.

1. Change the permissions of the .ssh directory using the following command:
2. chmod 755 ~/.ssh
3. Copy the contents of ~/.ssh/id\_rsa.pub into the file ~/.ssh/authorized\_keys on the machine to which you want to connect. If the file ~/.ssh/authorized\_keys exist, append the contents of the file ~/.ssh/id\_rsa.pub to the file ~/.ssh/authorized\_keys on the other machine.
4. Change the permissions of the authorized\_keys file using the following command:
5. chmod 644 ~/.ssh/authorized\_keys
6. If you are running GNOME or are running in a graphical desktop with GTK2+ libraries installed, skip to [Section 17.7.3.4, “Configuring ssh-agent with a GUI”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ssh-beyondshell.html#s3-openssh-ssh-agent-with-gnome). If you are not running the X Window System, skip to [Section 17.7.3.5, “Configuring ssh-agent”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ssh-beyondshell.html#s3-openssh-config-ssh-agent).

#### 17.7.3.2. Generating a DSA Key Pair for Version 2

Use the following steps to generate a DSA key pair for version 2 of the SSH Protocol.

1. To generate a DSA key pair to work with version 2 of the protocol, type the following command at a shell prompt:
2. ssh-keygen -t dsa

Accept the default file location of ~/.ssh/id\_dsa. Enter a passphrase different from your account password and confirm it by entering it again.

## Tip

A passphrase is a string of words and characters used to authenticate a user. Passphrases differ from passwords in that you can use spaces or tabs in the passphrase. Passphrases are generally longer than passwords because they are usually phrases instead of a single word.

The public key is written to ~/.ssh/id\_dsa.pub. The private key is written to ~/.ssh/id\_dsa. It is important never to give anyone the private key.

1. Change the permissions of the .ssh directory with the following command:
2. chmod 755 ~/.ssh
3. Copy the contents of ~/.ssh/id\_dsa.pub into the file ~/.ssh/authorized\_keys on the machine to which you want to connect. If the file ~/.ssh/authorized\_keys exist, append the contents of the file ~/.ssh/id\_dsa.pub to the file ~/.ssh/authorized\_keys on the other machine.
4. Change the permissions of the authorized\_keys file using the following command:
5. chmod 644 ~/.ssh/authorized\_keys
6. If you are running GNOME or a graphical desktop environment with the GTK2+ libraries installed, skip to [Section 17.7.3.4, “Configuring ssh-agent with a GUI”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ssh-beyondshell.html#s3-openssh-ssh-agent-with-gnome). If you are not running the X Window System, skip to [Section 17.7.3.5, “Configuring ssh-agent”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ssh-beyondshell.html#s3-openssh-config-ssh-agent).

#### 17.7.3.3. Generating an RSA Key Pair for Version 1.3 and 1.5

Use the following steps to generate an RSA key pair, which is used by version 1 of the SSH Protocol. If you are only connecting between systems that use DSA, you do not need an RSA version 1.3 or RSA version 1.5 key pair.

1. To generate an RSA (for version 1.3 and 1.5 protocol) key pair, type the following command at a shell prompt:
2. ssh-keygen -t rsa1

Accept the default file location (~/.ssh/identity). Enter a passphrase different from your account password. Confirm the passphrase by entering it again.

The public key is written to ~/.ssh/identity.pub. The private key is written to ~/.ssh/identity. Do not give anyone the private key.

1. Change the permissions of your .ssh directory and your key with the commands chmod 755 ~/.ssh and chmod 644 ~/.ssh/identity.pub.
2. Copy the contents of ~/.ssh/identity.pub into the file ~/.ssh/authorized\_keys on the machine to which you wish to connect. If the file ~/.ssh/authorized\_keys does not exist, you can copy the file ~/.ssh/identity.pub to the file ~/.ssh/authorized\_keys on the remote machine.
3. If you are running GNOME, skip to [Section 17.7.3.4, “Configuring ssh-agent with a GUI”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ssh-beyondshell.html#s3-openssh-ssh-agent-with-gnome). If you are not running GNOME, skip to [Section 17.7.3.5, “Configuring ssh-agent”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ssh-beyondshell.html#s3-openssh-config-ssh-agent).

#### 17.7.3.4. Configuring ssh-agent with a GUI

The ssh-agent utility can be used to save your passphrase so that you do not have to enter it each time you initiate an ssh or scp connection. If you are using GNOME, the gnome-ssh-askpass package contains the application used to prompt you for your passphrase when you log in to GNOME and save it until you log out of GNOME. You will not have to enter your password or passphrase for any ssh or scp connection made during that GNOME session. If you are not using GNOME, refer to [Section 17.7.3.5, “Configuring ssh-agent”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ssh-beyondshell.html#s3-openssh-config-ssh-agent).

To save your passphrase during your GNOME session, follow the following steps:

1. You will need to have the package gnome-ssh-askpass installed; you can use the command rpm -q openssh-askpass to determine if it is installed or not. If it is not installed, install it from your Red Hat Enterprise Linux CD-ROM set, from a Red Hat FTP mirror site, or using Red Hat Network.
2. Select **Main Menu Button** (on the Panel) => **Preferences** => **More Preferences** => **Sessions**, and click on the **Startup Programs** tab. Click **Add** and enter **/usr/bin/ssh-add** in the **Startup Command** text area. Set it a priority to a number higher than any existing commands to ensure that it is executed last. A good priority number for ssh-add is 70 or higher. The higher the priority number, the lower the priority. If you have other programs listed, this one should have the lowest priority. Click **Close** to exit the program.
3. Log out and then log back into GNOME; in other words, restart X. After GNOME is started, a dialog box will appear prompting you for your passphrase(s). Enter the passphrase requested. If you have both DSA and RSA key pairs configured, you will be prompted for both. From this point on, you should not be prompted for a password by ssh, scp, or sftp.

#### 17.7.3.5. Configuring ssh-agent

The ssh-agent can be used to store your passphrase so that you do not have to enter it each time you make a ssh or scp connection. If you are not running the X Window System, follow these steps from a shell prompt. If you are running GNOME but you do not want to configure it to prompt you for your passphrase when you log in (refer to [Section 17.7.3.4, “Configuring ssh-agent with a GUI”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ssh-beyondshell.html#s3-openssh-ssh-agent-with-gnome)), this procedure will work in a terminal window, such as an XTerm. If you are running X but not GNOME, this procedure will work in a terminal window. However, your passphrase will only be remembered for that terminal window; it is not a global setting.

1. At a shell prompt, type the following command:
2. exec /usr/bin/ssh-agent $SHELL
3. Then type the command:
4. ssh-add

and enter your passphrase(s). If you have more than one key pair configured, you will be prompted for each one.

1. When you log out, your passphrase(s) will be forgotten. You must execute these two commands each time you log in to a virtual console or open a terminal window.

**Chapter 18. Network File System (NFS)**

[18.1. How It Works](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-nfs.html#s1-nfs-how)

[18.2. NFS Client Configuration](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-nfs-client-config.html)

[18.3. autofs](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-nfs-client-config-autofs.html)

[18.4. Common NFS Mount Options](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-nfs-client-config-options.html)

[18.5. Starting and Stopping NFS](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-nfs-start.html)

[18.6. NFS Server Configuration](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-nfs-server-export.html)

[18.7. The /etc/exports Configuration File](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-nfs-server-config-exports.html)

[18.8. Securing NFS](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-nfs-security.html)

[18.9. NFS and portmap](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s2-nfs-methodology-portmap.html)

[18.10. Using NFS over TCP](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-nfs-tcp.html)

[18.11. Additional Resources](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-nfs-additional-resources.html)

A *Network File System* (*NFS*) allows remote hosts to mount file systems over a network and interact with those file systems as though they are mounted locally. This enables system administrators to consolidate resources onto centralized servers on the network.

This chapter focuses on fundamental NFS concepts and supplemental information.

**18.1. How It Works**

Currently, there are three versions of NFS. NFS version 2 (NFSv2) is older and is widely supported. NFS version 3 (NFSv3) has more features, including 64bit file handles, Safe Async writes and more robust error handling. NFS version 4 (NFSv4) works through firewalls and on the Internet, no longer requires portmapper, supports ACLs, and utilizes stateful operations. Red Hat Enterprise Linux supports NFSv2, NFSv3, and NFSv4 clients, and when mounting a file system via NFS, Red Hat Enterprise Linux uses NFSv3 by default, if the server supports it.

All versions of NFS can use *Transmission Control Protocol* (*TCP*) running over an IP network, with NFSv4 requiring it. NFSv2 and NFSv3 can use the *User Datagram Protocol* (*UDP*) running over an IP network to provide a stateless network connection between the client and server.

When using NFSv2 or NFSv3 with UDP, the stateless UDP connection under normal conditions has less Protocol overhead than TCP which can translate into better performance on very clean, non-congested networks. The NFS server sends the client a file handle after the client is authorized to access the shared volume. This file handle is an opaque object stored on the server's side and is passed along with RPC requests from the client. The NFS server can be restarted without affecting the clients and the cookie remains intact. However, because UDP is stateless, if the server goes down unexpectedly, UDP clients continue to saturate the network with requests for the server. For this reason, TCP is the preferred protocol when connecting to an NFS server.

NFSv4 has no interaction with portmapper, rpc.mountd, rpc.lockd, and rpc.statd, since protocol support has been incorporated into the v4 protocol. NFSv4 listens on the well known TCP port (2049) which eliminates the need for the portmapper interaction. The mounting and locking protocols have been incorpated into the V4 protocol which eliminates the need for interaction with rpc.mountd and rpc.lockd.

**Note**

TCP is the default transport protocol for NFS under Red Hat Enterprise Linux. UDP can be used for compatibility purposes as needed, but is not recommended for wide usage.

All the RPC/NFS daemon have a '-p' command line option that can set the port, making firewall configuration easier.

After the client is granted access by TCP wrappers, the NFS server refers to its configuration file, /etc/exports, to determine whether the client is allowed to access any of the exported file systems. Once access is granted, all file and directory operations are available to the user.

**Important**

In order for NFS to work with a default installation of Red Hat Enterprise Linux with a firewall enabled, IPTables with the default TCP port 2049 must be configured. Without proper IPTables configuration, NFS does not function properly.

The NFS initialization script and rpc.nfsd process now allow binding to any specified port during system start up. However, this can be error prone if the port is unavailable or conflicts with another daemon.

**18.1.1. Required Services**

Red Hat Enterprise Linux uses a combination of kernel-level support and daemon processes to provide NFS file sharing. All NFS versions rely on *Remote Procedure Calls* (*RPC*) between clients and servers. RPC services under Linux are controlled by the portmap service. To share or mount NFS file systems, the following services work together, depending on which version of NFS is implemented:

* nfs — (/sbin/service nfs start) starts the NFS server and the appropriate RPC processes to service requests for shared NFS file systems.
* nfslock — (/sbin/service nfslock start) is a mandatory service that starts the appropriate RPC processes to allow NFS clients to lock files on the server.
* portmap — accepts port reservations from local RPC services. These ports are then made available (or advertised) so the corresponding remote RPC services access them. portmap responds to requests for RPC services and sets up connections to the requested RPC service. This is not used with NFSv4.

The following RPC processes facilitate NFS services:

* rpc.mountd — This process receives mount requests from NFS clients and verifies the requested file system is currently exported. This process is started automatically by the nfs service and does not require user configuration. This is not used with NFSv4.
* rpc.nfsd — Allows explicit NFS versions and protocols the server advertises to be defined. It works with the Linux kernel to meet the dynamic demands of NFS clients, such as providing server threads each time an NFS client connects. This process corresponds to the nfs service.
* rpc.lockd — allows NFS clients to lock files on the server. If rpc.lockd is not started, file locking will fail. rpc.lockd implements the *Network Lock Manager (NLM)* protocol. This process corresponds to the nfslock service. This is not used with NFSv4.
* rpc.statd — This process implements the *Network Status Monitor (NSM)* RPC protocol which notifies NFS clients when an NFS server is restarted without being gracefully brought down. This process is started automatically by the nfslock service and does not require user configuration. This is not used with NFSv4.
* rpc.rquotad — This process provides user quota information for remote users. This process is started automatically by the nfs service and does not require user configuration.
* rpc.idmapd — This process provides NFSv4 client and server upcalls which map between on-the-wire NFSv4 names (which are strings in the form of user@domain) and local UIDs and GIDs. For idmapd to function with NFSv4, the /etc/idmapd.conf must be configured. This service is required for use with NFSv4.

## 18.2. NFS Client Configuration

NFS shares are mounted on the client side using the mount command. The format of the command is as follows:

mount -t *<nfs-type>* -o *<options>* *<host>*:*</remote/export>* *</local/directory>*

Replace *<nfs-type>* with either nfs for NFSv2 or NFSv3 servers, or nfs4 for NFSv4 servers. Replace *<options>* with a comma separated list of options for the NFS file system (refer to [Section 18.4, “Common NFS Mount Options”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-nfs-client-config-options.html) for details). Replace *<host>* with the remote host, *</remote/export>* with the remote directory being mounted, and *</local/directory>* with the local directory where the remote file system is to be mounted.

Refer to the mount man page for more details.

If accessing an NFS share by manually issuing the mount command, the file system must be remounted manually after the system is rebooted. Red Hat Enterprise Linux offers two methods for mounting remote file systems automatically at boot time: the /etc/fstab file or the autofs service.

### 18.2.1. Mounting NFS File Systems using /etc/fstab

An alternate way to mount an NFS share from another machine is to add a line to the /etc/fstab file. The line must state the hostname of the NFS server, the directory on the server being exported, and the directory on the local machine where the NFS share is to be mounted. You must be root to modify the /etc/fstab file.

The general syntax for the line in /etc/fstab is as follows:

server:/usr/local/pub /pub nfs rsize=8192,wsize=8192,timeo=14,intr

The mount point /pub must exist on the client machine before this command can be executed. After adding this line to /etc/fstab on the client system, type the command mount /pub at a shell prompt, and the mount point /pub is mounted from the server.

The /etc/fstab file is referenced by the netfs service at boot time, so lines referencing NFS shares have the same effect as manually typing the mount command during the boot process.

A sample /etc/fstab line to mount an NFS export looks like the following example:

*<server>*:*</remote/export>* *</local/directory>* *<nfs-type>* *<options>* 0 0

Replace *<server>* with the hostname, IP address, or fully qualified domain name of the server exporting the file system.

Replace *</remote/export>* with the path to the exported directory.

Replace *</local/directory>* with the local file system on which the exported directory is mounted. This mount point must exist before /etc/fstab is read or the mount fails.

Replace *<nfs-type>* with either nfs for NFSv2 or NFSv3 servers, or nfs4 for NFSv4 servers.

Replace *<options>* with a comma separated list of options for the NFS file system (refer to [Section 18.4, “Common NFS Mount Options”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-nfs-client-config-options.html) for details). Refer to the fstab man page for additional information.

## 18.3. autofs

One drawback to using /etc/fstab is that, regardless of how infrequently a user accesses the NFS mounted file system, the system must dedicate resources to keep the mounted file system in place. This is not a problem with one or two mounts, but when the system is maintaining mounts to many systems at one time, overall system performance can be affected. An alternative to /etc/fstab is to use the kernel-based automount utility. An automounter consists of two components. One is a kernel module that implements a file system, while the other is a user-space daemon that performs all of the other functions. The automount utility can mount and unmount NFS file systems automatically (on demand mounting) therefore saving system resources. The automount utility can be used to mount other file systems including AFS, SMBFS, CIFS and local file systems.

autofs uses /etc/auto.master (master map) as its default primary configuration file. This can be changed to use another supported network source and name using the autofs configuration (in /etc/sysconfig/autofs) in conjunction with the Name Service Switch mechanism. An instance of the version 4 daemon was run for each mount point configured in the master map and so it could be run manually from the command line for any given mount point. This is not possible with version 5 because it uses a single daemon to manage all configured mount points, so all automounts must be configured in the master map. This is in line with the usual requirements of other industry standard automounters. Mount point, hostname, exported directory, and options can all be specified in a set of files (or other supported network sources) rather than configuring them manually for each host. Please ensure that you have the autofs package installed if you wish to use this service.

### 18.3.1. What's new in autofs version 5?

Direct map support

Autofs direct maps provide a mechanism to automatically mount file systems at arbitrary points in the file system hierarchy. A direct map is denoted by a mount point of "/-" in the master map. Entries in a direct map contain an absolute path name as a key (instead of the relative path names used in indirect maps).

Lazy mount and unmount support

Multimount map entries describe a hierarchy of mount points under a single key. A good example of this is the "-hosts" map, commonly used for automounting all exports from a host under "/net/<host>" as a multi-mount map entry. When using the "-hosts" map, an 'ls' of "/net/<host>" will mount autofs trigger mounts for each export from <host> and mount and expire them as they are accessed. This can greatly reduce the number of active mounts needed when accessing a server with a large number of exports.

Enhanced LDAP support

The Lightweight Directory Access Protocol, or LDAP, support in autofs version 5 has been enhanced in several ways with respect to autofs version 4. The autofs configuration file (/etc/sysconfig/autofs) provides a mechanism to specify the autofs schema that a site implements, thus precluding the need to determine this via trial and error in the application itself. In addition, authenticated binds to the LDAP server are now supported, using most mechanisms supported by the common LDAP server implementations. A new configuration file has been added for this support: /etc/autofs\_ldap\_auth.conf. The default configuration file is self-documenting, and uses an XML format.

Proper use of the Name Service Switch (nsswitch) configuration.

The Name Service Switch configuration file exists to provide a means of determining from where specific configuration data comes. The reason for this configuration is to allow administrators the flexibility of using the back-end database of choice, while maintaining a uniform software interface to access the data. While the version 4 automounter is becoming increasingly better at handling the name service switch configuration, it is still not complete. Autofs version 5, on the other hand, is a complete implementation. See the manual page for nsswitch.conf for more information on the supported syntax of this file. Please note that not all nss databases are valid map sources and the parser will reject ones that are invalid. Valid sources are files, yp, nis, nisplus, ldap and hesiod.

Multiple master map entries per autofs mount point

One thing that is frequently used but not yet mentioned is the handling of multiple master map entries for the direct mount point "/-". The map keys for each entry are merged and behave as one map.

An example is seen in the connectathon test maps for the direct mounts below:

/- /tmp/auto\_dcthon

/- /tmp/auto\_test3\_direct

/- /tmp/auto\_test4\_direct

### 18.3.2. autofs Configuration

The primary configuration file for the automounter is /etc/auto.master, also referred to as the master map which may be changed as described in the introduction section above. The master map lists autofs-controlled mount points on the system, and their corresponding configuration files or network sources known as automount maps. The format of the master map is as follows:

<mount-point> <map-name> <options>

where:

* mount-point is the autofs mount point e.g /home.
* map-name is the name of a map source which contains a list of mount points, and the file system location from which those mount points should be mounted. The syntax for a map entry is described below.
* options if supplied, will apply to all entries in the given map provided they don't themselves have options specified. This behavior is different from autofs version 4 where the options where cumulative. This has been changed to meet our primary goal of mixed environment compatibility.

The following is a sample /etc/auto.master file:

$ cat /etc/auto.master

/home /etc/auto.misc

The general format of maps is similar to the master map, however the "options" appear between the mount point and the location instead of at the end of the entry as in the master map:

<mount-point> [<options>] <location>

where:

* <mount-point> is the autofs mount point. This can be a single directory name for an indirect mount or the full path of the mount point for direct mounts. Each direct and indirect map entry key (<mount-point> above) may be followed by a space separated list of offset directories (sub directory names each beginning with a "/") making them what is known as a mutli-mount entry.
* <options> if supplied, are the mount options for the map entries that do not specify their own options.
* <location> is the file system location such as a local file system path (preceded with the Sun map format escape character ":" for map names beginning with "/"), an NFS file system or other valid file system location.

The following is a sample map file:

$ cat /etc/auto.misc

payroll -fstype=nfs personnel:/dev/hda3

sales -fstype=ext3 :/dev/hda4

The first column in a map file indicates the autofs mount point (sales and payroll from the server called personnel). The second column indicates the options for the autofs mount while the third column indicates the source of the mount. Following the above configuration, the autofs mount points will be /home/payroll and /home/sales. The -fstype= option is often omitted and is generally not needed for correct operation.

The automounter will create the directories if they do not exist. If the directories exist before the automounter was started, the automounter will not remove them when it exits. You can start or restart the automount daemon by issuing the following command:

$/sbin/service autofs start

or

$/sbin/service autofs restart

Using the above configuration, if a process requires access to an autofs unmounted directory such as /home/payroll/2006/July.sxc, the automount daemon automatically mounts the directory. If a timeout is specified, the directory will automatically be unmounted if the directory is not accessed for the timeout period.

You can view the status of the automount daemon by issuing the following command in your terminal:

$/sbin/service/autofs status

### 18.3.3. autofs Common Tasks

#### 18.3.3.1. Overriding or augmenting site configuration files

It can be useful to override site defaults for a specific mount point on a client system. For example, assuming that the automounter maps are stored in NIS and the /etc/nsswitch.conf file has the following directive:

automount: files nis

and the NIS auto.master map file contains the following:

/home auto.home

Also assume the NIS auto.home map contains the following:

beth fileserver.example.com:/export/home/beth

joe fileserver.example.com:/export/home/joe

\* fileserver.example.com:/export/home/&

and the file map /etc/auto.home does not exist.

For the above example, lets assume that the client system needs to mount home directories from a different server. In this case, the client will need to use the following /etc/auto.master map:

/home /etc/auto.home2

+auto.master

And the /etc/auto.home2 map contains the entry:

\* labserver.example.com:/export/home/&

Because only the first occurrence of a mount point is processed, /home will contain the contents of /etc/auto.home2 instead of the NIS auto.home map.

Alternatively, if you just want to augment the site-wide

auto.home

map with a few entries, create a /etc/auto.home file map, and in it put your new entries and at the end, include the NIS auto.home map. Then the /etc/auto.home file map might look similar to:

mydir someserver:/export/mydir

+auto.home

Given the NIS auto.home map listed above, an ls of /home would now give:

$ ls /home

beth joe mydir

This last example works as expected because autofs knows not to include the contents of a file map of the same name as the one it is reading and so moves on to the next map source in the nsswitch configuration.

#### 18.3.3.2. Using LDAP to Store Automounter Maps

LDAP client libraries must be installed on all systems which are to retrieve automounter maps from LDAP. On RHEL 5, the openldap package should be installed automatically as a dependency of the automounter. To configure LDAP access, modify /etc/openldap/ldap.conf. Ensure that BASE and URI are set appropriately for your site. Please also ensure that the schema is set in the configuration.

The most recently established schema for storing automount maps in LDAP is described by rfc2307bis. To use this schema it is necessary to set it in the autofs configuration (/etc/sysconfig/autofs) by removing the comment characters from the schema definition. For example:

DEFAULT\_MAP\_OBJECT\_CLASS="automountMap"

DEFAULT\_ENTRY\_OBJECT\_CLASS="automount"

DEFAULT\_MAP\_ATTRIBUTE="automountMapName"

DEFAULT\_ENTRY\_ATTRIBUTE="automountKey"

DEFAULT\_VALUE\_ATTRIBUTE="automountInformation"

Ensure that these are the only schema entries not commented in the configuration. Please also note that the automountKey replaces the cn attribute in the rfc2307bis schema. An LDIF of a sample configuration is described below:

# extended LDIF

#

# LDAPv3

# base <> with scope subtree

# filter: (&(objectclass=automountMap)(automountMapName=auto.master))

# requesting: ALL

#

# auto.master, example.com

dn: automountMapName=auto.master,dc=example,dc=com

objectClass: top

objectClass: automountMap

automountMapName: auto.master

# extended LDIF

#

# LDAPv3

# base <automountMapName=auto.master,dc=example,dc=com> with scope subtree

# filter: (objectclass=automount)

# requesting: ALL

#

# /home, auto.master, example.com

dn: automountMapName=auto.master,dc=example,dc=com

objectClass: automount

cn: /home

automountKey: /home

automountInformation: auto.home

# extended LDIF

#

# LDAPv3

# base <> with scope subtree

# filter: (&(objectclass=automountMap)(automountMapName=auto.home))

# requesting: ALL

#

# auto.home, example.com

dn: automountMapName=auto.home,dc=example,dc=com

objectClass: automountMap

automountMapName: auto.home

# extended LDIF

#

# LDAPv3

# base <automountMapName=auto.home,dc=example,dc=com> with scope subtree

# filter: (objectclass=automount)

# requesting: ALL

#

# foo, auto.home, example.com

dn: automountKey=foo,automountMapName=auto.home,dc=example,dc=com

objectClass: automount

automountKey: foo

automountInformation: filer.example.com:/export/foo

# /, auto.home, example.com

dn: automountKey=/,automountMapName=auto.home,dc=example,dc=com

objectClass: automount

automountKey: /

automountInformation: filer.example.com:/export/&

#### 18.3.3.3. Adapting Autofs v4 Maps To Autofs v5

##### v4 Multi-map entries

Autofs version 4 introduced the notion of a multi-map entry in the master map. A multi-map entry is of the form:

<mount-point> <maptype1> <mapname1> <options1> -- <maptype2> <mapname2> <options2> -- ...

Any number of maps can be combined into a single map in this manner. This feature is no longer present in v5. This is because Version 5 supports included maps which can be used to attain the same results. Consider the following multi-map example: /home file /etc/auto.home -- nis auto.home

This can be replaced by the following configuration for v5:

/etc/nsswitch.conf must list:

automount: files nis

/etc/auto.master should contain:

/home auto.home

/etc/auto.home should contain:

<entries for the home directory>

+auto.home

In this way, the entries from /etc/auto.home and the nis auto.home map are combined.

##### Multiple master maps

In autofs version 4, it is possible to merge the contents of master maps from each source, such as files, nis, hesiod, and LDAP. The version 4 automounter looks for a master map for each of the sources listed in /etc/nsswitch.conf. The map is read if it exists and its contents are merged into one large auto.master map.

In version 5, this is no longer the behaviour. Only the first master map found from the list of sources in nsswitch.conf is consulted. If it is desirable to merge the contents of multiple master maps, included maps can be used. Consider the following example:

/etc/nsswitch.conf:

automount: files nis

/etc/auto.master:

/home /etc/auto.home

+auto.master

The above configuration will merge the contents of the file-based auto.master and the NIS-based auto.master. However, because included map entries are only allowed in file maps, there is no way to include both an NIS auto.master and an LDAP auto.master.

This limitation can be overcome by creating a master maps that have a different name in the source. In the example above if we had an LDAP master map named auto.master.ldap we could also add "+auto.master.ldap" to the file based master map and provided that "ldap" is listed as a source in our nsswitch configuration it would also be included.

## 18.4. Common NFS Mount Options

Beyond mounting a file system via NFS on a remote host, other options can be specified at the time of the mount to make it easier to use. These options can be used with manual mount commands, /etc/fstab settings, and autofs.

The following are options commonly used for NFS mounts:

* fsid=num — Forces the file handle and file attributes settings on the wire to be *num*, instead of a number derived from the major and minor number of the block device on the mounted file system. The value 0 has special meaning when used with NFSv4. NFSv4 has a concept of a root of the overall exported file system. The export point exported with fsid=0 is used as this root.
* hard or soft — Specifies whether the program using a file via an NFS connection should stop and wait (hard) for the server to come back online, if the host serving the exported file system is unavailable, or if it should report an error (soft).

If hard is specified, the user cannot terminate the process waiting for the NFS communication to resume unless the intr option is also specified.

If soft is specified, the user can set an additional timeo=*<value>* option, where *<value>* specifies the number of seconds to pass before the error is reported.

## Note

Using soft mounts is not recommended as they can generate I/O errors in very congested networks or when using a very busy server.

* intr — Allows NFS requests to be interrupted if the server goes down or cannot be reached.
* nfsvers=2 or nfsvers=3 — Specifies which version of the NFS protocol to use. This is useful for hosts that run multiple NFS servers. If no version is specified, NFS uses the highest supported version by the kernel and mount command. This option is not supported with NFSv4 and should not be used.
* noacl — Turns off all ACL processing. This may be needed when interfacing with older versions of Red Hat Enterprise Linux, Red Hat Linux, or Solaris, since the most recent ACL technology is not compatible with older systems.
* nolock — Disables file locking. This setting is occasionally required when connecting to older NFS servers.
* noexec — Prevents execution of binaries on mounted file systems. This is useful if the system is mounting a non-Linux file system via NFS containing incompatible binaries.
* nosuid — Disables set-user-identifier or set-group-identifier bits. This prevents remote users from gaining higher privileges by running a setuid program.
* port=num — Specifies the numeric value of the NFS server port. If *num* is 0 (the default), then mount queries the remote host's portmapper for the port number to use. If the remote host's NFS daemon is not registered with its portmapper, the standard NFS port number of TCP 2049 is used instead.
* rsize=num and wsize=num — These settings speed up NFS communication for reads (rsize) and writes (wsize) by setting a larger data block size, in bytes, to be transferred at one time. Be careful when changing these values; some older Linux kernels and network cards do not work well with larger block sizes. For NFSv2 or NFSv3, the default values for both parameters is set to 8192. For NFSv4, the default values for both parameters is set to 32768.
* sec=mode — Specifies the type of security to utilize when authenticating an NFS connection.

sec=sys is the default setting, which uses local UNIX UIDs and GIDs by means of AUTH\_SYS to authenticate NFS operations.

sec=krb5 uses Kerberos V5 instead of local UNIX UIDs and GIDs to authenticate users.

sec=krb5i uses Kerberos V5 for user authentication and performs integrity checking of NFS operations using secure checksums to prevent data tampering.

sec=krb5p uses Kerberos V5 for user authentication, integrity checking, and encrypts NFS traffic to prevent traffic sniffing. This is the most secure setting, but it also has the most performance overhead involved.

* tcp — Specifies for the NFS mount to use the TCP protocol.
* udp — Specifies for the NFS mount to use the UDP protocol.

Many more options are listed on the mount and nfs man page

## 18.5. Starting and Stopping NFS

To run an NFS server, the portmap service must be running. To verify that portmap is active, type the following command as root:

/sbin/service portmap status

If the portmap service is running, then the nfs service can be started. To start an NFS server, as root type:

/sbin/service nfs start

## Note

nfslock also has to be started for both the NFS client and server to function properly. To start NFS locking as root type: /sbin/service nfslock start. If NFS is set to start at boot, please ensure that nfslock also starts by running chkconfig --list nfslock. If nfslock is not set to on, this implies that you will need to manually run the /sbin/service nfslock start each time the computer starts. To set nfslock to automatically start on boot, type the following command in a terminal chkconfig nfslock on.

To stop the server, as root, type:

/sbin/service nfs stop

The restart option is a shorthand way of stopping and then starting NFS. This is the most efficient way to make configuration changes take effect after editing the configuration file for NFS.

To restart the server, as root, type:

/sbin/service nfs restart

The condrestart (*conditional restart*) option only starts nfs if it is currently running. This option is useful for scripts, because it does not start the daemon if it is not running.

To conditionally restart the server, as root, type:

/sbin/service nfs condrestart

To reload the NFS server configuration file without restarting the service, as root, type:

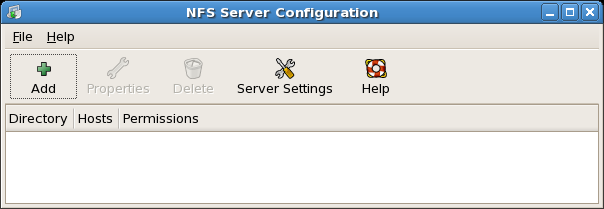
/sbin/service nfs reload

By default, the nfs service does not start automatically at boot time. To configure the NFS to start up at boot time, use an initscript utility, such as /sbin/chkconfig, **/usr/sbin/ntsysv**, or the **Services Configuration Tool** program. Refer to [Chapter 15, *Controlling Access to Services*](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-services.html) for more information regarding these tools.

**18.6. NFS Server Configuration**

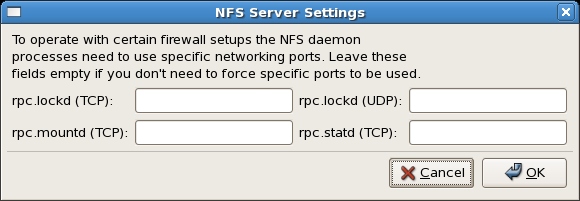
There are three ways to configure an NFS server under Red Hat Enterprise Linux: using the **NFS Server Configuration Tool** (system-config-nfs), manually editing its configuration file (/etc/exports), or using the /usr/sbin/exportfs command.

To use the NFS Server Configuration Tool, you must be running X Windows, have root privileges, and have the system-config-nfs RPM package installed. To start the application, click on **System** => **Administration** => **Server Settings** => **NFS**. You can also type the command system-config-nfs in a terminal. The NFS Server Configuration tool window is illustrated below.



**Figure 18.1. NFS Server Configuration Tool**

Based on certain firewall settings, you may need to configure the NFS daemon processes to use specific networking ports. The NFS server settings allows you to specify the ports for each process instead of using the random ports assigned by the portmapper. You can set the NFS Server settings by clicking on the **Server Settings** button. The figure below illustrates the NFS Server Settings window.



**Figure 18.2. NFS Server Settings**

**18.6.1. Exporting or Sharing NFS File Systems**

Sharing or serving files from an NFS server is known as exporting the directories. The **NFS Server Configuration Tool** can be used to configure a system as an NFS server.

To add an NFS share, click the **Add** button. The dialog box shown in [Figure 18.3, “Add Share”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-nfs-server-export.html#fig-add-share) appears.

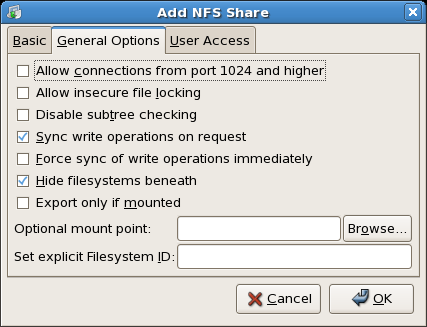
The **Basic** tab requires the following information:

* **Directory** — Specify the directory to share, such as /tmp.
* **Host(s)** — Specify the host(s) with which to share the directory. Refer to [Section 18.6.3, “Hostname Formats”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-nfs-server-export.html#s2-nfs-hostname-formats) for an explanation of possible formats.
* **Basic permissions** — Specify whether the directory should have read-only or read/write permissions.



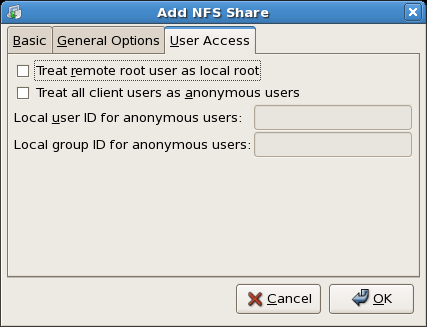
**Figure 18.3. Add Share**

The **General Options** tab allows the following options to be configured:



**Figure 18.4. NFS General Options**

* **Allow connections from port 1024 and higher** — Services started on port numbers less than 1024 must be started as root. Select this option to allow the NFS service to be started by a user other than root. This option corresponds to insecure.
* **Allow insecure file locking** — Do not require a lock request. This option corresponds to insecure\_locks.
* **Disable subtree checking** — If a subdirectory of a file system is exported, but the entire file system is not exported, the server checks to see if the requested file is in the subdirectory exported. This check is called *subtree checking*. Select this option to disable subtree checking. If the entire file system is exported, selecting to disable subtree checking can increase the transfer rate. This option corresponds to no\_subtree\_check.
* **Sync write operations on request** — Enabled by default, this option does not allow the server to reply to requests before the changes made by the request are written to the disk. This option corresponds to sync. If this is not selected, the async option is used.
  + **Force sync of write operations immediately** — Do not delay writing to disk. This option corresponds to no\_wdelay.
* **Hide filesystems beneath** turns the nohide option on or off. When the nohide option is off, nested directories are revealed. The clients can therefore navigate through a filesystem from the parent without noticing any changes.
* **Export only if mounted** sets the mountpoint option which allows a directory to be exported only if it has been mounted.
* **Optional Mount Point** specifies the path to an optional mount point. Click on the **Browse** to navigate to the preferred mount point or type the path if known.
* **Set explicit Filesystem ID:** sets the fsid=X option. This is mainly used in a clustered setup. Using a consistent filesystem ID in all clusters avoids having stale NFS filehandles.



**Figure 18.5. NFS User Access**

The **User Access** tab allows the following options to be configured:

* **Treat remote root user as local root** — By default, the user and group IDs of the root user are both 0. Root squashing maps the user ID 0 and the group ID 0 to the user and group IDs of anonymous so that root on the client does not have root privileges on the NFS server. If this option is selected, root is not mapped to anonymous, and root on a client has root privileges to exported directories. Selecting this option can greatly decrease the security of the system. Do not select it unless it is absolutely necessary. This option corresponds to no\_root\_squash.
* **Treat all client users as anonymous users** — If this option is selected, all user and group IDs are mapped to the anonymous user. This option corresponds to all\_squash.
  + **Specify local user ID for anonymous users** — If **Treat all client users as anonymous users** is selected, this option lets you specify a user ID for the anonymous user. This option corresponds to anonuid.
  + **Specify local group ID for anonymous users** — If **Treat all client users as anonymous users** is selected, this option lets you specify a group ID for the anonymous user. This option corresponds to anongid.

To edit an existing NFS share, select the share from the list, and click the **Properties** button. To delete an existing NFS share, select the share from the list, and click the **Delete** button.

After clicking **OK** to add, edit, or delete an NFS share from the list, the changes take place immediately — the server daemon is restarted and the old configuration file is saved as /etc/exports.bak. The new configuration is written to /etc/exports.

The **NFS Server Configuration Tool** reads and writes directly to the /etc/exports configuration file. Thus, the file can be modified manually after using the tool, and the tool can be used after modifying the file manually (provided the file was modified with correct syntax).

The next this section discusses manually editing /etc/exports and using the /usr/sbin/exportfs command to export NFS file systems.

**18.6.2. Command Line Configuration**

If you prefer editing configuration files using a text editor or if you do not have the X Window System installed, you can modify the configuration file directly.

The /etc/exports file controls what directories the NFS server exports. Its format is as follows:

*directory* *hostname*(*options*)

The only option that needs to be specified is one of sync or async (sync is recommended). If sync is specified, the server does not reply to requests before the changes made by the request are written to the disk.

For example,

/misc/export speedy.example.com(sync)

would allow users from speedy.example.com to mount /misc/export with the default read-only permissions, but,

/misc/export speedy.example.com(rw,sync)

would allow users from speedy.example.com to mount /misc/export with read/write privileges.

Refer to [Section 18.6.3, “Hostname Formats”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-nfs-server-export.html#s2-nfs-hostname-formats) for an explanation of possible hostname formats.

**Caution**

Be careful with spaces in the /etc/exports file. If there are no spaces between the hostname and the options in parentheses, the options apply only to the hostname. If there is a space between the hostname and the options, the options apply to the rest of the world. For example, examine the following lines:

/misc/export speedy.example.com(rw,sync) /misc/export speedy.example.com (rw,sync)

The first line grants users from speedy.example.com read-write access and denies all other users. The second line grants users from speedy.example.com read-only access (the default) and allows the rest of the world read-write access.

Each time you change /etc/exports, you must inform the NFS daemon of the change, or reload the configuration file with the following command:

/sbin/service nfs reload

**18.6.3. Hostname Formats**

The host(s) can be in the following forms:

* Single machine — A fully qualified domain name (that can be resolved by the server), hostname (that can be resolved by the server), or an IP address.
* Series of machines specified with wildcards — Use the \* or ? character to specify a string match. Wildcards are not to be used with IP addresses; however, they may accidentally work if reverse DNS lookups fail. When specifying wildcards in fully qualified domain names, dots (.) are not included in the wildcard. For example, \*.example.com includes one.example.com but does not include one.two.example.com.
* IP networks — Use *a.b.c.d/z*, where *a.b.c.d* is the network and *z* is the number of bits in the netmask (for example 192.168.0.0/24). Another acceptable format is *a.b.c.d/netmask*, where *a.b.c.d* is the network and *netmask* is the netmask (for example, 192.168.100.8/255.255.255.0).
* Netgroups — In the format *@group-name*, where *group-name* is the NIS netgroup name.

## 18.8. Securing NFS

NFS is well suited for sharing entire file systems with a large number of known hosts in a transparent manner. However, with ease of use comes a variety of potential security problems.

The following points should be considered when exporting NFS file systems on a server or mounting them on a client. Doing so minimizes NFS security risks and better protects data on the server.

### 18.8.1. Host Access

Depending on which version of NFS you plan to implement, depends on your existing network environment, and your security concerns. The following sections explain the differences between implementing security measures with NFSv2, NFSv3, and NFSv4. If at all possible, use of NFSv4 is recommended over other versions of NFS.

#### 18.8.1.1. Using NFSv2 or NFSv3

NFS controls who can mount an exported file system based on the host making the mount request, not the user that actually uses the file system. Hosts must be given explicit rights to mount the exported file system. Access control is not possible for users, other than through file and directory permissions. In other words, once a file system is exported via NFS, any user on any remote host connected to the NFS server can access the shared data. To limit the potential risks, administrators often allow read-only access or squash user permissions to a common user and group ID. Unfortunately, these solutions prevent the NFS share from being used in the way it was originally intended.

Additionally, if an attacker gains control of the DNS server used by the system exporting the NFS file system, the system associated with a particular hostname or fully qualified domain name can be pointed to an unauthorized machine. At this point, the unauthorized machine is the system permitted to mount the NFS share, since no username or password information is exchanged to provide additional security for the NFS mount.

Wildcards should be used sparingly when exporting directories via NFS as it is possible for the scope of the wildcard to encompass more systems than intended.

It is also possible to restrict access to the portmap service via TCP wrappers. Access to ports used by portmap, rpc.mountd, and rpc.nfsd can also be limited by creating firewall rules with iptables.

For more information on securing NFS and portmap, refer to [Section 42.9, “IPTables”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-iptables.html).

#### 18.8.1.2. Using NFSv4

The release of NFSv4 brought a revolution to authentication and security to NFS exports. NFSv4 mandates the implementation of the RPCSEC\_GSS kernel module, the Kerberos version 5 GSS-API mechanism, SPKM-3, and LIPKEY. With NFSv4, the mandatory security mechanisms are oriented towards authenticating individual users, and not client machines as used in NFSv2 and NFSv3.

## Note

It is assumed that a Kerberos ticket-granting server (KDC) is installed and configured correctly, prior to configuring an NFSv4 server. Kerberos is a network authentication system which allows clients and servers to authenticate to each other through use of symmetric encryption and a trusted third party, the KDC.

NFSv4 includes ACL support based on the Microsoft Windows NT model, not the POSIX model, because of its features and because it is widely deployed. NFSv2 and NFSv3 do not have support for native ACL attributes.

Another important security feature of NFSv4 is its removal of the rpc.mountd daemon. The rpc.mountd daemon presented possible security holes because of the way it dealt with filehandlers.

For more information on the RPCSEC\_GSS framework, including how rpc.svcgssd and rpc.gssd inter operate, refer to <http://www.citi.umich.edu/projects/nfsv4/gssd/>.

### 18.8.2. File Permissions

Once the NFS file system is mounted read/write by a remote host, the only protection each shared file has is its permissions. If two users that share the same user ID value mount the same NFS file system, they can modify each others files. Additionally, anyone logged in as root on the client system can use the su - command to become a user who could access particular files via the NFS share.

By default, access control lists (ACLs) are supported by NFS under Red Hat Enterprise Linux. It is not recommended that this feature be disabled.

The default behavior when exporting a file system via NFS is to use *root squashing*. This sets the user ID of anyone accessing the NFS share as the root user on their local machine to a value of the server's nfsnobody account. Never turn off root squashing.

If exporting an NFS share as read-only, consider using the all\_squash option, which makes every user accessing the exported file system take the user ID of the nfsnobody user.

**Chapter 19. Samba**

[19.1. Introduction to Samba](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-samba.html#samba-rgs-overview)

[19.2. Samba Daemons and Related Services](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-samba-daemons.html)

[19.3. Connecting to a Samba Share](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-samba-connect-share.html)

[19.4. Configuring a Samba Server](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-samba-configuring.html)

[19.5. Starting and Stopping Samba](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-samba-startstop.html)

[19.6. Samba Server Types and the smb.conf File](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-samba-servers.html)

[19.7. Samba Security Modes](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-samba-security-modes.html)

[19.8. Samba Account Information Databases](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-samba-account-info-dbs.html)

[19.9. Samba Network Browsing](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-samba-network-browsing.html)

[19.10. Samba with CUPS Printing Support](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-samba-cups.html)

[19.11. Samba Distribution Programs](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-samba-programs.html)

[19.12. Additional Resources](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-samba-resources.html)

*Samba* is an open source implementation of the Server Message Block (SMB) protocol. It allows the networking of Microsoft Windows®, Linux, UNIX, and other operating systems together, enabling access to Windows-based file and printer shares. Samba's use of SMB allows it to appear as a Windows server to Windows clients.

**19.1. Introduction to Samba**

The third major release of Samba, version 3.0.0, introduced numerous improvements from prior versions, including:

* The ability to join an Active Directory domain by means of LDAP and Kerberos
* Built in Unicode support for internationalization
* Support for Microsoft Windows XP Professional client connections to Samba servers without needing local registry hacking
* Two new documents developed by the Samba.org team, which include a 400+ page reference manual, and a 300+ page implementation and integration manual. For more information about these published titles, refer to [Section 19.12.2, “Related Books”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-samba-resources.html#s2-samba-resources-published).

**19.1.1. Samba Features**

Samba is a powerful and versatile server application. Even seasoned system administrators must know its abilities and limitations before attempting installation and configuration.

What Samba can do:

* Serve directory trees and printers to Linux, UNIX, and Windows clients
* Assist in network browsing (with or without NetBIOS)
* Authenticate Windows domain logins
* Provide Windows Internet Name Service (WINS) name server resolution
* Act as a Windows NT®-style Primary Domain Controller (PDC)
* Act as a Backup Domain Controller (BDC) for a Samba-based PDC
* Act as an Active Directory domain member server
* Join a Windows NT/2000/2003 PDC

What Samba cannot do:

* Act as a BDC for a Windows PDC (and vice versa)
* Act as an Active Directory domain controller

## 19.2. Samba Daemons and Related Services

The following is a brief introduction to the individual Samba daemons and services.

### 19.2.1. Samba Daemons

Samba is comprised of three daemons (smbd, nmbd, and winbindd). Two services (smb and windbind) control how the daemons are started, stopped, and other service-related features. Each daemon is listed in detail, as well as which specific service has control over it.

##### smbd

The smbd server daemon provides file sharing and printing services to Windows clients. In addition, it is responsible for user authentication, resource locking, and data sharing through the SMB protocol. The default ports on which the server listens for SMB traffic are TCP ports 139 and 445.

The smbd daemon is controlled by the smb service.

##### nmbd

The nmbd server daemon understands and replies to NetBIOS name service requests such as those produced by SMB/CIFS in Windows-based systems. These systems include Windows 95/98/ME, Windows NT, Windows 2000, Windows XP, and LanManager clients. It also participates in the browsing protocols that make up the Windows **Network Neighborhood** view. The default port that the server listens to for NMB traffic is UDP port 137.

The nmbd daemon is controlled by the smb service.

##### winbindd

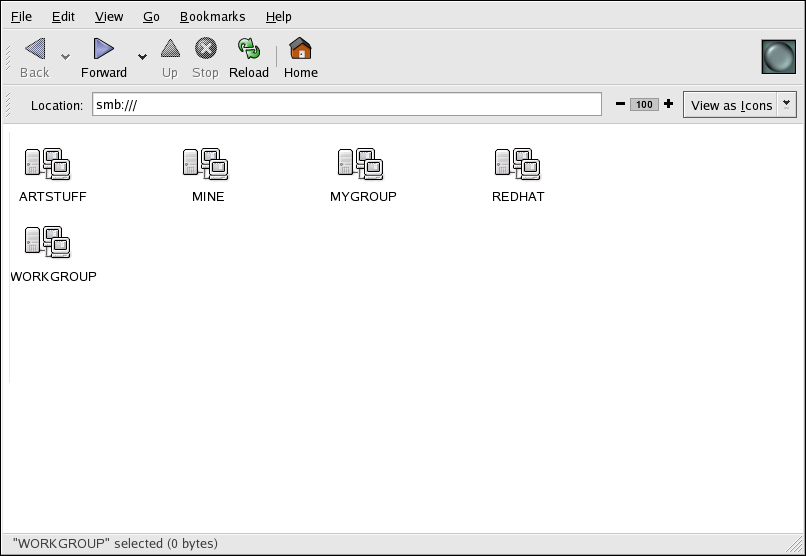
The winbind service resolves user and group information on a server running Windows NT 2000 or Windows Server 2003. This makes Windows user / group information understandable by UNIX platforms. This is achieved by using Microsoft RPC calls, Pluggable Authentication Modules (PAM), and the Name Service Switch (NSS). This allows Windows NT domain users to appear and operate as UNIX users on a UNIX machine. Though bundled with the Samba distribution, the winbind service is controlled separately from the smb service.

The winbindd daemon is controlled by the winbind service and does not require the smb service to be started in order to operate. Winbindd is also used when Samba is an Active Directory member, and may also be used on a Samba domain controller (to implement nested groups and/or interdomain trust). Because winbind is a client-side service used to connect to Windows NT-based servers, further discussion of winbind is beyond the scope of this manual.

**19.3. Connecting to a Samba Share**

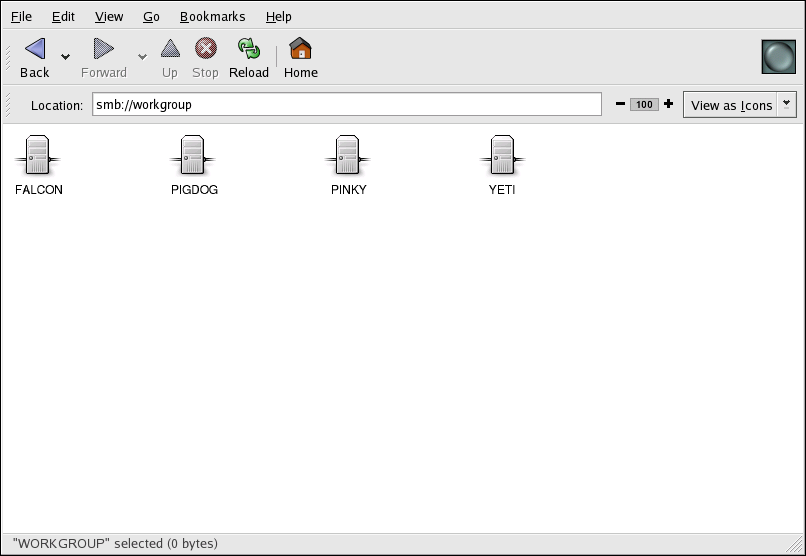
You can use **Nautilus** to view available Samba shares on your network. Select **Places** (on the Panel) => **Network Servers** to view a list of Samba workgroups on your network. You can also type **smb:** in the **File** => **Open Location** bar of Nautilus to view the workgroups.

As shown in [Figure 19.1, “SMB Workgroups in Nautilus”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-samba-connect-share.html#fig-samba-nautilus-workgroups), an icon appears for each available SMB workgroup on the network.



**Figure 19.1. SMB Workgroups in Nautilus**

Double-click one of the workgroup icons to view a list of computers within the workgroup.



**Figure 19.2. SMB Machines in Nautilus**

As you can see from [Figure 19.2, “SMB Machines in Nautilus”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-samba-connect-share.html#fig-samba-nautilus-machines), there is an icon for each machine within the workgroup. Double-click on an icon to view the Samba shares on the machine. If a username and password combination is required, you are prompted for them.

Alternately, you can also specify the Samba server and sharename in the **Location:** bar for **Nautilus** using the following syntax (replace *<servername>* and *<sharename>* with the appropriate values):

smb://*<servername>*/*<sharename>*

**19.3.1. Command Line**

To query the network for Samba servers, use the findsmb command. For each server found, it displays its IP address, NetBIOS name, workgroup name, operating system, and SMB server version.

To connect to a Samba share from a shell prompt, type the following command:

smbclient //*<hostname>*/*<sharename>* -U *<username>*

Replace *<hostname>* with the hostname or IP address of the Samba server you want to connect to, *<sharename>* with the name of the shared directory you want to browse, and *<username>* with the Samba username for the system. Enter the correct password or press **Enter** if no password is required for the user.

If you see the smb:\> prompt, you have successfully logged in. Once you are logged in, type **help** for a list of commands. If you wish to browse the contents of your home directory, replace *sharename* with your username. If the -U switch is not used, the username of the current user is passed to the Samba server.

To exit smbclient, type **exit** at the smb:\> prompt.

**19.3.2. Mounting the Share**

Sometimes it is useful to mount a Samba share to a directory so that the files in the directory can be treated as if they are part of the local file system.

To mount a Samba share to a directory, create create a directory to mount it to (if it does not already exist), and execute the following command as root:

mount -t cifs -o *<username>*,*<password>* //*<servername>*/*<sharename>* */mnt/point/*

This command mounts *<sharename>* from *<servername>* in the local directory */mnt/point/*. For more information about mounting a samba share, refer to man mount.cifs.

**19.4. Configuring a Samba Server**

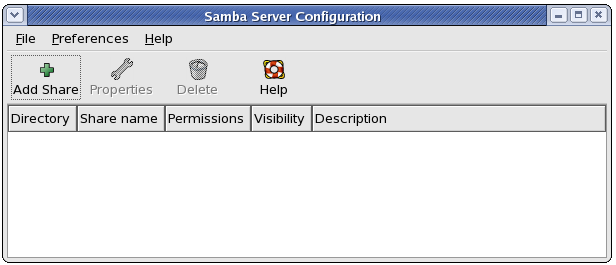
The default configuration file (/etc/samba/smb.conf) allows users to view their home directories as a Samba share. It also shares all printers configured for the system as Samba shared printers. In other words, you can attach a printer to the system and print to it from the Windows machines on your network.

**19.4.1. Graphical Configuration**

To configure Samba using a graphical interface, use the **Samba Server Configuration Tool**. For command line configuration, skip to [Section 19.4.2, “Command Line Configuration”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-samba-configuring.html#s2-samba-configuring-cmdline).

The **Samba Server Configuration Tool** is a graphical interface for managing Samba shares, users, and basic server settings. It modifies the configuration files in the /etc/samba/ directory. Any changes to these files not made using the application are preserved.

To use this application, you must be running the X Window System, have root privileges, and have the system-config-samba RPM package installed. To start the **Samba Server Configuration Tool** from the desktop, go to the **System** (on the Panel) => **Administration** => **Server Settings** => **Samba** or type the command system-config-samba at a shell prompt (for example, in an XTerm or a GNOME terminal).



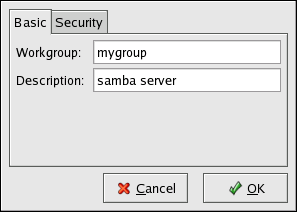
**Figure 19.3. Samba Server Configuration Tool**

**Note**

The **Samba Server Configuration Tool** does not display shared printers or the default stanza that allows users to view their own home directories on the Samba server.

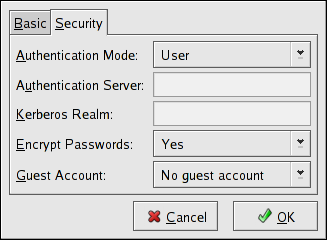
**19.4.1.1. Configuring Server Settings**

The first step in configuring a Samba server is to configure the basic settings for the server and a few security options. After starting the application, select **Preferences** => **Server Settings** from the pulldown menu. The **Basic** tab is displayed as shown in [Figure 19.4, “Configuring Basic Server Settings”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-samba-configuring.html#fig-samba-basic).



**Figure 19.4. Configuring Basic Server Settings**

On the **Basic** tab, specify which workgroup the computer should be in as well as a brief description of the computer. They correspond to the workgroup and server string options in smb.conf.



**Figure 19.5. Configuring Security Server Settings**

The **Security** tab contains the following options:

* **Authentication Mode** — This corresponds to the security option. Select one of the following types of authentication.
  + **ADS** — The Samba server acts as a domain member in an Active Directory Domain (ADS) realm. For this option, Kerberos must be installed and configured on the server, and Samba must become a member of the ADS realm using the net utility, which is part of the samba-client package. Refer to the net man page for details. This option does not configure Samba to be an ADS Controller. Specify the realm of the Kerberos server in the **Kerberos Realm** field.

**Note**

The **Kerberos Realm** field must be supplied in all uppercase letters, such as EXAMPLE.COM.

Using a Samba server as a domain member in an ADS realm assumes proper configuration of Kerberos, including the /etc/krb5.conf file.

* + **Domain** — The Samba server relies on a Windows NT Primary or Backup Domain Controller to verify the user. The server passes the username and password to the Controller and waits for it to return. Specify the NetBIOS name of the Primary or Backup Domain Controller in the **Authentication Server** field.

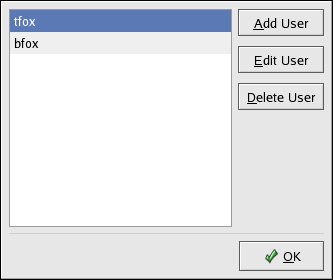
The **Encrypted Passwords** option must be set to **Yes** if this is selected.

* + **Server** — The Samba server tries to verify the username and password combination by passing them to another Samba server. If it can not, the server tries to verify using the user authentication mode. Specify the NetBIOS name of the other Samba server in the **Authentication Server** field.
  + **Share** — Samba users do not have to enter a username and password combination on a per Samba server basis. They are not prompted for a username and password until they try to connect to a specific shared directory from a Samba server.
  + **User** — (Default) Samba users must provide a valid username and password on a per Samba server basis. Select this option if you want the **Windows Username** option to work. Refer to [Section 19.4.1.2, “Managing Samba Users”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-samba-configuring.html#s3-samba-gui-users) for details.
* **Encrypt Passwords** — This option must be enabled if the clients are connecting from a system with Windows 98, Windows NT 4.0 with Service Pack 3, or other more recent versions of Microsoft Windows. The passwords are transfered between the server and the client in an encrypted format instead of as a plain-text word that can be intercepted. This corresponds to the encrypted passwords option. Refer to [Section 19.4.3, “Encrypted Passwords”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-samba-configuring.html#s2-samba-encrypted-passwords) for more information about encrypted Samba passwords.
* **Guest Account** — When users or guest users log into a Samba server, they must be mapped to a valid user on the server. Select one of the existing usernames on the system to be the guest Samba account. When guests log in to the Samba server, they have the same privileges as this user. This corresponds to the guest account option.

After clicking **OK**, the changes are written to the configuration file and the daemon is restarted; thus, the changes take effect immediately.

**19.4.1.2. Managing Samba Users**

The **Samba Server Configuration Tool** requires that an existing user account be active on the system acting as the Samba server before a Samba user can be added. The Samba user is associated with the existing user account.



**Figure 19.6. Managing Samba Users**

To add a Samba user, select **Preferences** => **Samba Users** from the pulldown menu, and click the **Add User** button. In the **Create New Samba User** window select a **Unix Username** from the list of existing users on the local system.

If the user has a different username on a Windows machine and needs to log into the Samba server from the Windows machine, specify that Windows username in the **Windows Username** field. The **Authentication Mode** on the **Security** tab of the **Server Settings** preferences must be set to **User** for this option to work.

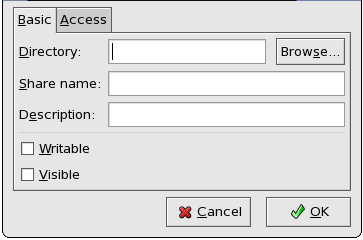
Also, configure a **Samba Password** for the Samba User and confirm it by typing it again. Even if you opt to use encrypted passwords for Samba, it is recommended that the Samba passwords for all users are different from their system passwords.

To edit an existing user, select the user from the list, and click **Edit User**. To delete an existing Samba user, select the user, and click the **Delete User** button. Deleting a Samba user does not delete the associated system user account.

The users are modified immediately after clicking the **OK** button.

**19.4.1.3. Adding a Share**

To create a Samba share, click the **Add** button from the main Samba configuration window.



**Figure 19.7. Adding a Share**

The **Basic** tab configures the following options:

* **Directory** — The directory to share via Samba. The directory must exist before it can be entered here.
* **Share name** — The actual name of the share that is seen from remote machines. By default, it is the same value as **Directory**, but can be configured.
* **Descriptions** — A brief description of the share.
* **Writable** — Enables users to read and write to the shared directory
* **Visible** — Grants read-only rights to users for the shared directory.

On the **Access** tab, select whether to allow only specified users to access the share or whether to allow all Samba users to access the share. If you select to allow access to specific users, select the users from the list of available Samba users.

The share is added immediately after clicking **OK**.

**19.4.2. Command Line Configuration**

Samba uses /etc/samba/smb.conf as its configuration file. If you change this configuration file, the changes do not take effect until you restart the Samba daemon with the command service smb restart.

To specify the Windows workgroup and a brief description of the Samba server, edit the following lines in your smb.conf file:

workgroup = *WORKGROUPNAME*

server string = *BRIEF COMMENT ABOUT SERVER*

Replace *WORKGROUPNAME* with the name of the Windows workgroup to which this machine should belong. The *BRIEF COMMENT ABOUT SERVER* is optional and is used as the Windows comment about the Samba system.

To create a Samba share directory on your Linux system, add the following section to your smb.conf file (after modifying it to reflect your needs and your system):

[*sharename*]

comment = *Insert a comment here*

path = */home/share/*

valid users = *tfox carole*

public = no

writable = yes

printable = no

create mask = 0765

The above example allows the users tfox and carole to read and write to the directory /home/share, on the Samba server, from a Samba client.

**19.4.3. Encrypted Passwords**

Encrypted passwords are enabled by default because it is more secure to do so. To create a user with an encrypted password, use the command smbpasswd -a *<username>*.

## 19.5. Starting and Stopping Samba

To start a Samba server, type the following command in a shell prompt while logged in as root:

/sbin/service smb start

## Important

To set up a domain member server, you must first join the domain or Active Directory using the net join command before starting the smb service.

To stop the server, type the following command in a shell prompt while logged in as root:

/sbin/service smb stop

The restart option is a quick way of stopping and then starting Samba. This is the most reliable way to make configuration changes take effect after editing the configuration file for Samba. Note that the restart option starts the daemon even if it was not running originally.

To restart the server, type the following command in a shell prompt while logged in as root:

/sbin/service smb restart

The condrestart (*conditional restart*) option only starts smb on the condition that it is currently running. This option is useful for scripts, because it does not start the daemon if it is not running.

## Note

When the smb.conf file is changed, Samba automatically reloads it after a few minutes. Issuing a manual restart or reload is just as effective.

To conditionally restart the server, type the following command as root:

/sbin/service smb condrestart

A manual reload of the smb.conf file can be useful in case of a failed automatic reload by the smb service. To ensure that the Samba server configuration file is reloaded without restarting the service, type the following command as root:

/sbin/service smb reload

By default, the smb service does not start automatically at boot time. To configure Samba to start at boot time, use an initscript utility, such as /sbin/chkconfig, /usr/sbin/ntsysv, or the **Services Configuration Tool** program. Refer to [Chapter 15, *Controlling Access to Services*](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-services.html) for more information regarding these tools.

## 19.6. Samba Server Types and the smb.conf File

Samba configuration is straightforward. All modifications to Samba are done in the /etc/samba/smb.conf configuration file. Although the default smb.conf file is well documented, it does not address complex topics such as LDAP, Active Directory, and the numerous domain controller implementations.

The following sections describe the different ways a Samba server can be configured. Keep in mind your needs and the changes required to the smb.conf file for a successful configuration.

### 19.6.1. Stand-alone Server

A stand-alone server can be a workgroup server or a member of a workgroup environment. A stand-alone server is not a domain controller and does not participate in a domain in any way. The following examples include several anonymous share-level security configurations and one user-level security configuration. For more information on share-level and user-level security modes, refer to [Section 19.7, “Samba Security Modes”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-samba-security-modes.html).

#### 19.6.1.1. Anonymous Read-Only

The following smb.conf file shows a sample configuration needed to implement anonymous read-only file sharing. The security = share parameter makes a share anonymous. Note, security levels for a single Samba server cannot be mixed. The security directive is a global Samba parameter located in the [global] configuration section of the smb.conf file.

[global]

workgroup = DOCS

netbios name = DOCS\_SRV

security = share

[data]

comment = Documentation Samba Server

path = /export

read only = Yes

guest only = Yes

#### 19.6.1.2. Anonymous Read/Write

The following smb.conf file shows a sample configuration needed to implement anonymous read/write file sharing. To enable anonymous read/write file sharing, set the read only directive to no. The force user and force group directives are also added to enforce the ownership of any newly placed files specified in the share.

## Note

Although having an anonymous read/write server is possible, it is not recommended. Any files placed in the share space, regardless of user, are assigned the user/group combination as specified by a generic user (force user) and group (force group) in the smb.conf file.

[global]

workgroup = DOCS

netbios name = DOCS\_SRV

security = share

[data]

comment = Data

path = /export

force user = docsbot

force group = users

read only = No

guest ok = Yes

#### 19.6.1.3. Anonymous Print Server

The following smb.conf file shows a sample configuration needed to implement an anonymous print server. Setting browseable to no as shown does not list the printer in Windows **Network Neighborhood**. Although hidden from browsing, configuring the printer explicitly is possible. By connecting to DOCS\_SRV using NetBIOS, the client can have access to the printer if the client is also part of the DOCS workgroup. It is also assumed that the client has the correct local printer driver installed, as the use client driver directive is set to Yes. In this case, the Samba server has no responsibility for sharing printer drivers to the client.

[global]

workgroup = DOCS

netbios name = DOCS\_SRV

security = share

printcap name = cups

disable spools= Yes

show add printer wizard = No

printing = cups

[printers]

comment = All Printers

path = /var/spool/samba

guest ok = Yes

printable = Yes

use client driver = Yes

browseable = Yes

#### 19.6.1.4. Secure Read/Write File and Print Server

The following smb.conf file shows a sample configuration needed to implement a secure read/write print server. Setting the security directive to user forces Samba to authenticate client connections. Notice the [homes] share does not have a force user or force group directive as the [public] share does. The [homes] share uses the authenticated user details for any files created as opposed to the force user and force group in [public].

[global]

workgroup = DOCS

netbios name = DOCS\_SRV

security = user

printcap name = cups

disable spools = Yes

show add printer wizard = No

printing = cups

[homes]

comment = Home Directories

valid users = %S

read only = No

browseable = No

[public]

comment = Data

path = /export

force user = docsbot

force group = users

guest ok = Yes

[printers]

comment = All Printers

path = /var/spool/samba

printer admin = john, ed, @admins

create mask = 0600

guest ok = Yes

printable = Yes

use client driver = Yes

browseable = Yes

### 19.6.2. Domain Member Server

A domain member, while similar to a stand-alone server, is logged into a domain controller (either Windows or Samba) and is subject to the domain's security rules. An example of a domain member server would be a departmental server running Samba that has a machine account on the Primary Domain Controller (PDC). All of the department's clients still authenticate with the PDC, and desktop profiles and all network policy files are included. The difference is that the departmental server has the ability to control printer and network shares.

#### 19.6.2.1. Active Directory Domain Member Server

The following smb.conf file shows a sample configuration needed to implement an Active Directory domain member server. In this example, Samba authenticates users for services being run locally but is also a client of the Active Directory. Ensure that your kerberos realm parameter is shown in all caps (for example realm = EXAMPLE.COM). Since Windows 2000/2003 requires Kerberos for Active Directory authentication, the realm directive is required. If Active Directory and Kerberos are running on different servers, the password server directive may be required to help the distinction.

[global]

realm = EXAMPLE.COM

security = ADS

encrypt passwords = yes

# Optional. Use only if Samba cannot determine the Kerberos server automatically.

password server = kerberos.example.com

In order to join a member server to an Active Directory domain, the following steps must be completed:

* Configuration of the smb.conf file on the member server
* Configuration of Kerberos, including the /etc/krb5.conf file, on the member server
* Creation of the machine account on the Active Directory domain server
* Association of the member server to the Active Directory domain

To create the machine account and join the Windows 2000/2003 Active Directory, Kerberos must first be initialized for the member server wishing to join the Active Directory domain. To create an administrative Kerberos ticket, type the following command as root on the member server:

**kinit administrator@EXAMPLE.COM**

The kinit command is a Kerberos initialization script that references the Active Directory administrator account and Kerberos realm. Since Active Directory requires Kerberos tickets, kinit obtains and caches a Kerberos ticket-granting ticket for client/server authentication. For more information on Kerberos, the /etc/krb5.conf file, and the kinit command, refer to [Section 42.6, “Kerberos”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-kerberos.html).

To join an Active Directory server (windows1.example.com), type the following command as root on the member server:

**net ads join -S windows1.example.com -U administrator%password**

Since the machine windows1 was automatically found in the corresponding Kerberos realm (the kinit command succeeded), the net command connects to the Active Directory server using its required administrator account and password. This creates the appropriate machine account on the Active Directory and grants permissions to the Samba domain member server to join the domain.

## Note

Since security = ads and not security = user is used, a local password backend such as smbpasswd is not needed. Older clients that do not support security = ads are authenticated as if security = domain had been set. This change does not affect functionality and allows local users not previously in the domain.

#### 19.6.2.2. Windows NT4-based Domain Member Server

The following smb.conf file shows a sample configuration needed to implement a Windows NT4-based domain member server. Becoming a member server of an NT4-based domain is similar to connecting to an Active Directory. The main difference is NT4-based domains do not use Kerberos in their authentication method, making the smb.conf file simpler. In this instance, the Samba member server functions as a pass through to the NT4-based domain server.

[global]

workgroup = DOCS

netbios name = DOCS\_SRV

security = domain

[homes]

comment = Home Directories

valid users = %S

read only = No

browseable = No

[public]

comment = Data

path = /export

force user = docsbot

force group = users

guest ok = Yes

Having Samba as a domain member server can be useful in many situations. There are times where the Samba server can have other uses besides file and printer sharing. It may be beneficial to make Samba a domain member server in instances where Linux-only applications are required for use in the domain environment. Administrators appreciate keeping track of all machines in the domain, even if not Windows-based. In the event the Windows-based server hardware is deprecated, it is quite easy to modify the smb.conf file to convert the server to a Samba-based PDC. If Windows NT-based servers are upgraded to Windows 2000/2003, the smb.conf file is easily modifiable to incorporate the infrastructure change to Active Directory if needed.

## Important

After configuring the smb.conf file, join the domain before starting Samba by typing the following command as root:

**net rpc join -U administrator%password**

Note that the -S option, which specifies the domain server hostname, does not need to be stated in the net rpc join command. Samba uses the hostname specified by the workgroup directive in the smb.conf file instead of it being stated explicitly.

### 19.6.3. Domain Controller

A domain controller in Windows NT is functionally similar to a Network Information Service (NIS) server in a Linux environment. Domain controllers and NIS servers both host user/group information databases as well as related services. Domain controllers are mainly used for security, including the authentication of users accessing domain resources. The service that maintains the user/group database integrity is called the *Security Account Manager* (SAM). The SAM database is stored differently between Windows and Linux Samba-based systems, therefore SAM replication cannot be achieved and platforms cannot be mixed in a PDC/BDC environment.

In a Samba environment, there can be only one PDC and zero or more BDCs.

## Important

Samba cannot exist in a mixed Samba/Windows domain controller environment (Samba cannot be a BDC of a Windows PDC or vice versa). Alternatively, Samba PDCs and BDCs can coexist.

#### 19.6.3.1. Primary Domain Controller (PDC) using tdbsam

The simplest and most common implementation of a Samba PDC uses the tdbsam password database backend. Planned to replace the aging smbpasswd backend, tdbsam has numerous improvements that are explained in more detail in [Section 19.8, “Samba Account Information Databases”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-samba-account-info-dbs.html). The passdb backend directive controls which backend is to be used for the PDC.

[global]

workgroup = DOCS

netbios name = DOCS\_SRV

passdb backend = tdbsam

security = user

add user script = /usr/sbin/useradd -m %u

delete user script = /usr/sbin/userdel -r %u

add group script = /usr/sbin/groupadd %g

delete group script = /usr/sbin/groupdel %g

add user to group script = /usr/sbin/usermod -G %g %u

add machine script = /usr/sbin/useradd -s /bin/false -d /dev/null -g machines %u

# The following specifies the default logon script

# Per user logon scripts can be specified in the user

# account using pdbedit logon script = logon.bat

# This sets the default profile path.

# Set per user paths with pdbedit

logon drive = H:

domain logons = Yes

os level = 35

preferred master = Yes

domain master = Yes

[homes]

comment = Home Directories

valid users = %S

read only = No

[netlogon]

comment = Network Logon Service

path = /var/lib/samba/netlogon/scripts

browseable = No

read only = No

# For profiles to work, create a user directory under the

# path shown.

mkdir -p /var/lib/samba/profiles/john

[Profiles]

comment = Roaming Profile Share

path = /var/lib/samba/profiles

read only = No

browseable = No

guest ok = Yes

profile acls = Yes

# Other resource shares ... ...

## Note

If you need more than one domain controller or have more than 250 users, do not use a tdbsam authentication backend. LDAP is recommended in these cases.

#### 19.6.3.2. Primary Domain Controller (PDC) with Active Directory

Although it is possible for Samba to be a member of an Active Directory, it is not possible for Samba to operate as an Active Directory domain controller.

## 19.7. Samba Security Modes

There are only two types of security modes for Samba, share-level and user-level, which are collectively known as security levels. Share-level security can only be implemented in one way, while user-level security can be implemented in one of four different ways. The different ways of implementing a security level are called security modes.

### 19.7.1. User-Level Security

User-level security is the default setting for Samba. Even if the security = user directive is not listed in the smb.conf file, it is used by Samba. If the server accepts the client's username/password, the client can then mount multiple shares without specifying a password for each instance. Samba can also accept session-based username/password requests. The client maintains multiple authentication contexts by using a unique UID for each logon.

In smb.conf, the security = user directive that sets user-level security is:

[GLOBAL]

...

security = user

...

The following sections describe other implementations of user-level security.

#### 19.7.1.1. Domain Security Mode (User-Level Security)

In domain security mode, the Samba server has a machine account (domain security trust account) and causes all authentication requests to be passed through to the domain controllers. The Samba server is made into a domain member server by using the following directives in smb.conf:

[GLOBAL]

...

security = domain

workgroup = MARKETING

...

#### 19.7.1.2. Active Directory Security Mode (User-Level Security)

If you have an Active Directory environment, it is possible to join the domain as a native Active Directory member. Even if a security policy restricts the use of NT-compatible authentication protocols, the Samba server can join an ADS using Kerberos. Samba in Active Directory member mode can accept Kerberos tickets.

In smb.conf, the following directives make Samba an Active Directory member server:

[GLOBAL]

...

security = ADS

realm = EXAMPLE.COM

password server = kerberos.example.com

...

#### 19.7.1.3. Server Security Mode (User-Level Security)

Server security mode was previously used when Samba was not capable of acting as a domain member server.

## Note

It is highly recommended to not use this mode since there are numerous security drawbacks.

In smb.conf, the following directives enable Samba to operate in server security mode:

[GLOBAL]

...

encrypt passwords = Yes

security = server

password server = "NetBIOS\_of\_Domain\_Controller"

...

### 19.7.2. Share-Level Security

With share-level security, the server accepts only a password without an explicit username from the client. The server expects a password for each share, independent of the username. There have been recent reports that Microsoft Windows clients have compatibility issues with share-level security servers. Samba developers strongly discourage use of share-level security.

In smb.conf, the security = share directive that sets share-level security is:

[GLOBAL]

...

security = share

...

**19.8. Samba Account Information Databases**

The latest release of Samba offers many new features including new password database backends not previously available. Samba version 3.0.0 fully supports all databases used in previous versions of Samba. However, although supported, many backends may not be suitable for production use.

The following is a list different backends you can use with Samba. Other backends not listed here may also be available.

Plain Text

Plain text backends are nothing more than the /etc/passwd type backends. With a plain text backend, all usernames and passwords are sent unencrypted between the client and the Samba server. This method is very unsecure and is not recommended for use by any means. It is possible that different Windows clients connecting to the Samba server with plain text passwords cannot support such an authentication method.

smbpasswd

A popular backend used in previous Samba packages, the smbpasswd backend utilizes a plain ASCII text layout that includes the MS Windows LanMan and NT account, and encrypted password information. The smbpasswd backend lacks the storage of the Windows NT/2000/2003 SAM extended controls. The smbpasswd backend is not recommended because it does not scale well or hold any Windows information, such as RIDs for NT-based groups. The tdbsam backend solves these issues for use in a smaller database (250 users), but is still not an enterprise-class solution.

ldapsam\_compat

The ldapsam\_compat backend allows continued OpenLDAP support for use with upgraded versions of Samba. This option normally used when migrating to Samba 3.0.

tdbsam

The tdbsam backend provides an ideal database backend for local servers, servers that do not need built-in database replication, and servers that do not require the scalability or complexity of LDAP. The tdbsam backend includes all of the smbpasswd database information as well as the previously-excluded SAM information. The inclusion of the extended SAM data allows Samba to implement the same account and system access controls as seen with Windows NT/2000/2003-based systems.

The tdbsam backend is recommended for 250 users at most. Larger organizations should require Active Directory or LDAP integration due to scalability and possible network infrastructure concerns.

ldapsam

The ldapsam backend provides an optimal distributed account installation method for Samba. LDAP is optimal because of its ability to replicate its database to any number of servers using the OpenLDAP slurpd daemon. LDAP databases are light-weight and scalable, and as such are preferred by large enterprises.

If you are upgrading from a previous version of Samba to 3.0, note that the /usr/share/doc/samba-*<version>*/LDAP/samba.schema has changed. This file contains the *attribute syntax definitions* and *objectclass definitions* that the ldapsam backend will need in order to function properly.

As such, if you are using the ldapsam backend for your Samba server, you will need to configure slapd to include this schema file. Refer to [Section 24.5, “The /etc/openldap/schema/ Directory”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ldap-files-schemas.html) for directions on how to do this.

**Note**

You will need to have the openldap-server package installed if you want to use the ldapsam backend.

mysqlsam

The mysqlsam backend uses a MySQL-based database backend. This is useful for sites that already implement MySQL. At present, mysqlsam is now packed in a module separate from Samba, and as such is not officially supported by Samba.

## 19.9. Samba Network Browsing

*Network browsing* enables Windows and Samba servers to appear in the Windows **Network Neighborhood**. Inside the **Network Neighborhood**, icons are represented as servers and if opened, the server's shares and printers that are available are displayed.

Network browsing capabilities require NetBIOS over TCP/IP. NetBIOS-based networking uses broadcast (UDP) messaging to accomplish browse list management. Without NetBIOS and WINS as the primary method for TCP/IP hostname resolution, other methods such as static files (/etc/hosts) or DNS, must be used.

A domain master browser collates the browse lists from local master browsers on all subnets so that browsing can occur between workgroups and subnets. Also, the domain master browser should preferably be the local master browser for its own subnet.

### 19.9.1. Domain Browsing

By default, a Windows server PDC for a domain is also the domain master browser for that domain. A Samba server must note be set up as a domain master server in this type of situation

For subnets that do not include the Windows server PDC, a Samba server can be implemented as a local master browser. Configuring the smb.conf for a local master browser (or no browsing at all) in a domain controller environment is the same as workgroup configuration.

### 19.9.2. WINS (Windows Internetworking Name Server)

Either a Samba server or a Windows NT server can function as a WINS server. When a WINS server is used with NetBIOS enabled, UDP unicasts can be routed which allows name resolution across networks. Without a WINS server, the UDP broadcast is limited to the local subnet and therefore cannot be routed to other subnets, workgroups, or domains. If WINS replication is necessary, do not use Samba as your primary WINS server, as Samba does not currently support WINS replication.

In a mixed NT/2000/2003 server and Samba environment, it is recommended that you use the Microsoft WINS capabilities. In a Samba-only environment, it is recommended that you use only one Samba server for WINS.

The following is an example of the smb.conf file in which the Samba server is serving as a WINS server:

[global]

wins support = Yes

## Tip

All servers (including Samba) should connect to a WINS server to resolve NetBIOS names. Without WINS, browsing only occurs on the local subnet. Furthermore, even if a domain-wide list is somehow obtained, hosts cannot be resolved for the client without WINS.

## 19.11. Samba Distribution Programs

##### findsmb

findsmb *<subnet\_broadcast\_address>*

The findsmb program is a Perl script which reports information about SMB-aware systems on a specific subnet. If no subnet is specified the local subnet is used. Items displayed include IP address, NetBIOS name, workgroup or domain name, operating system, and version.

The following example shows the output of executing findsmb as any valid user on a system:

**findsmb**

IP ADDR NETBIOS NAME WORKGROUP/OS/VERSION

------------------------------------------------------------------

10.1.59.25 VERVE [MYGROUP] [Unix] [Samba 3.0.0-15]

10.1.59.26 STATION22 [MYGROUP] [Unix] [Samba 3.0.2-7.FC1]

10.1.56.45 TREK +[WORKGROUP] [Windows 5.0] [Windows 2000 LAN Manager]

10.1.57.94 PIXEL [MYGROUP] [Unix] [Samba 3.0.0-15]

10.1.57.137 MOBILE001 [WORKGROUP] [Windows 5.0] [Windows 2000 LAN Manager]

10.1.57.141 JAWS +[KWIKIMART] [Unix] [Samba 2.2.7a-security-rollup-fix]

10.1.56.159 FRED +[MYGROUP] [Unix] [Samba 3.0.0-14.3E]

10.1.59.192 LEGION \*[MYGROUP] [Unix] [Samba 2.2.7-security-rollup-fix]

10.1.56.205 NANCYN +[MYGROUP] [Unix] [Samba 2.2.7a-security-rollup-fix]

##### net

net *<protocol> <function> <misc\_options> <target\_options>*

The net utility is similar to the net utility used for Windows and MS-DOS. The first argument is used to specify the protocol to use when executing a command. The *<protocol>* option can be ads, rap, or rpc for specifying the type of server connection. Active Directory uses ads, Win9x/NT3 uses rap, and Windows NT4/2000/2003 uses rpc. If the protocol is omitted, net automatically tries to determine it.

The following example displays a list the available shares for a host named wakko:

**net -l share -S wakko**

Password:

Enumerating shared resources (exports) on remote server:

Share name Type Description

---------- ---- -----------

data Disk Wakko data share

tmp Disk Wakko tmp share

IPC$ IPC IPC Service (Samba Server)

ADMIN$ IPC IPC Service (Samba Server)

The following example displays a list of Samba users for a host named wakko:

**net -l user -S wakko**

root password:

User name Comment

-----------------------------

andriusb Documentation

joe Marketing

lisa Sales

##### nmblookup

nmblookup *<options> <netbios\_name>*

The nmblookup program resolves NetBIOS names into IP addresses. The program broadcasts its query on the local subnet until the target machine replies.

Here is an example:

**nmblookup trek**

querying trek on 10.1.59.255

10.1.56.45 trek<00>

##### pdbedit

pdbedit *<options>*

The pdbedit program manages accounts located in the SAM database. All backends are supported including smbpasswd, LDAP, NIS+, and the tdb database library.

The following are examples of adding, deleting, and listing users:

**pdbedit -a kristin**

new password:

retype new password:

Unix username: kristin

NT username:

Account Flags: [U ]

User SID: S-1-5-21-1210235352-3804200048-1474496110-2012

Primary Group SID: S-1-5-21-1210235352-3804200048-1474496110-2077

Full Name: Home Directory: \\wakko\kristin

HomeDir Drive:

Logon Script:

Profile Path: \\wakko\kristin\profile

Domain: WAKKO

Account desc:

Workstations: Munged

dial:

Logon time: 0

Logoff time: Mon, 18 Jan 2038 22:14:07 GMT

Kickoff time: Mon, 18 Jan 2038 22:14:07 GMT

Password last set: Thu, 29 Jan 2004 08:29:28

GMT Password can change: Thu, 29 Jan 2004 08:29:28 GMT

Password must change: Mon, 18 Jan 2038 22:14:07 GMT

**pdbedit -v -L kristin**

Unix username: kristin

NT username:

Account Flags: [U ]

User SID: S-1-5-21-1210235352-3804200048-1474496110-2012

Primary Group SID: S-1-5-21-1210235352-3804200048-1474496110-2077

Full Name:

Home Directory: \\wakko\kristin

HomeDir Drive:

Logon Script:

Profile Path: \\wakko\kristin\profile

Domain: WAKKO

Account desc:

Workstations: Munged

dial:

Logon time: 0

Logoff time: Mon, 18 Jan 2038 22:14:07 GMT

Kickoff time: Mon, 18 Jan 2038 22:14:07 GMT

Password last set: Thu, 29 Jan 2004 08:29:28 GMT

Password can change: Thu, 29 Jan 2004 08:29:28 GMT

Password must change: Mon, 18 Jan 2038 22:14:07 GMT

**pdbedit -L**

andriusb:505:

joe:503:

lisa:504:

kristin:506:

**pdbedit -x joe**

**pdbedit -L**

andriusb:505: lisa:504: kristin:506:

##### rpcclient

rpcclient *<server> <options>*

The rpcclient program issues administrative commands using Microsoft RPCs, which provide access to the Windows administration graphical user interfaces (GUIs) for systems management. This is most often used by advanced users that understand the full complexity of Microsoft RPCs.

##### smbcacls

smbcacls *<//server/share> <filename> <options>*

The smbcacls program modifies Windows ACLs on files and directories shared by the Samba server.

##### smbclient

smbclient *<//server/share> <password> <options>*

The smbclient program is a versatile UNIX client which provides functionality similar to ftp.

##### smbcontrol

smbcontrol -i *<options>*

smbcontrol *<options> <destination> <messagetype> <parameters>*

The smbcontrol program sends control messages to running smbd or nmbd daemons. Executing smbcontrol -i runs commands interactively until a blank line or a 'q' is entered.

##### smbpasswd

smbpasswd *<options> <username> <password>*

The smbpasswd program manages encrypted passwords. This program can be run by a superuser to change any user's password as well as by an ordinary user to change their own Samba password.

##### smbspool

smbspool *<job> <user> <title> <copies> <options> <filename>*

The smbspool program is a CUPS-compatible printing interface to Samba. Although designed for use with CUPS printers, smbspool can work with non-CUPS printers as well.

##### smbstatus

smbstatus *<options>*

The smbstatus program displays the status of current connections to a Samba server.

##### smbtar

smbtar *<options>*

The smbtar program performs backup and restores of Windows-based share files and directories to a local tape archive. Though similar to the tar command, the two are not compatible.

##### testparm

testparm *<options> <filename> <hostname IP\_address>*

The testparm program checks the syntax of the smb.conf file. If your smb.conf file is in the default location (/etc/samba/smb.conf) you do not need to specify the location. Specifying the hostname and IP address to the testparm program verifies that the hosts.allow and host.deny files are configured correctly. The testparm program also displays a summary of your smb.conf file and the server's role (stand-alone, domain, etc.) after testing. This is convenient when debugging as it excludes comments and concisely presents information for experienced administrators to read.

For example:

**testparm**

Load smb config files from /etc/samba/smb.conf

Processing section "[homes]"

Processing section "[printers]"

Processing section "[tmp]"

Processing section "[html]"

Loaded services file OK.

Server role: ROLE\_STANDALONE

Press enter to see a dump of your service definitions

**<enter>**

# Global parameters

[global]

workgroup = MYGROUP

server string = Samba Server

security = SHARE

log file = /var/log/samba/%m.log

max log size = 50

socket options = TCP\_NODELAY SO\_RCVBUF=8192 SO\_SNDBUF=8192

dns proxy = No

[homes]

comment = Home Directories

read only = No

browseable = No

[printers]

comment = All Printers

path = /var/spool/samba

printable = Yes

browseable = No

[tmp]

comment = Wakko tmp

path = /tmp

guest only = Yes

[html]

comment = Wakko www

path = /var/www/html

force user = andriusb

force group = users

read only = No

guest only = Yes

##### wbinfo

wbinfo *<options>*

The wbinfo program displays information from the winbindd daemon. The winbindd daemon must be running for wbinfo to work.

**Chapter 20. Dynamic Host Configuration Protocol (DHCP)**

[20.1. Why Use DHCP?](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-dhcp.html#s1-dhcp-why)

[20.2. Configuring a DHCP Server](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-dhcp-configuring-server.html)

[20.3. Configuring a DHCP Client](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-dhcp-configuring-client.html)

[20.4. Additional Resources](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-dhcp-additional-resources.html)

Dynamic Host Configuration Protocol (DHCP) is a network protocol that automatically assigns TCP/IP information to client machines. Each DHCP client connects to the centrally located DHCP server, which returns that client's network configuration (including the IP address, gateway, and DNS servers).

**20.1. Why Use DHCP?**

DHCP is useful for automatic configuration of client network interfaces. When configuring the client system, the administrator chooses DHCP instead of specifying an IP address, netmask, gateway, or DNS servers. The client retrieves this information from the DHCP server. DHCP is also useful if an administrator wants to change the IP addresses of a large number of systems. Instead of reconfiguring all the systems, he can just edit one DHCP configuration file on the server for the new set of IP addresses. If the DNS servers for an organization changes, the changes are made on the DHCP server, not on the DHCP clients. When the administrator restarts the network or reboots the clients, the changes will go into effect.

If an organization has a functional DHCP server properly connected to a network, laptops and other mobile computer users can move these devices from office to office.

## 20.2. Configuring a DHCP Server

To configure a DHCP server, you must create the dhcpd.conf configuration file in the /etc/ directory. A sample file can be found at /usr/share/doc/dhcp-<*version*>/dhcpd.conf.sample.

DHCP also uses the file /var/lib/dhcpd/dhcpd.leases to store the client lease database. Refer to [Section 20.2.2, “Lease Database”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-dhcp-configuring-server.html#lease-database) for more information.

### 20.2.1. Configuration File

The first step in configuring a DHCP server is to create the configuration file that stores the network information for the clients.Use this file to declare options and global options for client systems.

The configuration file can contain extra tabs or blank lines for easier formatting. Keywords are case-insensitive and lines beginning with a hash mark (#) are considered comments.

Two DNS update schemes are currently implemented — the ad-hoc DNS update mode and the interim DHCP-DNS interaction draft update mode. If and when these two are accepted as part of the Internet Engineering Task Force (IETF) standards process, there will be a third mode — the standard DNS update method. You must configure the DNS server for compatibility with these schemes. Version 3.0b2pl11 and previous versions used the ad-hoc mode; however, it has been deprecated. To keep the same behavior, add the following line to the top of the configuration file:

ddns-update-style ad-hoc;

To use the recommended mode, add the following line to the top of the configuration file:

ddns-update-style interim;

Refer to the dhcpd.conf man page for details about the different modes.

There are two types of statements in the configuration file:

* Parameters — State how to perform a task, whether to perform a task, or what network configuration options to send to the client.
* Declarations — Describe the topology of the network, describe the clients, provide addresses for the clients, or apply a group of parameters to a group of declarations.

The parameters that start with the keyword option are reffered to as *options*. These options control DHCP options; whereas, parameters configure values that are not optional or control how the DHCP server behaves.

Parameters (including options) declared before a section enclosed in curly brackets ({ }) are considered global parameters. Global parameters apply to all the sections below it.

## Important

If the configuration file is changed, the changes do not take effect until the DHCP daemon is restarted with the command service dhcpd restart.

## Tip

Instead of changing a DHCP configuration file and restarting the service each time, using the omshell command provides an interactive way to connect to, query, and change the configuration of a DHCP server. By using omshell, all changes can be made while the server is running. For more information on omshell, refer to the omshell man page.

In [Example 20.1, “Subnet Declaration”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-dhcp-configuring-server.html#subnet), the routers, subnet-mask, domain-name, domain-name-servers, and time-offset options are used for any host statements declared below it.

Additionally, a subnet can be declared, a subnet declaration must be included for every subnet in the network. If it is not, the DHCP server fails to start.

In this example, there are global options for every DHCP client in the subnet and a range declared. Clients are assigned an IP address within the range.

subnet 192.168.1.0 netmask 255.255.255.0 {

option routers 192.168.1.254;

option subnet-mask 255.255.255.0;

option domain-name "example.com";

option domain-name-servers 192.168.1.1;

option time-offset -18000; # Eastern Standard Time

range 192.168.1.10 192.168.1.100;

}

**Example 20.1. Subnet Declaration**

All subnets that share the same physical network should be declared within a shared-network declaration as shown in [Example 20.2, “Shared-network Declaration”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-dhcp-configuring-server.html#shared-network). Parameters within the shared-network, but outside the enclosed subnet declarations, are considered to be global parameters. The name of the shared-network must be a descriptive title for the network, such as using the title 'test-lab' to describe all the subnets in a test lab environment.

shared-network name {

option domain-name "test.redhat.com";

option domain-name-servers ns1.redhat.com, ns2.redhat.com;

option routers 192.168.0.254;

more parameters for EXAMPLE shared-network

subnet 192.168.1.0 netmask 255.255.252.0 {

parameters for subnet

range 192.168.1.1 192.168.1.254;

}

subnet 192.168.2.0 netmask 255.255.252.0 {

parameters for subnet

range 192.168.2.1 192.168.2.254;

}

}

**Example 20.2. Shared-network Declaration**

As demonstrated in [Example 20.3, “Group Declaration”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-dhcp-configuring-server.html#group), the group declaration is used to apply global parameters to a group of declarations. For example, shared networks, subnets, and hosts can be grouped.

group {

option routers 192.168.1.254;

option subnet-mask 255.255.255.0;

option domain-name "example.com";

option domain-name-servers 192.168.1.1;

option time-offset -18000; # Eastern Standard Time

host apex {

option host-name "apex.example.com";

hardware ethernet 00:A0:78:8E:9E:AA;

fixed-address 192.168.1.4;

}

host raleigh {

option host-name "raleigh.example.com";

hardware ethernet 00:A1:DD:74:C3:F2;

fixed-address 192.168.1.6;

}

}

**Example 20.3. Group Declaration**

To configure a DHCP server that leases a dynamic IP address to a system within a subnet, modify [Example 20.4, “Range Parameter”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-dhcp-configuring-server.html#dynamic-ip) with your values. It declares a default lease time, maximum lease time, and network configuration values for the clients. This example assigns IP addresses in the range 192.168.1.10 and 192.168.1.100 to client systems.

default-lease-time 600;

max-lease-time 7200;

option subnet-mask 255.255.255.0;

option broadcast-address 192.168.1.255;

option routers 192.168.1.254;

option domain-name-servers 192.168.1.1, 192.168.1.2;

option domain-name "example.com";

subnet 192.168.1.0 netmask 255.255.255.0 {

range 192.168.1.10 192.168.1.100;

}

**Example 20.4. Range Parameter**

To assign an IP address to a client based on the MAC address of the network interface card, use the hardware ethernet parameter within a host declaration. As demonstrated in [Example 20.5, “Static IP Address using DHCP”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-dhcp-configuring-server.html#static-ip), the host apex declaration specifies that the network interface card with the MAC address 00:A0:78:8E:9E:AA always receives the IP address 192.168.1.4.

Note that the optional parameter host-name can also be used to assign a host name to the client.

host apex {

option host-name "apex.example.com";

hardware ethernet 00:A0:78:8E:9E:AA;

fixed-address 192.168.1.4;

}

**Example 20.5. Static IP Address using DHCP**

## Tip

The sample configuration file provided can be used as a starting point and custom configuration options can be added to it. To copy it to the proper location, use the following command:

cp /usr/share/doc/dhcp-*<version-number>*/dhcpd.conf.sample /etc/dhcpd.conf

(where *<version-number>* is the DHCP version number).

For a complete list of option statements and what they do, refer to the dhcp-options man page.

### 20.2.2. Lease Database

On the DHCP server, the file /var/lib/dhcpd/dhcpd.leases stores the DHCP client lease database. Do not change this file. DHCP lease information for each recently assigned IP address is automatically stored in the lease database. The information includes the length of the lease, to whom the IP address has been assigned, the start and end dates for the lease, and the MAC address of the network interface card that was used to retrieve the lease.

All times in the lease database are in Coordinated Universal Time (UTC), not local time.

The lease database is recreated from time to time so that it is not too large. First, all known leases are saved in a temporary lease database. The dhcpd.leases file is renamed dhcpd.leases~ and the temporary lease database is written to dhcpd.leases.

The DHCP daemon could be killed or the system could crash after the lease database has been renamed to the backup file but before the new file has been written. If this happens, the dhcpd.leases file does not exist, but it is required to start the service. Do not create a new lease file. If you do, all old leases are lost which causes many problems. The correct solution is to rename the dhcpd.leases~ backup file to dhcpd.leases and then start the daemon.

### 20.2.3. Starting and Stopping the Server

## Important

When the DHCP server is started for the first time, it fails unless the dhcpd.leases file exists. Use the command touch /var/lib/dhcpd/dhcpd.leases to create the file if it does not exist.

If the same server is also running BIND as a DNS server, this step is not necessary, as starting the named service automatically checks for a dhcpd.leases file.

To start the DHCP service, use the command /sbin/service dhcpd start. To stop the DHCP server, use the command /sbin/service dhcpd stop.

By default, the DHCP service does not start at boot time. To configure the daemon to start automatically at boot time, refer to [Chapter 15, *Controlling Access to Services*](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-services.html).

If more than one network interface is attached to the system, but the DHCP server should only be started on one of the interfaces, configure the DHCP server to start only on that device. In /etc/sysconfig/dhcpd, add the name of the interface to the list of DHCPDARGS:

# Command line options here

DHCPDARGS=eth0

This is useful for a firewall machine with two network cards. One network card can be configured as a DHCP client to retrieve an IP address to the Internet. The other network card can be used as a DHCP server for the internal network behind the firewall. Specifying only the network card connected to the internal network makes the system more secure because users can not connect to the daemon via the Internet.

Other command line options that can be specified in /etc/sysconfig/dhcpd include:

* -p *<portnum>* — Specifies the UDP port number on which dhcpd should listen. The default is port 67. The DHCP server transmits responses to the DHCP clients at a port number one greater than the UDP port specified. For example, if the default port 67 is used, the server listens on port 67 for requests and responses to the client on port 68. If a port is specified here and the DHCP relay agent is used, the same port on which the DHCP relay agent should listen must be specified. Refer to [Section 20.2.4, “DHCP Relay Agent”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-dhcp-configuring-server.html#dhcp-relay-agent) for details.
* -f — Runs the daemon as a foreground process. This is mostly used for debugging.
* -d — Logs the DHCP server daemon to the standard error descriptor. This is mostly used for debugging. If this is not specified, the log is written to /var/log/messages.
* -cf *<filename>* — Specifies the location of the configuration file. The default location is /etc/dhcpd.conf.
* -lf *<filename>* — Specifies the location of the lease database file. If a lease database file already exists, it is very important that the same file be used every time the DHCP server is started. It is strongly recommended that this option only be used for debugging purposes on non-production machines. The default location is /var/lib/dhcpd/dhcpd.leases.
* -q — Do not print the entire copyright message when starting the daemon.

### 20.2.4. DHCP Relay Agent

The DHCP Relay Agent (dhcrelay) allows for the relay of DHCP and BOOTP requests from a subnet with no DHCP server on it to one or more DHCP servers on other subnets.

When a DHCP client requests information, the DHCP Relay Agent forwards the request to the list of DHCP servers specified when the DHCP Relay Agent is started. When a DHCP server returns a reply, the reply is broadcast or unicast on the network that sent the original request.

The DHCP Relay Agent listens for DHCP requests on all interfaces unless the interfaces are specified in /etc/sysconfig/dhcrelay with the INTERFACES directive.

To start the DHCP Relay Agent, use the command service dhcrelay start.

## 20.3. Configuring a DHCP Client

The first step for configuring a DHCP client is to make sure the kernel recognizes the network interface card. Most cards are recognized during the installation process and the system is configured to use the correct kernel module for the card. If a card is added after installation, **Kudzu** [[9](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-dhcp-configuring-client.html#ftn.id2892432)] will recognize it and prompt you for the proper kernel module (Be sure to check the Hardware Compatibility List at <http://hardware.redhat.com/hcl/>). If either the installation program or kudzu does not recognize the network card, you can load the correct kernel module (refer to [Chapter 40, *General Parameters and Modules*](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-modules.html) for details).

To configure a DHCP client manually, modify the /etc/sysconfig/network file to enable networking and the configuration file for each network device in the /etc/sysconfig/network-scripts directory. In this directory, each device should have a configuration file named ifcfg-eth0, where eth0 is the network device name.

The /etc/sysconfig/network file should contain the following line:

NETWORKING=yes

The NETWORKING variable must be set to yes if you want networking to start at boot time.

The /etc/sysconfig/network-scripts/ifcfg-eth0 file should contain the following lines:

DEVICE=eth0

BOOTPROTO=dhcp

ONBOOT=yes

A configuration file is needed for each device to be configured to use DHCP.

Other options for the network script includes:

* DHCP\_HOSTNAME — Only use this option if the DHCP server requires the client to specify a hostname before receiving an IP address. (The DHCP server daemon in Red Hat Enterprise Linux does not support this feature.)
* PEERDNS=*<answer>*, where *<answer>* is one of the following:
  + yes — Modify /etc/resolv.conf with information from the server. If using DHCP, then yes is the default.
  + no — Do not modify /etc/resolv.conf.
* SRCADDR=*<address>*, where *<address>* is the specified source IP address for outgoing packets.
* USERCTL=*<answer>*, where *<answer>* is one of the following:
  + yes — Non-root users are allowed to control this device.
  + no — Non-root users are not allowed to control this device.

If you prefer using a graphical interface, refer to [Chapter 14, *Network Configuration*](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-network-config.html) for instructions on using the **Network Administration Tool** to configure a network interface to use DHCP.

## Tip

For advanced configurations of client DHCP options such as protocol timing, lease requirements and requests, dynamic DNS support, aliases, as well as a wide variety of values to override, prepend, or append to client-side configurations, refer to the dhclient and dhclient.conf man pages.

**Chapter 21. Apache HTTP Server**

[21.1. Apache HTTP Server 2.2](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-httpd.html#s1-httpd-v2)

[21.2. Migrating Apache HTTP Server Configuration Files](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-httpd-mig.html)

[21.3. Starting and Stopping httpd](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-startstop.html)

[21.4. Apache HTTP Server Configuration](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-config-ui.html)

[21.5. Configuration Directives in httpd.conf](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-config.html)

[21.6. Adding Modules](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-addmods.html)

[21.7. Virtual Hosts](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-virtualhosts.html)

[21.8. Apache HTTP Secure Server Configuration](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-httpd-secure-server.html)

[21.9. Additional Resources](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-additional-resources.html)

The Apache HTTP Server is a robust, commercial-grade open source Web server developed by the Apache Software Foundation (<http://www.apache.org/>). Red Hat Enterprise Linux includes the Apache HTTP Server 2.2 as well as a number of server modules designed to enhance its functionality.

The default configuration file installed with the Apache HTTP Server works without alteration for most situations. This chapter outlines many of the directives found within its configuration file (/etc/httpd/conf/httpd.conf) to aid those who require a custom configuration or need to convert a configuration file from the older Apache HTTP Server 1.3 format.

**Warning**

If using the graphical **HTTP Configuration Tool** (*system-config-httpd* ), *do not* hand edit the Apache HTTP Server's configuration file as the **HTTP Configuration Tool** regenerates this file whenever it is used.

**21.1. Apache HTTP Server 2.2**

There are important differences between the Apache HTTP Server 2.2 and version 2.0 (version 2.0 shipped with Red Hat Enterprise Linux 4 and earlier). This section reviews some of the features of Apache HTTP Server 2.2 and outlines important changes. If you are upgrading from version 1.3, you should also read the instructions on migrating from version 1.3 to version 2.0. For instructions on migrating a version 1.3 configuration file to the 2.0 format, refer to [Section 21.2.2, “Migrating Apache HTTP Server 1.3 Configuration Files to 2.0”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-httpd-mig.html#s3-httpd-v2-mig).

**21.1.1. Features of Apache HTTP Server 2.2**

Apache HTTP Server 2.2 features the following improvements over version 2.0 :

* Improved caching modules (mod\_cache, mod\_disk\_cache, mod\_mem\_cache).
* A new structure for authentication and authorization support, replacing the authentication modules provided in previous versions.
* Support for proxy load balancing (mod\_proxy\_balancer)
* support for handling large files (namely, greater than 2GB) on 32-bit platforms

The following changes have been made to the default httpd configuration:

* The mod\_cern\_meta and mod\_asis modules are no longer loaded by default.
* The mod\_ext\_filter module is now loaded by default.

## 21.2. Migrating Apache HTTP Server Configuration Files

### 21.2.1. Migrating Apache HTTP Server 2.0 Configuration Files

This section outlines migration from version 2.0 to 2.2. If you are migrating from version 1.3, please refer to [Section 21.2.2, “Migrating Apache HTTP Server 1.3 Configuration Files to 2.0”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-httpd-mig.html#s3-httpd-v2-mig).

* Configuration files and startup scripts from version 2.0 need minor adjustments particularly in module names which may have changed. Third party modules which worked in version 2.0 can also work in version 2.2 but need to be recompiled before you load them. Key modules that need to be noted are authentication and authorization modules. For each of the modules which has been renamed the [LoadModule](http://httpd.apache.org/docs/2.2/mod/mod_so.html#loadmodule) line will need to be updated.
* The mod\_userdir module will only act on requests if you provide a UserDir directive indicating a directory name. If you wish to maintain the procedures used in version 2.0, add the directive UserDir public\_html in your configuration file.
* To enable SSL, edit the httpd.conf file adding the necessary mod\_ssl directives. Use apachectl start as apachectl startssl is unavailable in version 2.2. You can view an example of SSL configuration for httpd in conf/extra/httpd-ssl.conf.
* To test your configuration it is advisable to use service httpd configtest which will detect configuration errors.

More information on upgrading from version 2.0 to 2.2 can be found on <http://httpd.apache.org/docs/2.2/upgrading.html>.

### 21.2.2. Migrating Apache HTTP Server 1.3 Configuration Files to 2.0

This section details migrating an Apache HTTP Server 1.3 configuration file to be utilized by Apache HTTP Server 2.0.

If upgrading to Red Hat Enterprise Linux 5 from Red Hat Enterprise Linux 2.1, note that the new stock configuration file for the Apache HTTP Server 2.0 package is installed as /etc/httpd/conf/httpd.conf.rpmnew and the original version 1.3 httpd.conf is left untouched. It is entirely up to you whether to use the new configuration file and migrate the old settings to it, or use the existing file as a base and modify it to suit; however, some parts of the file have changed more than others and a mixed approach is generally the best. The stock configuration files for both version 1.3 and 2.0 are divided into three sections.

If the /etc/httpd/conf/httpd.conf file is a modified version of the newly installed default and a saved a copy of the original configuration file is available, it may be easiest to invoke the diff command, as in the following example (logged in as root):

diff -u httpd.conf.orig httpd.conf | less

This command highlights any modifications made. If a copy of the original file is not available, extract it from an RPM package using the rpm2cpio and cpio commands, as in the following example:

rpm2cpio apache-*<version-number>*.i386.rpm | cpio -i --make

In the above command, replace *<version-number>* with the version number for the apache package.

Finally, it is useful to know that the Apache HTTP Server has a testing mode to check for configuration errors. To use access it, type the following command:

apachectl configtest

#### 21.2.2.1. Global Environment Configuration

The global environment section of the configuration file contains directives which affect the overall operation of the Apache HTTP Server, such as the number of concurrent requests it can handle and the locations of the various files. This section requires a large number of changes and should be based on the Apache HTTP Server 2.0 configuration file, while migrating the old settings into it.

##### 21.2.2.1.1. Interface and Port Binding

The BindAddress and Port directives no longer exist; their functionality is now provided by a more flexible Listen directive.

If Port 80 was set in the 1.3 version configuration file, change it to Listen 80 in the 2.0 configuration file. If Port was set to some value other than 80, then append the port number to the contents of the ServerName directive.

For example, the following is a sample Apache HTTP Server 1.3 directive:

Port 123 ServerName www.example.com

To migrate this setting to Apache HTTP Server 2.0, use the following structure:

**Listen** 123 ServerName www.example.com:**123**

For more on this topic, refer to the following documentation on the Apache Software Foundation's website:

* <http://httpd.apache.org/docs-2.0/mod/mpm_common.html#listen>
* <http://httpd.apache.org/docs-2.0/mod/core.html#servername>

##### 21.2.2.1.2. Server-Pool Size Regulation

When the Apache HTTP Server accepts requests, it dispatches child processes or threads to handle them. This group of child processes or threads is known as a *server-pool*. Under Apache HTTP Server 2.0, the responsibility for creating and maintaining these server-pools has been abstracted to a group of modules called *Multi-Processing Modules* (*MPMs*). Unlike other modules, only one module from the MPM group can be loaded by the Apache HTTP Server. There are three MPM modules that ship with 2.0: prefork, worker, and perchild. Currently only the prefork and worker MPMs are available, although the perchild MPM may be available at a later date.

The original Apache HTTP Server 1.3 behavior has been moved into the prefork MPM. The prefork MPM accepts the same directives as Apache HTTP Server 1.3, so the following directives may be migrated directly:

* StartServers
* MinSpareServers
* MaxSpareServers
* MaxClients
* MaxRequestsPerChild

The worker MPM implements a multi-process, multi-threaded server providing greater scalability. When using this MPM, requests are handled by threads, conserving system resources and allowing large numbers of requests to be served efficiently. Although some of the directives accepted by the worker MPM are the same as those accepted by the prefork MPM, the values for those directives should not be transfered directly from an Apache HTTP Server 1.3 installation. It is best to instead use the default values as a guide, then experiment to determine what values work best.

## Important

To use the worker MPM, create the file /etc/sysconfig/httpd and add the following directive:

HTTPD=/usr/sbin/httpd.worker

For more on the topic of MPMs, refer to the following documentation on the Apache Software Foundation's website:

* <http://httpd.apache.org/docs-2.0/mpm.html>

##### 21.2.2.1.3. Dynamic Shared Object (DSO) Support

There are many changes required here, and it is highly recommended that anyone trying to modify an Apache HTTP Server 1.3 configuration to suit version 2.0 (as opposed to migrating the changes into the version 2.0 configuration) copy this section from the stock Apache HTTP Server 2.0 configuration file.

Those who do not want to copy the section from the stock Apache HTTP Server 2.0 configuration should note the following:

* The AddModule and ClearModuleList directives no longer exist. These directives where used to ensure that modules could be enabled in the correct order. The Apache HTTP Server 2.0 API allows modules to specify their ordering, eliminating the need for these two directives.
* The order of the LoadModule lines are no longer relevant in most cases.
* Many modules have been added, removed, renamed, split up, or incorporated into others.
* LoadModule lines for modules packaged in their own RPMs (mod\_ssl, php, mod\_perl, and the like) are no longer necessary as they can be found in their relevant files within the /etc/httpd/conf.d/ directory.
* The various HAVE\_XXX definitions are no longer defined.

## Important

If modifying the original file, note that it is of paramount importance that the httpd.conf contains the following directive:

Include conf.d/\*.conf

Omission of this directive results in the failure of all modules packaged in their own RPMs (such as mod\_perl, php, and mod\_ssl).

##### 21.2.2.1.4. Other Global Environment Changes

The following directives have been removed from Apache HTTP Server 2.0's configuration:

* *ServerType* — The Apache HTTP Server can only be run as ServerType standalone making this directive irrelevant.
* *AccessConfig* and *ResourceConfig* — These directives have been removed as they mirror the functionality of the Include directive. If the AccessConfig and ResourceConfig directives are set, replace them with Include directives.

To ensure that the files are read in the order implied by the older directives, the Include directives should be placed at the end of the httpd.conf, with the one corresponding to ResourceConfig preceding the one corresponding to AccessConfig. If using the default values, include them explicitly as conf/srm.conf and conf/access.conf files.

#### 21.2.2.2. Main Server Configuration

The main server configuration section of the configuration file sets up the main server, which responds to any requests that are not handled by a virtual host defined within a <VirtualHost> container. Values here also provide defaults for any <VirtualHost> containers defined.

The directives used in this section have changed little between Apache HTTP Server 1.3 and version 2.0. If the main server configuration is heavily customized, it may be easier to modify the existing configuration file to suit Apache HTTP Server 2.0. Users with only lightly customized main server sections should migrate their changes into the default 2.0 configuration.

##### 21.2.2.2.1. UserDir Mapping

The UserDir directive is used to enable URLs such as http://example.com/~bob/ to map to a subdirectory within the home directory of the user bob, such as /home/bob/public\_html/. A side-effect of this feature allows a potential attacker to determine whether a given username is present on the system. For this reason, the default configuration for Apache HTTP Server 2.0 disables this directive.

To enable UserDir mapping, change the directive in httpd.conf from:

UserDir disable

to the following:

UserDir **public\_html**

For more on this topic, refer to the following documentation on the Apache Software Foundation's website:

* <http://httpd.apache.org/docs-2.0/mod/mod_userdir.html#userdir>

##### 21.2.2.2.2. Logging

The following logging directives have been removed:

* AgentLog
* RefererLog
* RefererIgnore

However, agent and referrer logs are still available using the CustomLog and LogFormat directives.

For more on this topic, refer to the following documentation on the Apache Software Foundation's website:

* <http://httpd.apache.org/docs-2.0/mod/mod_log_config.html#customlog>
* <http://httpd.apache.org/docs-2.0/mod/mod_log_config.html#logformat>

##### 21.2.2.2.3. Directory Indexing

The deprecated FancyIndexing directive has now been removed. The same functionality is available through the FancyIndexing option within the IndexOptions directive.

The VersionSort option to the IndexOptions directive causes files containing version numbers to be sorted in a more natural way. For example, httpd-2.0.6.tar appears before httpd-2.0.36.tar in a directory index page.

The defaults for the ReadmeName and HeaderName directives have changed from README and HEADER to README.html and HEADER.html.

For more on this topic, refer to the following documentation on the Apache Software Foundation's website:

* <http://httpd.apache.org/docs-2.0/mod/mod_autoindex.html#indexoptions>
* <http://httpd.apache.org/docs-2.0/mod/mod_autoindex.html#readmename>
* <http://httpd.apache.org/docs-2.0/mod/mod_autoindex.html#headername>

##### 21.2.2.2.4. Content Negotiation

The CacheNegotiatedDocs directive now takes the argument on or off. Existing instances of CacheNegotiatedDocs should be replaced with CacheNegotiatedDocs on.

For more on this topic, refer to the following documentation on the Apache Software Foundation's website:

* <http://httpd.apache.org/docs-2.0/mod/mod_negotiation.html#cachenegotiateddocs>

##### 21.2.2.2.5. Error Documents

To use a hard-coded message with the ErrorDocument directive, the message should be enclosed in a pair of double quotation marks ", rather than just preceded by a double quotation mark as required in Apache HTTP Server 1.3.

For example, the following is a sample Apache HTTP Server 1.3 directive:

ErrorDocument 404 "The document was not found

To migrate an ErrorDocument setting to Apache HTTP Server 2.0, use the following structure:

ErrorDocument 404 "The document was not found"

Note the trailing double quote in the previous ErrorDocument directive example.

For more on this topic, refer to the following documentation on the Apache Software Foundation's website:

* <http://httpd.apache.org/docs-2.0/mod/core.html#errordocument>

#### 21.2.2.3. Virtual Host Configuration

The contents of all <VirtualHost> containers should be migrated in the same way as the main server section as described in [Section 21.2.2.2, “Main Server Configuration”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-httpd-mig.html#s2-httpd-v2-mig-main).

## Important

Note that SSL/TLS virtual host configuration has been moved out of the main server configuration file and into /etc/httpd/conf.d/ssl.conf.

* <http://httpd.apache.org/docs-2.0/vhosts/>

#### 21.2.2.4. Modules and Apache HTTP Server 2.0

In Apache HTTP Server 2.0, the module system has been changed to allow modules to be chained together or combined in new and interesting ways. *Common Gateway Interface* (*CGI*) scripts, for example, can generate server-parsed HTML documents which can then be processed by mod\_include. This opens up a tremendous number of possibilities with regards to how modules can be combined to achieve a specific goal.

The way this works is that each request is served by exactly one *handler* module followed by zero or more *filter* modules.

Under Apache HTTP Server 1.3, for example, a Perl script would be handled in its entirety by the Perl module (mod\_perl). Under Apache HTTP Server 2.0, the request is initially handled by the core module — which serves static files — and is then filtered by mod\_perl.

Exactly how to use this, and all other new features of Apache HTTP Server 2.0, is beyond the scope of this document; however, the change has ramifications if the PATH\_INFO directive is used for a document which is handled by a module that is now implemented as a filter, as each contains trailing path information after the true file name. The core module, which initially handles the request, does not by default understand PATH\_INFO and returns 404 Not Found errors for requests that contain such information. As an alternative, use the AcceptPathInfo directive to coerce the core module into accepting requests with PATH\_INFO.

The following is an example of this directive:

AcceptPathInfo on

For more on this topic, refer to the following documentation on the Apache Software Foundation's website:

* <http://httpd.apache.org/docs-2.0/mod/core.html#acceptpathinfo>
* <http://httpd.apache.org/docs-2.0/handler.html>
* <http://httpd.apache.org/docs-2.0/filter.html>

##### 21.2.2.4.1. The suexec Module

In Apache HTTP Server 2.0, the mod\_suexec module uses the SuexecUserGroup directive, rather than the User and Group directives, which is used for configuring virtual hosts. The User and Group directives can still be used in general, but are deprecated for configuring virtual hosts.

For example, the following is a sample Apache HTTP Server 1.3 directive:

<VirtualHost vhost.example.com:80> User someone Group somegroup </VirtualHost>

To migrate this setting to Apache HTTP Server 2.0, use the following structure:

<VirtualHost vhost.example.com:80> SuexecUserGroup someone somegroup </VirtualHost>

##### 21.2.2.4.2. The mod\_ssl Module

The configuration for mod\_ssl has been moved from the httpd.conf file into the /etc/httpd/conf.d/ssl.conf file. For this file to be loaded, and for mod\_ssl to work, the statement Include conf.d/\*.conf must be in the httpd.conf file as described in [Section 21.2.2.1.3, “Dynamic Shared Object (DSO) Support”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-httpd-mig.html#s3-httpd-v2-mig-dso).

ServerName directives in SSL virtual hosts must explicitly specify the port number.

For example, the following is a sample Apache HTTP Server 1.3 directive:

<VirtualHost \_default\_:443> # General setup for the virtual host ServerName ssl.example.name ... </VirtualHost>

To migrate this setting to Apache HTTP Server 2.0, use the following structure:

<VirtualHost \_default\_:443> # General setup for the virtual host ServerName ssl.host.name**:443** ... </VirtualHost>

It is also important to note that both the SSLLog and SSLLogLevel directives have been removed. The mod\_ssl module now obeys the ErrorLog and LogLevel directives. Refer to [ErrorLog](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-config.html#s2-apache-errorlog) and [LogLevel](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-config.html#s2-apache-loglevel) for more information about these directives.

For more on this topic, refer to the following documentation on the Apache Software Foundation's website:

* <http://httpd.apache.org/docs-2.0/mod/mod_ssl.html>
* <http://httpd.apache.org/docs-2.0/vhosts/>

##### 21.2.2.4.3. The mod\_proxy Module

Proxy access control statements are now placed inside a <Proxy> block rather than a <Directory proxy:>.

The caching functionality of the old mod\_proxy has been split out into the following three modules:

* mod\_cache
* mod\_disk\_cache
* mod\_mem\_cache

These generally use directives similar to the older versions of the mod\_proxy module, but it is advisable to verify each directive before migrating any cache settings.

For more on this topic, refer to the following documentation on the Apache Software Foundation's website:

* <http://httpd.apache.org/docs-2.0/mod/mod_proxy.html>

##### 21.2.2.4.4. The mod\_include Module

The mod\_include module is now implemented as a filter and is therefore enabled differently. Refer to [Section 21.2.2.4, “Modules and Apache HTTP Server 2.0”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-httpd-mig.html#s2-httpd-v2-mig-mod) for more about filters.

For example, the following is a sample Apache HTTP Server 1.3 directive:

AddType text/html .shtml AddHandler server-parsed .shtml

To migrate this setting to Apache HTTP Server 2.0, use the following structure:

AddType text/html .shtml **AddOutputFilter INCLUDES** .shtml

Note that the Options +Includes directive is still required for the <Directory> container or in a .htaccess file.

For more on this topic, refer to the following documentation on the Apache Software Foundation's website:

* <http://httpd.apache.org/docs-2.0/mod/mod_include.html>

##### 21.2.2.4.5. The mod\_auth\_dbm and mod\_auth\_db Modules

Apache HTTP Server 1.3 supported two authentication modules, mod\_auth\_db and mod\_auth\_dbm, which used Berkeley Databases and DBM databases respectively. These modules have been combined into a single module named mod\_auth\_dbm in Apache HTTP Server 2.0, which can access several different database formats. To migrate from mod\_auth\_db, configuration files should be adjusted by replacing AuthDBUserFile and AuthDBGroupFile with the mod\_auth\_dbm equivalents, AuthDBMUserFile and AuthDBMGroupFile. Also, the directive AuthDBMType DB must be added to indicate the type of database file in use.

The following example shows a sample mod\_auth\_db configuration for Apache HTTP Server 1.3:

<Location /private/> AuthType Basic AuthName "My Private Files" AuthDBUserFile /var/www/authdb require valid-user </Location>

To migrate this setting to version 2.0 of Apache HTTP Server, use the following structure:

<Location /private/> AuthType Basic AuthName "My Private Files" **AuthDBMUserFile** /var/www/authdb **AuthDBMType DB** require valid-user </Location>

Note that the AuthDBMUserFile directive can also be used in .htaccess files.

The dbmmanage Perl script, used to manipulate username and password databases, has been replaced by htdbm in Apache HTTP Server 2.0. The htdbm program offers equivalent functionality and, like mod\_auth\_dbm, can operate a variety of database formats; the -T option can be used on the command line to specify the format to use.

[Table 21.1, “Migrating from dbmmanage to htdbm”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-httpd-mig.html#table-htdbm) shows how to migrate from a DBM-format database to htdbm format using dbmmanage.

| **Action** | **dbmmanage command (1.3)** | **Equivalent htdbm command (2.0)** |
| --- | --- | --- |
| Add user to database (using given password) | dbmmanage authdb add username password | htdbm -b -TDB authdb username password |
| Add user to database (prompts for password) | dbmmanage authdb adduser username | htdbm -TDB authdb username |
| Remove user from database | dbmmanage authdb delete username | htdbm -x -TDB authdb username |
| List users in database | dbmmanage authdb view | htdbm -l -TDB authdb |
| Verify a password | dbmmanage authdb check username | htdbm -v -TDB authdb username |

**Table 21.1. Migrating from dbmmanage to htdbm**

The -m and -s options work with both dbmmanage and htdbm, enabling the use of the MD5 or SHA1 algorithms for hashing passwords, respectively.

When creating a new database with htdbm, the -c option must be used.

For more on this topic, refer to the following documentation on the Apache Software Foundation's website:

* <http://httpd.apache.org/docs-2.0/mod/mod_auth_dbm.html>

##### 21.2.2.4.6. The mod\_perl Module

The configuration for mod\_perl has been moved from httpd.conf into the file /etc/httpd/conf.d/perl.conf. For this file to be loaded, and hence for mod\_perl to work, the statement Include conf.d/\*.conf must be included in httpd.conf as described in [Section 21.2.2.1.3, “Dynamic Shared Object (DSO) Support”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-httpd-mig.html#s3-httpd-v2-mig-dso).

Occurrences of Apache:: in httpd.conf must be replaced with ModPerl::. Additionally, the manner in which handlers are registered has been changed.

This is a sample Apache HTTP Server 1.3 mod\_perl configuration:

<Directory /var/www/perl> SetHandler perl-script PerlHandler Apache::Registry Options +ExecCGI </Directory>

This is the equivalent mod\_perl for Apache HTTP Server 2.0:

<Directory /var/www/perl> SetHandler perl-script **PerlResponseHandler ModPerl::Registry** Options +ExecCGI </Directory>

Most modules for mod\_perl 1.x should work without modification with mod\_perl 2.x. XS modules require recompilation and may require minor Makefile modifications.

##### 21.2.2.4.7. The mod\_python Module

Configuration for mod\_python has moved from httpd.conf to the /etc/httpd/conf.d/python.conf file. For this file to be loaded, and hence for mod\_python to work, the statement Include conf.d/\*.conf must be in httpd.conf as described in [Section 21.2.2.1.3, “Dynamic Shared Object (DSO) Support”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-httpd-mig.html#s3-httpd-v2-mig-dso).

##### 21.2.2.4.8. PHP

The configuration for PHP has been moved from httpd.conf into the file /etc/httpd/conf.d/php.conf. For this file to be loaded, the statement Include conf.d/\*.conf must be in httpd.conf as described in [Section 21.2.2.1.3, “Dynamic Shared Object (DSO) Support”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-httpd-mig.html#s3-httpd-v2-mig-dso).

## Note

Any PHP configuration directives used in Apache HTTP Server 1.3 are now fully compatible, when migrating to Apache HTTP Server 2.0 on Red Hat Enterprise Linux 5.

In PHP version 4.2.0 and later the default set of predefined variables which are available in the global scope has changed. Individual input and server variables are, by default, no longer placed directly into the global scope. This change may cause scripts to break. Revert to the old behavior by setting register\_globals to On in the file /etc/php.ini.

For more on this topic, refer to the following URL for details concerning the global scope changes:

* <http://www.php.net/release_4_1_0.php>

##### 21.2.2.4.9. The mod\_authz\_ldap Module

Red Hat Enterprise Linux ships with the mod\_authz\_ldap module for the Apache HTTP Server. This module uses the short form of the distinguished name for a subject and the issuer of the client SSL certificate to determine the distinguished name of the user within an LDAP directory. It is also capable of authorizing users based on attributes of that user's LDAP directory entry, determining access to assets based on the user and group privileges of the asset, and denying access for users with expired passwords. The mod\_ssl module is required when using the mod\_authz\_ldap module.

## Important

The mod\_authz\_ldap module does not authenticate a user to an LDAP directory using an encrypted password hash. This functionality is provided by the experimental mod\_auth\_ldap module. Refer to the mod\_auth\_ldap module documentation online at <http://httpd.apache.org/docs-2.0/mod/mod_auth_ldap.html> for details on the status of this module.

The /etc/httpd/conf.d/authz\_ldap.conf file configures the mod\_authz\_ldap module.

Refer to /usr/share/doc/mod\_authz\_ldap-*<version>*/index.html (replacing *<version>* with the version number of the package) or <http://authzldap.othello.ch/> for more information on configuring the mod\_authz\_ldap third party module.

## 21.3. Starting and Stopping httpd

After installing the httpd package, review the Apache HTTP Server's documentation available online at <http://httpd.apache.org/docs/2.2/>.

The httpd RPM installs the /etc/init.d/httpd script, which can be accessed using the /sbin/service command.

Starting httpd using the apachectl control script sets the environmental variables in /etc/sysconfig/httpd and starts httpd. You can also set the environment variables using the init script.

To start the server using the apachectl control script as root type:

apachectl start

You can also start httpd using /sbin/service httpd start. This starts httpd but does not set the environment variables. If you are using the default Listen directive in httpd.conf, which is port 80, you will need to have root privileges to start the apache server.

To stop the server, as root type:

apachectl stop

You can also stop httpd using /sbin/service httpd stop. The restart option is a shorthand way of stopping and then starting the Apache HTTP Server.

You can restart the server as root by typing:

apachectl restart

or:

/sbin/service httpd restart

Apache will display a message on the console or in the ErrorLog if it encounters an error while starting.

By default, the httpd service does not start automatically at boot time. If you would wish to have Apache startup at boot time, you will need to add a call to apachectl in your startup files within the rc.N directory. A typical file used is rc.local. As this starts Apache as root, it is recommended to properly configure your security and authentication before adding this call.

You can also configure the httpd service to start up at boot time, using an initscript utility, such as /sbin/chkconfig, **/usr/sbin/ntsysv**, or the **Services Configuration Tool** program.

You can also display the status of your httpd server by typing:

apachectl status

The status module mod\_status however needs to be enabled in your httpd.conf configuration file for this to work. For more details on mod\_status can be found on <http://httpd.apache.org/docs/2.2/mod/mod_status.html>.

## Note

If running the Apache HTTP Server as a secure server, the secure server's password is required after the machine boots when using an encrypted private SSL key.

You can find more information on <http://httpd.apache.org/docs/2.2/ssl>

**21.4. Apache HTTP Server Configuration**

The **HTTP Configuration Tool** allows you to configure the /etc/httpd/conf/httpd.conf configuration file for the Apache HTTP Server. It does not use the old srm.conf or access.conf configuration files; leave them empty. Through the graphical interface, you can configure directives such as virtual hosts, logging attributes, and maximum number of connections. To start the HTTD Configuration Tool, click on System => Administration => Server Settings => HTTP.

Only modules provided with Red Hat Enterprise Linux can be configured with the **HTTP Configuration Tool**. If additional modules are installed, they can not be configured using this tool.

**Caution**

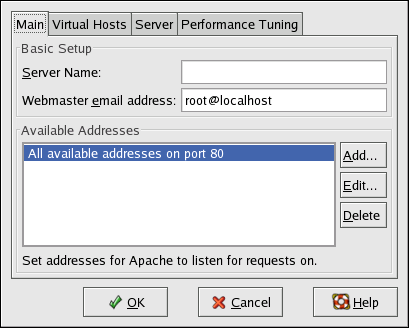
Do not edit the /etc/httpd/conf/httpd.conf configuration file by hand if you wish to use this tool. The **HTTP Configuration Tool** generates this file after you save your changes and exit the program. If you want to add additional modules or configuration options that are not available in **HTTP Configuration Tool**, you cannot use this tool.

The general steps for configuring the Apache HTTP Server using the **HTTP Configuration Tool** are as follows:

1. Configure the basic settings under the **Main** tab.
2. Click on the **Virtual Hosts** tab and configure the default settings.
3. Under the **Virtual Hosts** tab, configure the Default Virtual Host.
4. To serve more than one URL or virtual host, add any additional virtual hosts.
5. Configure the server settings under the **Server** tab.
6. Configure the connections settings under the **Performance Tuning** tab.
7. Copy all necessary files to the DocumentRoot and cgi-bin directories.
8. Exit the application and select to save your settings.

**21.4.1. Basic Settings**

Use the **Main** tab to configure the basic server settings.



**Figure 21.1. Basic Settings**

Enter a fully qualified domain name that you have the right to use in the **Server Name** text area. This option corresponds to the [ServerName](http://httpd.apache.org/docs/2.2/mod/core.html#servername) directive in httpd.conf. The ServerName directive sets the hostname of the Web server. It is used when creating redirection URLs. If you do not define a server name, the Web server attempts to resolve it from the IP address of the system. The server name does not have to be the domain name resolved from the IP address of the server. For example, you might set the server name to www.example.com while the server's real DNS name is foo.example.com.

Enter the email address of the person who maintains the Web server in the **Webmaster email address** text area. This option corresponds to the [ServerAdmin](http://httpd.apache.org/docs/2.2/mod/core.html#serveradmin) directive in httpd.conf. If you configure the server's error pages to contain an email address, this email address is used so that users can report a problem to the server's administrator. The default value is root@localhost.

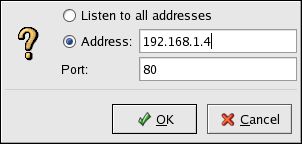
Use the **Available Addresses** area to define the ports on which the server accepts incoming requests. This option corresponds to the [Listen](http://httpd.apache.org/docs/2.2/mod/mpm_common.html#listen) directive in httpd.conf. By default, Red Hat configures the Apache HTTP Server to listen to port 80 for non-secure Web communications.

Click the **Add** button to define additional ports on which to accept requests. A window as shown in [Figure 21.2, “Available Addresses”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-config-ui.html#httpd-listen) appears. Either choose the **Listen to all addresses** option to listen to all IP addresses on the defined port or specify a particular IP address over which the server accepts connections in the **Address** field. Only specify one IP address per port number. To specify more than one IP address with the same port number, create an entry for each IP address. If at all possible, use an IP address instead of a domain name to prevent a DNS lookup failure. Refer to <http://httpd.apache.org/docs/2.2/dns-caveats.html> for more information about *Issues Regarding DNS and Apache*.

Entering an asterisk (\*) in the **Address** field is the same as choosing **Listen to all addresses**. Clicking the **Edit** button in the **Available Addresses** frame shows the same window as the **Add** button except with the fields populated for the selected entry. To delete an entry, select it and click the **Delete** button.

**Tip**

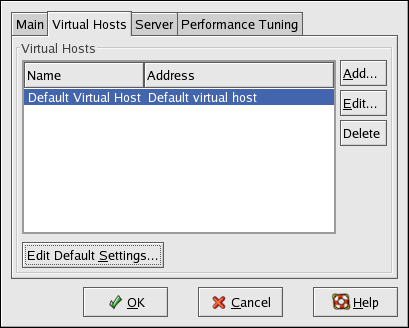
If you set the server to listen to a port under 1024, you must be root to start it. For port 1024 and above, httpd can be started as a regular user.



**Figure 21.2. Available Addresses**

**21.4.2. Default Settings**

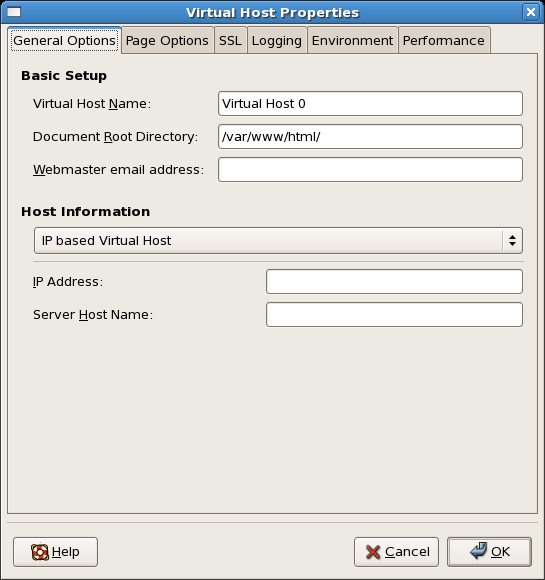
After defining the **Server Name**, **Webmaster email address**, and **Available Addresses**, click the **Virtual Hosts** tab. The figure below illustrates the **Virtual Hosts** tab.



**Figure 21.3. Virtual Hosts Tab**

Clicking on **Edit** will display the **Virtual Host Properties** window from which you can set your preferred settings. To add new settings, click on the **Add** button which will also display the **Virtual Host Properties** window. Clicking on the **Edit Default Settings** button, displays the **Virtual Host Properties** window without the **General Options** tab.

In the **General Options** tab, you can change the hostname, the document root directory and also set the webmaster's email address. In the Host information, you can set the Virtual Host's IP Address and Host Name. The figure below illustrates the **General Options** tab.

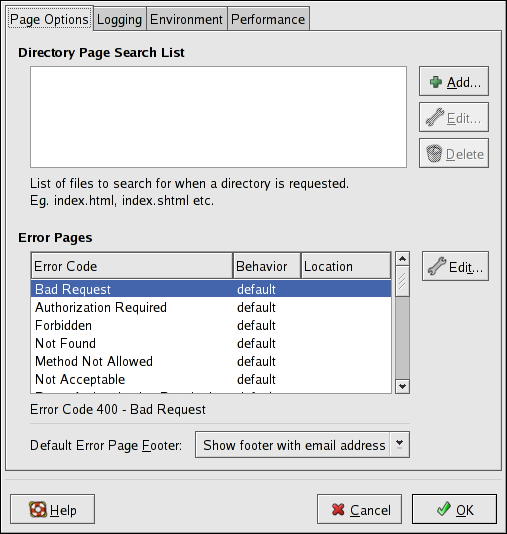


**Figure 21.4. General Options**

If you add a virtual host, the settings you configure for the virtual host take precedence for that virtual host. For a directive not defined within the virtual host settings, the default value is used.

**21.4.2.1. Site Configuration**

The figure below illustrates the **Page Options**tab from which you can configure the **Directory Page Search List** and **Error Pages**. If you are unsure of these settings, do not modify them.



**Figure 21.5. Site Configuration**

The entries listed in the **Directory Page Search List** define the [DirectoryIndex](http://httpd.apache.org/docs/2.2/mod/mod_dir.html#directoryindex) directive. The DirectoryIndex is the default page served by the server when a user requests an index of a directory by specifying a forward slash (/) at the end of the directory name.

For example, when a user requests the page http://www.example.com/this\_directory/, they are going to get either the DirectoryIndex page, if it exists, or a server-generated directory list. The server tries to find one of the files listed in the DirectoryIndex directive and returns the first one it finds. If it does not find any of these files and if Options Indexes is set for that directory, the server generates and returns a list, in HTML format, of the subdirectories and files in the directory.

Use the **Error Code** section to configure Apache HTTP Server to redirect the client to a local or external URL in the event of a problem or error. This option corresponds to the [ErrorDocument](http://httpd.apache.org/docs/2.2/mod/core.html#errordocument) directive. If a problem or error occurs when a client tries to connect to the Apache HTTP Server, the default action is to display the short error message shown in the **Error Code** column. To override this default configuration, select the error code and click the **Edit** button. Choose **Default** to display the default short error message. Choose **URL** to redirect the client to an external URL and enter a complete URL, including the http://, in the **Location** field. Choose **File** to redirect the client to an internal URL and enter a file location under the document root for the Web server. The location must begin the a slash (/) and be relative to the Document Root.

For example, to redirect a 404 Not Found error code to a webpage that you created in a file called 404.html, copy 404.html to *DocumentRoot*/../error/404.html. In this case, *DocumentRoot* is the Document Root directory that you have defined (the default is /var/www/html/). If the Document Root is left as the default location, the file should be copied to /var/www/error/404.html. Then, choose **File** as the Behavior for **404 - Not Found** error code and enter /error/404.html as the **Location**.

From the **Default Error Page Footer** menu, you can choose one of the following options:

* **Show footer with email address** — Display the default footer at the bottom of all error pages along with the email address of the website maintainer specified by the [ServerAdmin](http://httpd.apache.org/docs/2.2/mod/core.html#serveradmin) directive.
* **Show footer** — Display just the default footer at the bottom of error pages.
* **No footer** — Do not display a footer at the bottom of error pages.

**21.4.2.2. SSL Support**

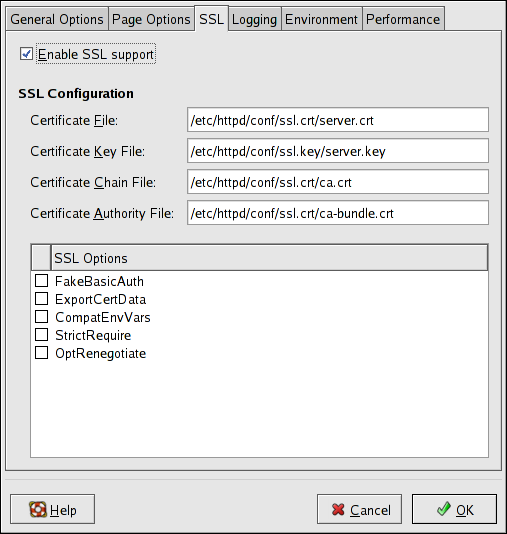
The mod\_ssl enables encryption of the HTTP protocol over SSL. SSL (Secure Sockets Layer) protocol is used for communication and encryption over TCP/IP networks. The SSL tab enables you to configure SSL for your server. To configure SSL you need to provide the path to your:

* Certificate file - equivalent to using the SSLCertificateFile directive which points the path to the PEM (Privacy Enhanced Mail)-encoded server certificate file.
* Key file - equivalent to using the SSLCertificateKeyFile directive which points the path to the PEM-encoded server private key file.
* Certificate chain file - equivalent to using the SSLCertificateChainFile directive which points the path to the certificate file containing all the server's chain of certificates.
* Certificate authority file - is an encrypted file used to confirm the authenticity or identity of parties communicating with the server.

You can find out more about configuration directives for SSL on <http://httpd.apache.org/docs/2.2/mod/directives.html#S>. You also need to determine which SSL options to enable. These are equivalent to using the SSLOptions with the following options:

* FakeBasicAuth - enables standard authentication methods used by Apache. This means that the Client X509 certificate's Subject Distinguished Name (DN) is translated into a basic HTTP username.
* ExportCertData - creates CGI environment variables in SSL\_SERVER\_CERT, SSL\_CLIENT\_CERT and SSL\_CLIENT\_CERT\_CHAIN\_n where n is a number 0,1,2,3,4... These files are used for more certificate checks by CGI scripts.
* CompatEnvVars - enables backward compatibility for Apache SSL by adding CGI environment variables.
* StrictRequire - enables strict access which forces denial of access whenever the SSLRequireSSL and SSLRequire directives indicate access is forbiden.
* OptRenegotiate - enables avoidance of unnecessary handshakes by mod\_ssl which also performs safe parameter checks. It is recommended to enable OptRenegotiate on a per directory basis.

More information on the above SSL options can be found on <http://httpd.apache.org/docs/2.2/mod/mod_ssl.html#ssloptions>. The figure below illustrates the SSL tab and the options discussed above.



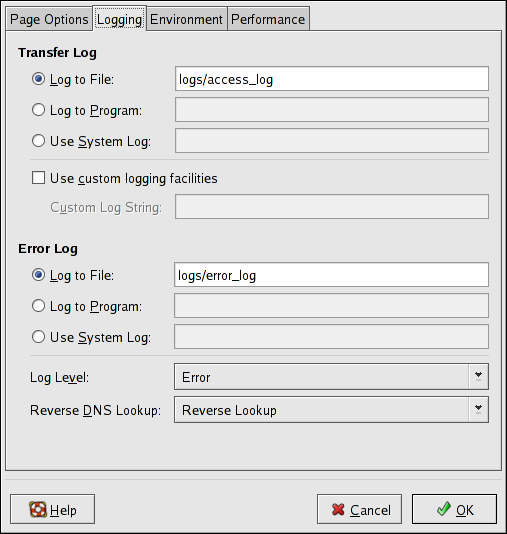
**Figure 21.6. SSL**

**21.4.2.3. Logging**

Use the **Logging** tab to configure options for specific transfer and error logs.

By default, the server writes the transfer log to the /var/log/httpd/access\_log file and the error log to the /var/log/httpd/error\_log file.

The transfer log contains a list of all attempts to access the Web server. It records the IP address of the client that is attempting to connect, the date and time of the attempt, and the file on the Web server that it is trying to retrieve. Enter the name of the path and file in which to store this information. If the path and file name do not start with a slash (/), the path is relative to the server root directory as configured. This option corresponds to the [TransferLog](http://httpd.apache.org/docs/2.2/mod/mod_log_config.html#transferlog) directive.



**Figure 21.7. Logging**

You can configure a custom log format by checking **Use custom logging facilities** and entering a custom log string in the **Custom Log String** field. This configures the [LogFormat](http://httpd.apache.org/docs/2.2/mod/mod_log_config.html#logformat) directive. Refer to [http://httpd.apache.org/docs/2.2/mod/mod\_log\_config.html#logformat](http://httpd.apache.org/docs-2.0/mod/mod_log_config.html#formats) for details on the format of this directive.

The error log contains a list of any server errors that occur. Enter the name of the path and file in which to store this information. If the path and file name do not start with a slash (/), the path is relative to the server root directory as configured. This option corresponds to the [ErrorLog](http://httpd.apache.org/docs/2.2/mod/core.html#errorlog) directive.

Use the **Log Level** menu to set the verbosity of the error messages in the error logs. It can be set (from least verbose to most verbose) to emerg, alert, crit, error, warn, notice, info or debug. This option corresponds to the [LogLevel](http://httpd.apache.org/docs/2.2/mod/core.html#loglevel) directive.

The value chosen with the **Reverse DNS Lookup** menu defines the [HostnameLookups](http://httpd.apache.org/docs/2.2/mod/core.html#hostnamelookups) directive. Choosing **No Reverse Lookup** sets the value to off. Choosing **Reverse Lookup** sets the value to on. Choosing **Double Reverse Lookup** sets the value to double.

If you choose **Reverse Lookup**, your server automatically resolves the IP address for each connection which requests a document from your Web server. Resolving the IP address means that your server makes one or more connections to the DNS in order to find out the hostname that corresponds to a particular IP address.

If you choose **Double Reverse Lookup**, your server performs a double-reverse DNS. In other words, after a reverse lookup is performed, a forward lookup is performed on the result. At least one of the IP addresses in the forward lookup must match the address from the first reverse lookup.

Generally, you should leave this option set to **No Reverse Lookup**, because the DNS requests add a load to your server and may slow it down. If your server is busy, the effects of trying to perform these reverse lookups or double reverse lookups may be quite noticeable.

Reverse lookups and double reverse lookups are also an issue for the Internet as a whole. Each individual connection made to look up each hostname adds up. Therefore, for your own Web server's benefit, as well as for the Internet's benefit, you should leave this option set to **No Reverse Lookup**.

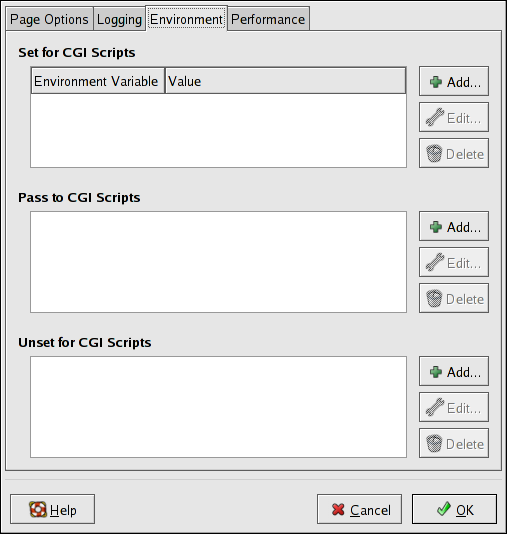
**21.4.2.4. Environment Variables**

Use the **Environment** tab to configure options for specific variables to set, pass, or unset for CGI scripts.

Sometimes it is necessary to modify environment variables for CGI scripts or server-side include (SSI) pages. The Apache HTTP Server can use the mod\_env module to configure the environment variables which are passed to CGI scripts and SSI pages. Use the **Environment Variables** page to configure the directives for this module.

Use the **Set for CGI Scripts** section to set an environment variable that is passed to CGI scripts and SSI pages. For example, to set the environment variable MAXNUM to 50, click the **Add** button inside the **Set for CGI Script** section, as shown in [Figure 21.8, “Environment Variables”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-config-ui.html#httpd-environment-variables-screen), and type **MAXNUM** in the **Environment Variable** text field and **50** in the **Value to set** text field. Click **OK** to add it to the list. The **Set for CGI Scripts** section configures the [SetEnv](http://httpd.apache.org/docs/2.2/mod/mod_env.html#setenv) directive.

Use the **Pass to CGI Scripts** section to pass the value of an environment variable when the server is first started to CGI scripts. To see this environment variable, type the command env at a shell prompt. Click the **Add** button inside the **Pass to CGI Scripts** section and enter the name of the environment variable in the resulting dialog box. Click **OK** to add it to the list. The **Pass to CGI Scripts** section configures the [PassEnv](http://httpd.apache.org/docs/2.2/mod/mod_env.html#passenv) directive.



**Figure 21.8. Environment Variables**

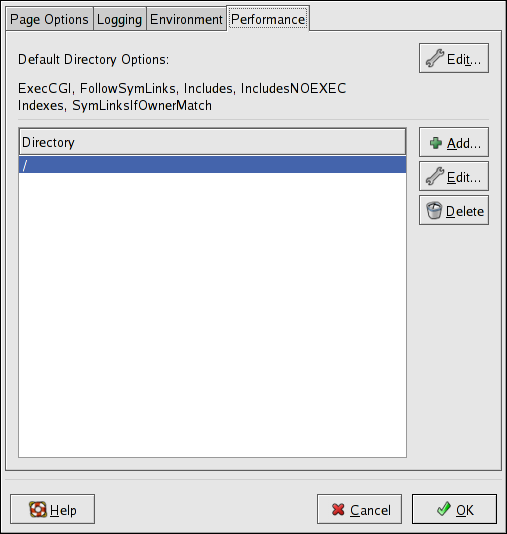
To remove an environment variable so that the value is not passed to CGI scripts and SSI pages, use the **Unset for CGI Scripts** section. Click **Add** in the **Unset for CGI Scripts** section, and enter the name of the environment variable to unset. Click **OK** to add it to the list. This corresponds to the [UnsetEnv](http://httpd.apache.org/docs/2.2/mod/mod_env.html#unsetenv) directive.

To edit any of these environment values, select it from the list and click the corresponding **Edit** button. To delete any entry from the list, select it and click the corresponding **Delete** button.

To learn more about environment variables in the Apache HTTP Server, refer to the following: <http://httpd.apache.org/docs/2.2/env.html>

**21.4.2.5. Directories**

Use the **Directories** page in the **Performance** tab to configure options for specific directories. This corresponds to the [<Directory>](http://httpd.apache.org/docs/2.2/mod/core.html#directory) directive.



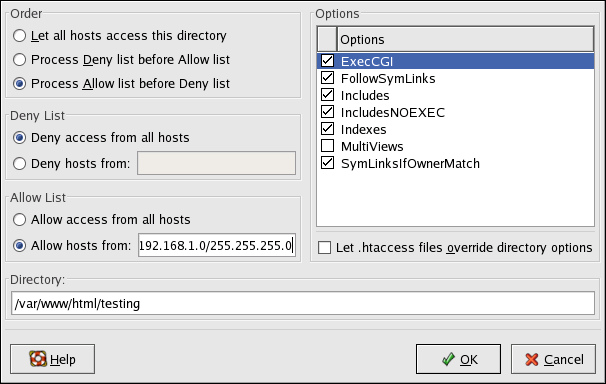
**Figure 21.9. Directories**

Click the **Edit** button in the top right-hand corner to configure the **Default Directory Options** for all directories that are not specified in the **Directory** list below it. The options that you choose are listed as the [Options](http://httpd.apache.org/docs/2.2/mod/core.html#options) directive within the [<Directory>](http://httpd.apache.org/docs/2.2/mod/core.html#directory) directive. You can configure the following options:

* **ExecCGI** — Allow execution of CGI scripts. CGI scripts are not executed if this option is not chosen.
* **FollowSymLinks** — Allow symbolic links to be followed.
* **Includes** — Allow server-side includes.
* **IncludesNOEXEC** — Allow server-side includes, but disable the #exec and #include commands in CGI scripts.
* **Indexes** — Display a formatted list of the directory's contents, if no DirectoryIndex (such as index.html) exists in the requested directory.
* **Multiview** — Support content-negotiated multiviews; this option is disabled by default.
* **SymLinksIfOwnerMatch** — Only follow symbolic links if the target file or directory has the same owner as the link.

To specify options for specific directories, click the **Add** button beside the **Directory** list box. A window as shown in [Figure 21.10, “Directory Settings”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-config-ui.html#httpd-directories-add) appears. Enter the directory to configure in the **Directory** text field at the bottom of the window. Select the options in the right-hand list and configure the [Order](http://httpd.apache.org/docs-2.0/mod/mod_access.html#order) directive with the left-hand side options. The Order directive controls the order in which allow and deny directives are evaluated. In the **Allow hosts from** and **Deny hosts from** text field, you can specify one of the following:

* Allow all hosts — Type **all** to allow access to all hosts.
* Partial domain name — Allow all hosts whose names match or end with the specified string.
* Full IP address — Allow access to a specific IP address.
* A subnet — Such as **192.168.1.0/255.255.255.0**
* A network CIDR specification — such as **10.3.0.0/16**



**Figure 21.10. Directory Settings**

If you check the **Let .htaccess files override directory options**, the configuration directives in the .htaccess file take precedence.

## 21.5. Configuration Directives in httpd.conf

The Apache HTTP Server configuration file is /etc/httpd/conf/httpd.conf. The httpd.conf file is well-commented and mostly self-explanatory. The default configuration works for most situations; however, it is a good idea to become familiar some of the more important configuration options.

## Warning

With the release of Apache HTTP Server 2.2, many configuration options have changed. If migrating from version 1.3 to 2.2, please firstly read [Section 21.2.2, “Migrating Apache HTTP Server 1.3 Configuration Files to 2.0”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-httpd-mig.html#s3-httpd-v2-mig).

### 21.5.1. General Configuration Tips

If configuring the Apache HTTP Server, edit /etc/httpd/conf/httpd.conf and then either reload, restart, or stop and start the httpd process as outlined in [Section 21.3, “Starting and Stopping httpd”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-startstop.html).

Before editing httpd.conf, make a copy the original file. Creating a backup makes it easier to recover from mistakes made while editing the configuration file.

If a mistake is made and the Web server does not work correctly, first review recently edited passages in httpd.conf to verify there are no typos.

Next look in the Web server's error log, /var/log/httpd/error\_log. The error log may not be easy to interpret, depending on your level of expertise. However, the last entries in the error log should provide useful information.

The following subsections contain a list of short descriptions for many of the directives included in httpd.conf. These descriptions are not exhaustive. For more information, refer to the Apache documentation online at <http://httpd.apache.org/docs/2.2/>.

For more information about mod\_ssl directives, refer to the documentation online at <http://httpd.apache.org/docs/2.2/mod/mod_ssl.html>.

##### AccessFileName

AccessFileName names the file which the server should use for access control information in each directory. The default is .htaccess.

Immediately after the AccessFileName directive, a set of Files tags apply access control to any file beginning with a .ht. These directives deny Web access to any .htaccess files (or other files which begin with .ht) for security reasons.

##### Action

Action specifies a MIME content type and CGI script pair, so that when a file of that media type is requested, a particular CGI script is executed.

##### AddDescription

When using FancyIndexing as an IndexOptions parameter, the AddDescription directive can be used to display user-specified descriptions for certain files or file types in a server generated directory listing. The AddDescription directive supports listing specific files, wildcard expressions, or file extensions.

##### AddEncoding

AddEncoding names file name extensions which should specify a particular encoding type. AddEncoding can also be used to instruct some browsers to uncompress certain files as they are downloaded.

##### AddHandler

AddHandler maps file extensions to specific handlers. For example, the cgi-script handler can be matched with the extension .cgi to automatically treat a file ending with .cgi as a CGI script. The following is a sample AddHandler directive for the .cgi extension.

AddHandler cgi-script .cgi

This directive enables CGIs outside of the cgi-bin to function in any directory on the server which has the ExecCGI option within the directories container. Refer to [Directory](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-config.html#s2-apache-directory) for more information about setting the ExecCGI option for a directory.

In addition to CGI scripts, the AddHandler directive is used to process server-parsed HTML and image-map files.

##### AddIcon

AddIcon specifies which icon to show in server generated directory listings for files with certain extensions. For example, the Web server is set to show the icon binary.gif for files with .bin or .exe extensions.

##### AddIconByEncoding

This directive names icons which are displayed by files with MIME encoding in server generated directory listings. For example, by default, the Web server shows the compressed.gif icon next to MIME encoded x-compress and x-gzip files in server generated directory listings.

##### AddIconByType

This directive names icons which are displayed next to files with MIME types in server generated directory listings. For example, the server shows the icon text.gif next to files with a mime-type of text, in server generated directory listings.

##### AddLanguage

AddLanguage associates file name extensions with specific languages. This directive is useful for Apache HTTP Servers which serve content in multiple languages based on the client Web browser's language settings.

##### AddType

Use the AddType directive to define or override a default MIME type and file extension pairs. The following example directive tells the Apache HTTP Server to recognize the .tgz file extension:

AddType application/x-tar .tgz

##### Alias

The Alias setting allows directories outside the DocumentRoot directory to be accessible. Any URL ending in the alias automatically resolves to the alias' path. By default, one alias for an icons/ directory is already set up. An icons/ directory can be accessed by the Web server, but the directory is not in the DocumentRoot.

##### Allow

Allow specifies which client can access a given directory. The client can be all, a domain name, an IP address, a partial IP address, a network/netmask pair, and so on. The DocumentRoot directory is configured to Allow requests from all, meaning everyone has access.

##### AllowOverride

The AllowOverride directive sets whether any Options can be overridden by the declarations in an .htaccess file. By default, both the root directory and the DocumentRoot are set to allow no .htaccess overrides.

##### BrowserMatch

The BrowserMatch directive allows the server to define environment variables and take appropriate actions based on the User-Agent HTTP header field — which identifies the client's Web browser type. By default, the Web server uses BrowserMatch to deny connections to specific browsers with known problems and also to disable keepalives and HTTP header flushes for browsers that are known to have problems with those actions.

##### Cache Directives

A number of commented cache directives are supplied by the default Apache HTTP Server configuration file. In most cases, uncommenting these lines by removing the hash mark (#) from the beginning of the line is sufficient. The following, however, is a list of some of the more important cache-related directives.

* CacheEnable — Specifies whether the cache is a disk, memory, or file descriptor cache. By default CacheEnable configures a disk cache for URLs at or below /.
* CacheRoot — Specifies the name of the directory containing cached files. The default CacheRoot is the /var/httpd/proxy/ directory.
* CacheSize — Specifies how much space the cache can use in kilobytes. The default CacheSize is 5 KB.

The following is a list of some of the other common cache-related directives.

* CacheMaxExpire — Specifies how long HTML documents are retained (without a reload from the originating Web server) in the cache. The default is 24 hours (86400 seconds).
* CacheLastModifiedFactor — Specifies the creation of an expiry (expiration) date for a document which did not come from its originating server with its own expiry set. The default CacheLastModifiedFactor is set to 0.1, meaning that the expiry date for such documents equals one-tenth of the amount of time since the document was last modified.
* CacheDefaultExpire — Specifies the expiry time in hours for a document that was received using a protocol that does not support expiry times. The default is set to 1 hour (3600 seconds).
* NoProxy — Specifies a space-separated list of subnets, IP addresses, domains, or hosts whose content is not cached. This setting is most useful for Intranet sites.

##### CacheNegotiatedDocs

By default, the Web server asks proxy servers not to cache any documents which were negotiated on the basis of content (that is, they may change over time or because of the input from the requester). If CacheNegotiatedDocs is set to on, this function is disabled and proxy servers are allowed to cache such documents.

##### CustomLog

CustomLog identifies the log file and the log file format. By default, the access log is recorded to the /var/log/httpd/access\_log file while errors are recorded in the /var/log/httpd/error\_log file.

The default CustomLog format is the combined log file format, as illustrated here:

*remotehost rfc931 user date "request" status bytes referrer user-agent*

##### DefaultIcon

DefaultIcon specifies the icon displayed in server generated directory listings for files which have no other icon specified. The unknown.gif image file is the default.

##### DefaultType

DefaultType sets a default content type for the Web server to use for documents whose MIME types cannot be determined. The default is text/plain.

##### Deny

Deny works similar to Allow, except it specifies who is denied access. The DocumentRoot is not configured to Deny requests from anyone by default.

##### Directory

<Directory /path/to/directory> and </Directory> tags create a container used to enclose a group of configuration directives which apply only to a specific directory and its subdirectories. Any directive which is applicable to a directory may be used within Directory tags.

By default, very restrictive parameters are applied to the root directory (/), using the Options (refer to [Options](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-config.html#s2-apache-options)) and AllowOverride (refer to [AllowOverride](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-config.html#s2-apache-allowoverride)) directives. Under this configuration, any directory on the system which needs more permissive settings has to be explicitly given those settings.

In the default configuration, another Directory container is configured for the DocumentRoot which assigns less rigid parameters to the directory tree so that the Apache HTTP Server can access the files residing there.

The Directory container can be also be used to configure additional cgi-bin directories for server-side applications outside of the directory specified in the ScriptAlias directive (refer to [ScriptAlias](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-config.html#s2-apache-scriptalias) for more information).

To accomplish this, the Directory container must set the ExecCGI option for that directory.

For example, if CGI scripts are located in /home/my\_cgi\_directory, add the following Directory container to the httpd.conf file:

<Directory /home/my\_cgi\_directory> Options +ExecCGI </Directory>

Next, the AddHandler directive must be uncommented to identify files with the .cgi extension as CGI scripts. Refer to [AddHandler](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-config.html#s2-apache-addhandler) for instructions on setting AddHandler.

For this to work, permissions for CGI scripts, and the entire path to the scripts, must be set to 0755.

##### DirectoryIndex

The DirectoryIndex is the default page served by the server when a user requests an index of a directory by specifying a forward slash (/) at the end of the directory name.

When a user requests the page http://*example*/*this\_directory*/, they get either the DirectoryIndex page, if it exists, or a server-generated directory list. The default for DirectoryIndex is index.html and the index.html.var type map. The server tries to find either of these files and returns the first one it finds. If it does not find one of these files and Options Indexes is set for that directory, the server generates and returns a listing, in HTML format, of the subdirectories and files within the directory, unless the directory listing feature is turned off.

##### DocumentRoot

DocumentRoot is the directory which contains most of the HTML files which are served in response to requests. The default DocumentRoot, for both the non-secure and secure Web servers, is the /var/www/html directory. For example, the server might receive a request for the following document:

http://example.com/foo.html

The server looks for the following file in the default directory:

/var/www/html/foo.html

To change the DocumentRoot so that it is not shared by the secure and the non-secure Web servers, refer to [Section 21.7, “Virtual Hosts”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-virtualhosts.html).

##### ErrorDocument

The ErrorDocument directive associates an HTTP response code with a message or a URL to be sent back to the client. By default, the Web server outputs a simple and usually cryptic error message when an error occurs. The ErrorDocument directive forces the Web server to instead output a customized message or page.

## Important

To be valid, the message must be enclosed in a pair of double quotes ".

##### ErrorLog

ErrorLog specifies the file where server errors are logged. By default, this directive is set to /var/log/httpd/error\_log.

##### ExtendedStatus

The ExtendedStatus directive controls whether Apache generates basic (off) or detailed server status information (on), when the server-status handler is called. The server-status handler is called using Location tags. More information on calling server-status is included in [Location](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-config.html#s2-apache-location).

##### Group

Specifies the group name of the Apache HTTP Server processes.

This directive has been deprecated for the configuration of virtual hosts.

By default, Group is set to apache.

##### HeaderName

HeaderName names the file which, if it exists in the directory, is prepended to the start of server generated directory listings. Like ReadmeName, the server tries to include it as an HTML document if possible or in plain text if not.

##### HostnameLookups

HostnameLookups can be set to on, off, or double. If HostnameLookups is set to on, the server automatically resolves the IP address for each connection. Resolving the IP address means that the server makes one or more connections to a DNS server, adding processing overhead. If HostnameLookups is set to double, the server performs a double-reverse DNS look up adding even more processing overhead.

To conserve resources on the server, HostnameLookups is set to off by default.

If hostnames are required in server log files, consider running one of the many log analyzer tools that perform the DNS lookups more efficiently and in bulk when rotating the Web server log files.

##### IfDefine

The IfDefine tags surround configuration directives that are applied if the "test" stated in the IfDefine tag is true. The directives are ignored if the test is false.

The test in the IfDefine tags is a parameter name (for example, HAVE\_PERL). If the parameter is defined, meaning that it is provided as an argument to the server's start-up command, then the test is true. In this case, when the Web server is started, the test is true and the directives contained in the IfDefine tags are applied.

##### IfModule

<IfModule> and </IfModule> tags create a conditional container which are only activated if the specified module is loaded. Directives within the IfModule container are processed under one of two conditions. The directives are processed if the module contained within the starting <IfModule> tag is loaded. Or, if an exclamation point **!** appears before the module name, the directives are processed only if the module specified in the <IfModule> tag is not loaded.

For more information about Apache HTTP Server modules, refer to [Section 21.6, “Adding Modules”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-addmods.html).

##### Include

Include allows other configuration files to be included at runtime.

The path to these configuration files can be absolute or relative to the ServerRoot.

## Important

For the server to use individually packaged modules, such as mod\_ssl, mod\_perl, and php, the following directive must be included in Section 1: Global Environment of httpd.conf:

Include conf.d/\*.conf

##### IndexIgnore

IndexIgnore lists file extensions, partial file names, wildcard expressions, or full file names. The Web server does not include any files which match any of those parameters in server generated directory listings.

##### IndexOptions

IndexOptions controls the appearance of server generated directing listings, by adding icons, file descriptions, and so on. If Options Indexes is set (refer to [Options](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-config.html#s2-apache-options)), the Web server generates a directory listing when the Web server receives an HTTP request for a directory without an index.

First, the Web server looks in the requested directory for a file matching the names listed in the DirectoryIndex directive (usually, index.html). If an index.html file is not found, Apache HTTP Server creates an HTML directory listing of the requested directory. The appearance of this directory listing is controlled, in part, by the IndexOptions directive.

The default configuration turns on FancyIndexing. This means that a user can re-sort a directory listing by clicking on column headers. Another click on the same header switches from ascending to descending order. FancyIndexing also shows different icons for different files, based upon file extensions.

The AddDescription option, when used in conjunction with FancyIndexing, presents a short description for the file in server generated directory listings.

IndexOptions has a number of other parameters which can be set to control the appearance of server generated directories. The IconHeight and IconWidth parameters require the server to include HTML HEIGHT and WIDTH tags for the icons in server generated webpages. The IconsAreLinks parameter combines the graphical icon with the HTML link anchor, which contains the URL link target.

##### KeepAlive

KeepAlive sets whether the server allows more than one request per connection and can be used to prevent any one client from consuming too much of the server's resources.

By default Keepalive is set to off. If Keepalive is set to on and the server becomes very busy, the server can quickly spawn the maximum number of child processes. In this situation, the server slows down significantly. If Keepalive is enabled, it is a good idea to set the the KeepAliveTimeout low (refer to [KeepAliveTimeout](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-config.html#s2-apache-keepalivetimeout) for more information about the KeepAliveTimeout directive) and monitor the /var/log/httpd/error\_log log file on the server. This log reports when the server is running out of child processes.

##### KeepAliveTimeout

KeepAliveTimeout sets the number of seconds the server waits after a request has been served before it closes the connection. Once the server receives a request, the Timeout directive applies instead. The KeepAliveTimeout directive is set to 15 seconds by default.

##### LanguagePriority

LanguagePriority sets precedence for different languages in case the client Web browser has no language preference set.

##### Listen

The Listen command identifies the ports on which the Web server accepts incoming requests. By default, the Apache HTTP Server is set to listen to port 80 for non-secure Web communications and (in the /etc/httpd/conf.d/ssl.conf file which defines any secure servers) to port 443 for secure Web communications.

If the Apache HTTP Server is configured to listen to a port under 1024, only the root user can start it. For port 1024 and above, httpd can be started as a regular user.

The Listen directive can also be used to specify particular IP addresses over which the server accepts connections.

##### LoadModule

LoadModule is used to load Dynamic Shared Object (DSO) modules. More information on the Apache HTTP Server's DSO support, including instructions for using the LoadModule directive, can be found in [Section 21.6, “Adding Modules”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-addmods.html). Note, the load order of the modules is no longer important with Apache HTTP Server 2.0. Refer to [Section 21.2.2.1.3, “Dynamic Shared Object (DSO) Support”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-httpd-mig.html#s3-httpd-v2-mig-dso) for more information about Apache HTTP Server 2.0 DSO support.

##### Location

The <Location> and </Location> tags create a container in which access control based on URL can be specified.

For instance, to allow people connecting from within the server's domain to see status reports, use the following directives:

<Location /server-status> SetHandler server-status Order deny,allow Deny from all Allow from *<.example.com>* </Location>

Replace *<.example.com>* with the second-level domain name for the Web server.

To provide server configuration reports (including installed modules and configuration directives) to requests from inside the domain, use the following directives:

<Location /server-info> SetHandler server-info Order deny,allow Deny from all Allow from *<.example.com>* </Location>

Again, replace *<.example.com>* with the second-level domain name for the Web server.

##### LogFormat

The LogFormat directive configures the format of the various Web server log files. The actual LogFormat used depends on the settings given in the CustomLog directive (refer to [CustomLog](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-config.html#s2-apache-customlog)).

The following are the format options if the CustomLog directive is set to combined:

%h (remote host's IP address or hostname)

Lists the remote IP address of the requesting client. If HostnameLookups is set to on, the client hostname is recorded unless it is not available from DNS.

%l (rfc931)

Not used. A hyphen **-** appears in the log file for this field.

%u (authenticated user)

Lists the username of the user recorded if authentication was required. Usually, this is not used, so a hyphen **-** appears in the log file for this field.

%t (date)

Lists the date and time of the request.

%r (request string)

Lists the request string exactly as it came from the browser or client.

%s (status)

Lists the HTTP status code which was returned to the client host.

%b (bytes)

Lists the size of the document.

%\"%{Referer}i\" (referrer)

Lists the URL of the webpage which referred the client host to Web server.

%\"%{User-Agent}i\" (user-agent)

Lists the type of Web browser making the request.

##### LogLevel

LogLevel sets how verbose the error messages in the error logs are. LogLevel can be set (from least verbose to most verbose) to emerg, alert, crit, error, warn, notice, info, or debug. The default LogLevel is warn.

##### MaxKeepAliveRequests

This directive sets the maximum number of requests allowed per persistent connection. The Apache Project recommends a high setting, which improves the server's performance. MaxKeepAliveRequests is set to 100 by default, which should be appropriate for most situations.

##### NameVirtualHost

The NameVirtualHost directive associates an IP address and port number, if necessary, for any name-based virtual hosts. Name-based virtual hosting allows one Apache HTTP Server to serve different domains without using multiple IP addresses.

## Note

Name-based virtual hosts only work with non-secure HTTP connections. If using virtual hosts with a secure server, use IP address-based virtual hosts instead.

To enable name-based virtual hosting, uncomment the NameVirtualHost configuration directive and add the correct IP address. Then add additional VirtualHost containers for each virtual host as is necessary for your configuration.

##### Options

The Options directive controls which server features are available in a particular directory. For example, under the restrictive parameters specified for the root directory, Options is only set to the FollowSymLinks directive. No features are enabled, except that the server is allowed to follow symbolic links in the root directory.

By default, in the DocumentRoot directory, Options is set to include Indexes and FollowSymLinks. Indexes permits the server to generate a directory listing for a directory if no DirectoryIndex (for example, index.html) is specified. FollowSymLinks allows the server to follow symbolic links in that directory.

## Note

Options statements from the main server configuration section need to be replicated to each VirtualHost container individually. Refer to [VirtualHost](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-config.html#s2-apache-virtualhost) for more information.

##### Order

The Order directive controls the order in which allow and deny directives are evaluated. The server is configured to evaluate the Allow directives before the Deny directives for the DocumentRoot directory.

##### PidFile

PidFile names the file where the server records its process ID (PID). By default the PID is listed in /var/run/httpd.pid.

##### Proxy

<Proxy \*> and </Proxy> tags create a container which encloses a group of configuration directives meant to apply only to the proxy server. Many directives which are allowed within a <Directory> container may also be used within <Proxy> container.

##### ProxyRequests

To configure the Apache HTTP Server to function as a proxy server, remove the hash mark (#) from the beginning of the <IfModule mod\_proxy.c> line, the ProxyRequests, and each line in the <Proxy> stanza. Set the ProxyRequests directive to On, and set which domains are allowed access to the server in the Allow from directive of the <Proxy> stanza.

##### ReadmeName

ReadmeName names the file which, if it exists in the directory, is appended to the end of server generated directory listings. The Web server first tries to include the file as an HTML document and then tries to include it as plain text. By default, ReadmeName is set to README.html.

##### Redirect

When a webpage is moved, Redirect can be used to map the file location to a new URL. The format is as follows:

Redirect /*<old-path>*/*<file-name>* http://*<current-domain>*/*<current-path>*/*<file-name>*

In this example, replace *<old-path>* with the old path information for *<file-name>* and *<current-domain>* and *<current-path>* with the current domain and path information for *<file-name>*.

In this example, any requests for *<file-name>* at the old location is automatically redirected to the new location.

For more advanced redirection techniques, use the mod\_rewrite module included with the Apache HTTP Server. For more information about configuring the mod\_rewrite module, refer to the Apache Software Foundation documentation online at <http://httpd.apache.org/docs/2.2/mod/mod_rewrite.html>.

##### ScriptAlias

The ScriptAlias directive defines where CGI scripts are located. Generally, it is not good practice to leave CGI scripts within the DocumentRoot, where they can potentially be viewed as text documents. For this reason, a special directory outside of the DocumentRoot directory containing server-side executables and scripts is designated by the ScriptAlias directive. This directory is known as a cgi-bin and is set to /var/www/cgi-bin/ by default.

It is possible to establish directories for storing executables outside of the cgi-bin/ directory. For instructions on doing so, refer to [AddHandler](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-config.html#s2-apache-addhandler) and [Directory](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-config.html#s2-apache-directory).

##### ServerAdmin

Sets the ServerAdmin directive to the email address of the Web server administrator. This email address shows up in error messages on server-generated Web pages, so users can report a problem by sending email to the server administrator.

By default, ServerAdmin is set to root@localhost.

A common way to set up ServerAdmin is to set it to webmaster@example.com. Once set, alias webmaster to the person responsible for the Web server in /etc/aliases and run /usr/bin/newaliases.

##### ServerName

ServerName specifies a hostname and port number (matching the Listen directive) for the server. The ServerName does not need to match the machine's actual hostname. For example, the Web server may be www.example.com, but the server's hostname is actually foo.example.com. The value specified in ServerName must be a valid Domain Name Service (DNS) name that can be resolved by the system — do not make something up.

The following is a sample ServerName directive:

ServerName www.example.com:80

When specifying a ServerName, be sure the IP address and server name pair are included in the /etc/hosts file.

##### ServerRoot

The ServerRoot directive specifies the top-level directory containing website content. By default, ServerRoot is set to "/etc/httpd" for both secure and non-secure servers.

##### ServerSignature

The ServerSignature directive adds a line containing the Apache HTTP Server server version and the ServerName to any server-generated documents, such as error messages sent back to clients. ServerSignature is set to on by default.

ServerSignature can be set to EMail which adds a mailto:ServerAdmin HTML tag to the signature line of auto-generated responses. ServerSignature can also be set to Off to stop Apache from sending out its version number and module information. Please also check the ServerTokens settings.

##### ServerTokens

The ServerTokens directive determines if the Server response header field sent back to clients should include details of the Operating System type and information about compiled-in modules. By default, ServerTokens is set to Full which sends information about the Operating System type and compiled-in modules. Setting the ServerTokens to Prod sends the product name only and is recommended as many hackers check information in the Server header when scanning for vulnerabilities. You can also set the ServerTokens to Min (minimal) or to OS (operating system).

##### SuexecUserGroup

The SuexecUserGroup directive, which originates from the mod\_suexec module, allows the specification of user and group execution privileges for CGI programs. Non-CGI requests are still processed with the user and group specified in the User and Group directives.

## Note

From version 2.0, the SuexecUserGroup directive replaced the Apache HTTP Server 1.3 configuration of using the User and Group directives inside the configuration of VirtualHosts sections.

##### Timeout

Timeout defines, in seconds, the amount of time that the server waits for receipts and transmissions during communications. Timeout is set to 300 seconds by default, which is appropriate for most situations.

##### TypesConfig

TypesConfig names the file which sets the default list of MIME type mappings (file name extensions to content types). The default TypesConfig file is /etc/mime.types. Instead of editing /etc/mime.types, the recommended way to add MIME type mappings is to use the AddType directive.

For more information about AddType, refer to [AddType](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-config.html#s2-apache-addtype).

##### UseCanonicalName

When set to on, this directive configures the Apache HTTP Server to reference itself using the value specified in the ServerName and Port directives. When UseCanonicalName is set to off, the server instead uses the value used by the requesting client when referring to itself.

UseCanonicalName is set to off by default.

##### User

The User directive sets the username of the server process and determines what files the server is allowed to access. Any files inaccessible to this user are also inaccessible to clients connecting to the Apache HTTP Server.

By default User is set to apache.

This directive has been deprecated for the configuration of virtual hosts.

## Note

For security reasons, the Apache HTTP Server does not run as the root user.

##### UserDir

UserDir is the subdirectory within each user's home directory where they should place personal HTML files which are served by the Web server. This directive is set to disable by default.

The name for the subdirectory is set to public\_html in the default configuration. For example, the server might receive the following request:

http://*example.com*/~*username*/foo.html

The server would look for the file:

/home/username/public\_html/foo.html

In the above example, /home/username/ is the user's home directory (note that the default path to users' home directories may vary).

Make sure that the permissions on the users' home directories are set correctly. Users' home directories must be set to 0711. The read (r) and execute (x) bits must be set on the users' public\_html directories (0755 also works). Files that are served in a users' public\_html directories must be set to at least 0644.

##### VirtualHost

<VirtualHost> and </VirtualHost> tags create a container outlining the characteristics of a virtual host. The VirtualHost container accepts most configuration directives.

A commented VirtualHost container is provided in httpd.conf, which illustrates the minimum set of configuration directives necessary for each virtual host. Refer to [Section 21.7, “Virtual Hosts”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-virtualhosts.html) for more information about virtual hosts.

## Note

The default SSL virtual host container now resides in the file /etc/httpd/conf.d/ssl.conf.

### 21.5.2. Configuration Directives for SSL

The directives in /etc/httpd/conf.d/ssl.conf file can be configured to enable secure Web communications using SSL and TLS.

##### SetEnvIf

SetEnvIf sets environment variables based on the headers of incoming connections. It is not solely an SSL directive, though it is present in the supplied /etc/httpd/conf.d/ssl.conf file. It's purpose in this context is to disable HTTP keepalive and to allow SSL to close the connection without a closing notification from the client browser. This setting is necessary for certain browsers that do not reliably shut down the SSL connection.

For more information on other directives within the SSL configuration file, refer to the following URLs:

* http://localhost/manual/mod/mod\_ssl.html
* <http://httpd.apache.org/docs/2.2/mod/mod_ssl.html>

## Note

In most cases, SSL directives are configured appropriately during the installation of Red Hat Enterprise Linux. Be careful when altering Apache HTTP Secure Server directives, misconfiguration can lead to security vulnerabilities.

### 21.5.3. MPM Specific Server-Pool Directives

As explained in [Section 21.2.2.1.2, “Server-Pool Size Regulation”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-httpd-mig.html#s3-httpd-v2-mig-pool), the responsibility for managing characteristics of the server-pool falls to a module group called MPMs under Apache HTTP Server 2.0. The characteristics of the server-pool differ depending upon which MPM is used. For this reason, an IfModule container is necessary to define the server-pool for the MPM in use.

By default, Apache HTTP Server 2.0 defines the server-pool for both the prefork and worker MPMs.

The following section list directives found within the MPM-specific server-pool containers.

##### MaxClients

MaxClients sets a limit on the total number of server processes, or simultaneously connected clients, that can run at one time. The main purpose of this directive is to keep a runaway Apache HTTP Server from crashing the operating system. For busy servers this value should be set to a high value. The server's default is set to 150 regardless of the MPM in use. However, it is not recommended that the value for MaxClients exceeds 256 when using the prefork MPM.

##### MaxRequestsPerChild

MaxRequestsPerChild sets the total number of requests each child server process serves before the child dies. The main reason for setting MaxRequestsPerChild is to avoid long-lived process induced memory leaks. The default MaxRequestsPerChild for the prefork MPM is 4000 and for the worker MPM is 0.

##### MinSpareServers and MaxSpareServers

These values are only used with the prefork MPM. They adjust how the Apache HTTP Server dynamically adapts to the perceived load by maintaining an appropriate number of spare server processes based on the number of incoming requests. The server checks the number of servers waiting for a request and kills some if there are more than MaxSpareServers or creates some if the number of servers is less than MinSpareServers.

The default MinSpareServers value is 5; the default MaxSpareServers value is 20. These default settings should be appropriate for most situations. Be careful not to increase the MinSpareServers to a large number as doing so creates a heavy processing load on the server even when traffic is light.

##### MinSpareThreads and MaxSpareThreads

These values are only used with the worker MPM. They adjust how the Apache HTTP Server dynamically adapts to the perceived load by maintaining an appropriate number of spare server threads based on the number of incoming requests. The server checks the number of server threads waiting for a request and kills some if there are more than MaxSpareThreads or creates some if the number of servers is less than MinSpareThreads.

The default MinSpareThreads value is 25; the default MaxSpareThreads value is 75. These default settings should be appropriate for most situations. The value for MaxSpareThreads must be greater than or equal to the sum of MinSpareThreads and ThreadsPerChild, else the Apache HTTP Server automatically corrects it.

##### StartServers

The StartServers directive sets how many server processes are created upon startup. Since the Web server dynamically kills and creates server processes based on traffic load, it is not necessary to change this parameter. The Web server is set to start 8 server processes at startup for the prefork MPM and 2 for the worker MPM.

##### ThreadsPerChild

This value is only used with the worker MPM. It sets the number of threads within each child process. The default value for this directive is 25.

## 21.6. Adding Modules

The Apache HTTP Server is distributed with a number of modules. More information about Apache HTTP modules can be found on <http://httpd.apache.org/docs/2.2/mod/>.

The Apache HTTP Server supports *Dynamically Shared Objects* (*DSO*s), or modules, which can easily be loaded at runtime as necessary.

The Apache Project provides complete DSO documentation online at <http://httpd.apache.org/docs/2.2/dso.html>. Or, if the http-manual package is installed, documentation about DSOs can be found online at <http://localhost/manual/mod/>.

For the Apache HTTP Server to use a DSO, it must be specified in a LoadModule directive within /etc/httpd/conf/httpd.conf. If the module is provided by a separate package, the line must appear within the modules configuration file in the /etc/httpd/conf.d/ directory. Refer to [LoadModule](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-config.html#s2-apache-loadmodule) for more information.

If adding or deleting modules from http.conf, Apache HTTP Server must be reloaded or restarted, as referred to in [Section 21.3, “Starting and Stopping httpd”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-startstop.html).

If creating a new module, first install the httpd-devel package which contains the include files, the header files, as well as the *APache eXtenSion* (/usr/sbin/apxs) application, which uses the include files and the header files to compile DSOs.

After writing a module, use /usr/sbin/apxs to compile the module sources outside the Apache source tree. For more information about using the /usr/sbin/apxs command, refer to the the Apache documentation online at <http://httpd.apache.org/docs/2.2/dso.html> as well as the apxs man page.

Once compiled, put the module in the /usr/lib/httpd/modules/ directory. For RHEL platforms using default-64-bit userspace (x86\_64, ia64, ?) this path will be /usr/lib64/httpd/modules/. Then add a LoadModule line to the httpd.conf, using the following structure:

LoadModule *<module-name> <path/to/module.so>*

Where *<module-name>* is the name of the module and *<path/to/module.so>* is the path to the DSO.

## 21.7. Virtual Hosts

The Apache HTTP Server's built in virtual hosting allows the server to provide different information based on which IP address, hostname, or port is being requested. A complete guide to using virtual hosts is available online at <http://httpd.apache.org/docs/2.2/vhosts/>.

### 21.7.1. Setting Up Virtual Hosts

To create a name-based virtual host, it is best to use the virtual host container provided in httpd.conf as an example.

The virtual host example read as follows:

#NameVirtualHost \*:80 # #<VirtualHost \*:80> # ServerAdmin webmaster@dummy-host.example.com # DocumentRoot /www/docs/dummy-host.example.com # ServerName dummy-host.example.com # ErrorLog logs/dummy-host.example.com-error\_log # CustomLog logs/dummy-host.example.com-access\_log common #</VirtualHost>

To activate name-based virtual hosting, uncomment the NameVirtualHost line by removing the hash mark (#) and replace the asterisk (\*) with the IP address assigned to the machine.

Next, configure a virtual host by uncommenting and customizing the <VirtualHost> container.

On the <VirtualHost> line, change the asterisk (\*) to the server's IP address. Change the ServerName to a valid DNS name assigned to the machine, and configure the other directives as necessary.

The <VirtualHost> container is highly customizable and accepts almost every directive available within the main server configuration.

## Tip

If configuring a virtual host to listen on a non-default port, that port must be added to the Listen directive in the global settings section of /etc/httpd/conf/httpd.conf file.

To activate a newly created virtual host, the Apache HTTP Server must be reloaded or restarted. Refer to [Section 21.3, “Starting and Stopping httpd”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-apache-startstop.html) for further instructions.

Comprehensive information about creating and configuring both name-based and IP address-based virtual hosts is provided online at <http://httpd.apache.org/docs/2.2/vhosts/>.

**21.8. Apache HTTP Secure Server Configuration**

This section provides basic information on the Apache HTTP Server with the mod\_ssl security module enabled to use the OpenSSL library and toolkit. The combination of these three components are referred to in this section as the secure Web server or just as the secure server.

The mod\_ssl module is a security module for the Apache HTTP Server. The mod\_ssl module uses the tools provided by the OpenSSL Project to add a very important feature to the Apache HTTP Server — the ability to encrypt communications. In contrast, regular HTTP communications between a browser and a Web server are sent in plain text, which could be intercepted and read by someone along the route between the browser and the server.

This section is not meant to be complete and exclusive documentation for any of these programs. When possible, this guide points to appropriate places where you can find more in-depth documentation on particular subjects.

This section shows you how to install these programs. You can also learn the steps necessary to generate a private key and a certificate request, how to generate your own self-signed certificate, and how to install a certificate to use with your secure server.

The mod\_ssl configuration file is located at /etc/httpd/conf.d/ssl.conf. For this file to be loaded, and hence for mod\_ssl to work, you must have the statement Include conf.d/\*.conf in the /etc/httpd/conf/httpd.conf file. This statement is included by default in the default Apache HTTP Server configuration file.

**21.8.1. An Overview of Security-Related Packages**

To enable the secure server, you must have the following packages installed at a minimum:

httpd

The httpd package contains the httpd daemon and related utilities, configuration files, icons, Apache HTTP Server modules, man pages, and other files used by the Apache HTTP Server.

mod\_ssl

The mod\_ssl package includes the mod\_ssl module, which provides strong cryptography for the Apache HTTP Server via the Secure Sockets Layer (SSL) and Transport Layer Security (TLS) protocols.

openssl

The openssl package contains the OpenSSL toolkit. The OpenSSL toolkit implements the SSL and TLS protocols, and also includes a general purpose cryptography library.

crypto-utils

The crypto-utils package provides a set of utilities to generate and manage SSL certificates and private keys. Among these utilities is genkey.

Additionally, other software packages provide certain security functionalities (but are not required by the secure server to function):

**21.8.2. An Overview of Certificates and Security**

Your secure server provides security using a combination of the Secure Sockets Layer (SSL) protocol and (in most cases) a digital certificate from a Certificate Authority (CA). SSL handles the encrypted communications as well as the mutual authentication between browsers and your secure server. The CA-approved digital certificate provides authentication for your secure server (the CA puts its reputation behind its certification of your organization's identity). When your browser is communicating using SSL encryption, the https:// prefix is used at the beginning of the Uniform Resource Locator (URL) in the navigation bar.

Encryption depends upon the use of keys (think of them as secret encoder/decoder rings in data format). In conventional or symmetric cryptography, both ends of the transaction have the same key, which they use to decode each other's transmissions. In public or asymmetric cryptography, two keys co-exist: a public key and a private key. A person or an organization keeps their private key a secret and publishes their public key. Data encoded with the public key can only be decoded with the private key; data encoded with the private key can only be decoded with the public key.

To set up your secure server, use public cryptography to create a public and private key pair. In most cases, you send your certificate request (including your public key), proof of your company's identity, and payment to a CA. The CA verifies the certificate request and your identity, and then sends back a certificate for your secure server.

A secure server uses a certificate to identify itself to Web browsers. You can generate your own certificate (called a "self-signed" certificate), or you can get a certificate from a CA. A certificate from a reputable CA guarantees that a website is associated with a particular company or organization.

Alternatively, you can create your own self-signed certificate. Note, however, that self-signed certificates should not be used in most production environments. Self-signed certificates are not automatically accepted by a user's browser — users are prompted by the browser to accept the certificate and create the secure connection. Refer to [Section 21.8.4, “Types of Certificates”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-httpd-secure-server.html#s2-secureserver-certs) for more information on the differences between self-signed and CA-signed certificates.

Once you have a self-signed certificate or a signed certificate from the CA of your choice, you must install it on your secure server.

**21.8.3. Using Pre-Existing Keys and Certificates**

If you already have an existing key and certificate (for example, if you are installing the secure server to replace another company's secure server product), you can probably use your existing key and certificate with the secure server. The following two situations provide instances where you are not able to use your existing key and certificate:

* *If you are changing your IP address or domain name* — Certificates are issued for a particular IP address and domain name pair. You must get a new certificate if you are changing your IP address or domain name.
* *If you have a certificate from VeriSign and you are changing your server software* — VeriSign is a widely used CA. If you already have a VeriSign certificate for another purpose, you may have been considering using your existing VeriSign certificate with your new secure server. However, you are not be allowed to because VeriSign issues certificates for one specific server software and IP address/domain name combination.

If you change either of those parameters (for example, if you previously used a different secure server product), the VeriSign certificate you obtained to use with the previous configuration will not work with the new configuration. You must obtain a new certificate.

If you have an existing key and certificate that you can use, you do not have to generate a new key and obtain a new certificate. However, you may need to move and rename the files which contain your key and certificate.

Move your existing key file to:

/etc/pki/tls/private/server.key

Move your existing certificate file to:

/etc/pki/tls/certs/server.crt

If you are upgrading from the Red Hat Secure Web Server, your old key (httpsd.key) and certificate (httpsd.crt) are located in /etc/httpd/conf/. Move and rename your key and certificate so that the secure server can use them. Use the following two commands to move and rename your key and certificate files:

mv /etc/httpd/conf/httpsd.key /etc/pki/tls/private/server.key mv /etc/httpd/conf/httpsd.crt /etc/pki/tls/certs/server.crt

Then, start your secure server with the command:

/sbin/service httpd start

**21.8.4. Types of Certificates**

If you installed your secure server from the RPM package provided by Red Hat, a randomly generated private key and a test certificate are generated and put into the appropriate directories. Before you begin using your secure server, however, you must generate your own key and obtain a certificate which correctly identifies your server.

You need a key and a certificate to operate your secure server — which means that you can either generate a self-signed certificate or purchase a CA-signed certificate from a CA. What are the differences between the two?

A CA-signed certificate provides two important capabilities for your server:

* Browsers (usually) automatically recognize the certificate and allow a secure connection to be made, without prompting the user.
* When a CA issues a signed certificate, they are guaranteeing the identity of the organization that is providing the webpages to the browser.

If your secure server is being accessed by the public at large, your secure server needs a certificate signed by a CA so that people who visit your website know that the website is owned by the organization who claims to own it. Before signing a certificate, a CA verifies that the organization requesting the certificate was actually who they claimed to be.

Most Web browsers that support SSL have a list of CAs whose certificates they automatically accept. If a browser encounters a certificate whose authorizing CA is not in the list, the browser asks the user to either accept or decline the connection.

You can generate a self-signed certificate for your secure server, but be aware that a self-signed certificate does not provide the same functionality as a CA-signed certificate. A self-signed certificate is not automatically recognized by most Web browsers and does not provide any guarantee concerning the identity of the organization that is providing the website. A CA-signed certificate provides both of these important capabilities for a secure server. If your secure server is to be used in a production environment, a CA-signed certificate is recommended.

The process of getting a certificate from a CA is fairly easy. A quick overview is as follows:

1. Create an encryption private and public key pair.
2. Create a certificate request based on the public key. The certificate request contains information about your server and the company hosting it.
3. Send the certificate request, along with documents proving your identity, to a CA. Red Hat does not make recommendations on which certificate authority to choose. Your decision may be based on your past experiences, on the experiences of your friends or colleagues, or purely on monetary factors.

Once you have decided upon a CA, you need to follow the instructions they provide on how to obtain a certificate from them.

1. When the CA is satisfied that you are indeed who you claim to be, they provide you with a digital certificate.
2. Install this certificate on your secure server and begin handling secure transactions.

Whether you are getting a certificate from a CA or generating your own self-signed certificate, the first step is to generate a key. Refer to [Section 21.8.5, “Generating a Key”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-httpd-secure-server.html#s2-secureserver-generatingkey) for instructions.

**21.8.5. Generating a Key**

You must be root to generate a key.

First, use the cd command to change to the /etc/httpd/conf/ directory. Remove the fake key and certificate that were generated during the installation with the following commands:

rm ssl.key/server.key rm ssl.crt/server.crt

The crypto-utils package contains the genkey utility which you can use to generate keys as the name implies. To create your own private key, please ensure the crypto-utils package is installed. You can view more options by typing man genkey in your terminal. Assuming you wish to generate keys for www.example.com using the genkey utility, type in the following command in your terminal:

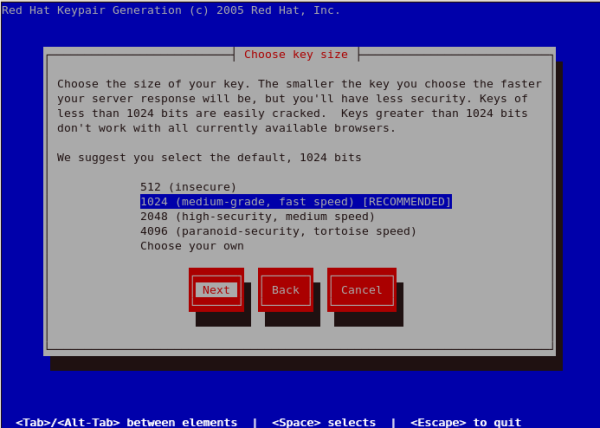
genkey www.example.com

Please note that the make based process is no longer shipped with RHEL 5. This will start the genkey graphical user interface. The figure below illustrates the first screen. To navigate, use the keyboard arrow and tab keys. This windows indicates where your key will be stored and prompts you to proceed or cancel the operation. To proceed to the next step, select **Next** and press the Return (Enter) key.



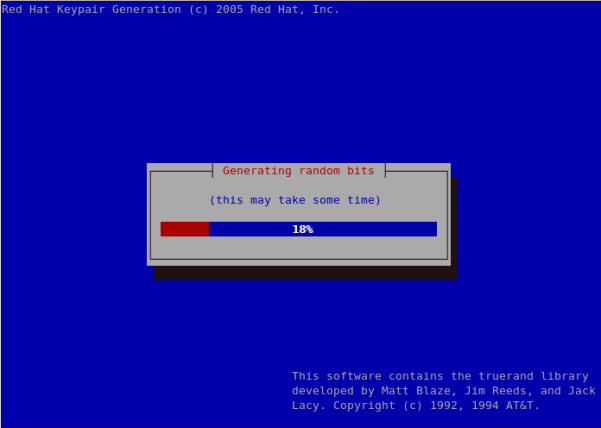
**Figure 21.11. Keypair generation**

The next screen prompts you to choose the size of your key. As indicated, the smaller the size of your key, the faster will the response from your server be and the lesser your level of security. On selecting your preferred, key size using the arrow keys, select **Next** to proceed to the next step. The figure below illustrates the key size selection screen.



**Figure 21.12. Choose key size**

Selecting the next step will initiate the random bits generation process which may take some time depending on the size of your selected key. The larger the size of your key, the longer it will take to generate it.



**Figure 21.13. Generating random bits**

On generating your key, you will be prompted to send a Certificate Request (CSR) to a Certificate Authority (CA).



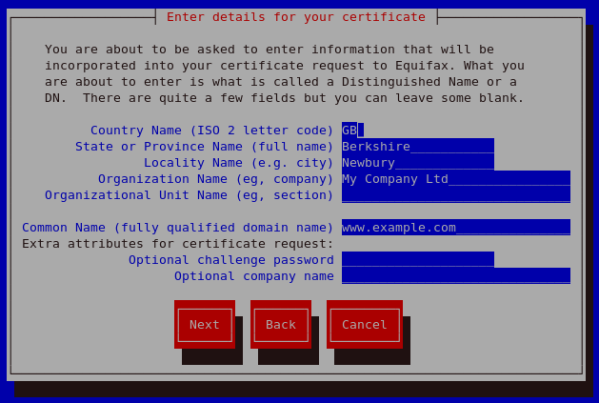
**Figure 21.14. Generate CSR**

Selecting **Yes** will prompt you to select the Certificate Authority you wish to send your request to. Selecting **No** will allow you to generate a self-signed certificate. The next step for this is illustrated in [Figure 21.17, “Generating a self signed certificate for your server”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-httpd-secure-server.html#private-signed-cert).



**Figure 21.15. Choose Certificate Authority (CA)**

On Selecting your preferred option, select **Next** to proceed to the next step. The next screen allows you to enter the details of your certificate.



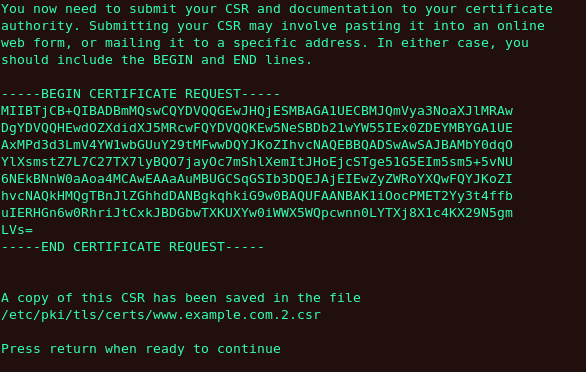
**Figure 21.16. Enter details for your certificate**

If you prefer to generate a self signed cert key pair, you should not generate a CSR. To do this, select **No** as your preferred option in the Generate CSR screen. This will display the figure below from which you can enter your certificate details. Entering your certificate details and pressing the return key will display the [Figure 21.19, “Protecting your private key”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-httpd-secure-server.html#protect-private-key) from which you can choose to encrypt your private key or not.



**Figure 21.17. Generating a self signed certificate for your server**

On entering the details of your certificate, select **Next** to proceed. The figure below illustrates an example of a the next screen displayed after completing the details for a certificate to be sent to Equifax. Please note that if you are generating a self signed key, for your server, this screen is not displayed.



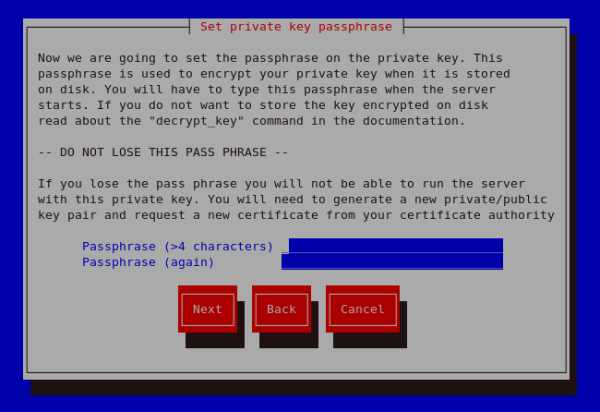
**Figure 21.18. Begin certificate request**

Pressing the return key, will display the next screen from which you can enable or disable the encryption of the private key. Use the spacebar to enable or disable this. When enabled, a [\*] character will be displayed. On selecting your preferred option, select **Next** to proceed to the next step.



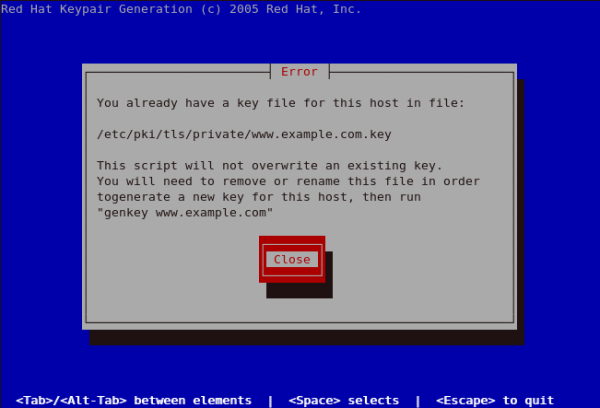
**Figure 21.19. Protecting your private key**

The next screen allows you to set your key passphase. Please do not lose this pass phase as you will not be able to run the server without it. You will need to regenerate a new private or public key pair and request a new certificate from your CA as indicated. For security, the passphase is not displayed as you type. On typing your preferred passphase, select **Next** to go back to your terminal.



**Figure 21.20. Set passphase**

If you attempt to run genkey makeca on a server that has an existing key pair, an error message will be displayed as illustrated below. You need to delete your existing key file as indicated to generate a new key pair.



**Figure 21.21. genkey error**

* <http://httpd.apache.org/docs/2.2/ssl/>
* <http://httpd.apache.org/docs/2.2/vhosts/>

**21.8.6. How to configure the server to use the new key**

The steps to configure the Apache HTTP Server to use the new key are:

* Obtain the signed certificate from the CA after submitting the CSR.
* Copy the certificate to the path, for example /etc/pki/tls/certs/www.example.com.crt
* Edit /etc/httpd/conf.d/ssl.conf. Change the SSLCertificateFile and SSLCertificateKey lines to be.
* SSLCertificateFile /etc/pki/tls/certs/www.example.com.crt
* SSLCertificateKeyFile /etc/pki/tls/private/www.example.com.key

where the "www.example.com" part should match the argument passed on the genkey command.

**Chapter 22. FTP**

[22.1. The File Transport Protocol](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-ftp.html#s1-ftp-protocol)

[22.2. FTP Servers](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ftp-servers.html)

[22.3. Files Installed with vsftpd](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s2-ftp-vsftpd-conf.html)

[22.4. Starting and Stopping vsftpd](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ftp-vsftpd-start.html)

[22.5. vsftpd Configuration Options](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ftp-vsftpd-conf.html)

[22.6. Additional Resources](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ftp-resources.html)

File Transfer Protocol (FTP) is one of the oldest and most commonly used protocols found on the Internet today. Its purpose is to reliably transfer files between computer hosts on a network without requiring the user to log directly into the remote host or have knowledge of how to use the remote system. It allows users to access files on remote systems using a standard set of simple commands.

This chapter outlines the basics of the FTP protocol, as well as configuration options for the primary FTP server shipped with Red Hat Enterprise Linux, vsftpd.

**22.1. The File Transport Protocol**

However, because FTP is so prevalent on the Internet, it is often required to share files to the public. System administrators, therefore, should be aware of the FTP protocol's unique characteristics.

**22.1.1. Multiple Ports, Multiple Modes**

Unlike most protocols used on the Internet, FTP requires multiple network ports to work properly. When an FTP client application initiates a connection to an FTP server, it opens port 21 on the server — known as the *command port*. This port is used to issue all commands to the server. Any data requested from the server is returned to the client via a *data port*. The port number for data connections, and the way in which data connections are initialized, vary depending upon whether the client requests the data in *active* or *passive* mode.

The following defines these modes:

active mode

Active mode is the original method used by the FTP protocol for transferring data to the client application. When an active mode data transfer is initiated by the FTP client, the server opens a connection from port 20 on the server to the IP address and a random, unprivileged port (greater than 1024) specified by the client. This arrangement means that the client machine must be allowed to accept connections over any port above 1024. With the growth of insecure networks, such as the Internet, the use of firewalls to protect client machines is now prevalent. Because these client-side firewalls often deny incoming connections from active mode FTP servers, passive mode was devised.

passive mode

Passive mode, like active mode, is initiated by the FTP client application. When requesting data from the server, the FTP client indicates it wants to access the data in passive mode and the server provides the IP address and a random, unprivileged port (greater than 1024) on the server. The client then connects to that port on the server to download the requested information.

While passive mode resolves issues for client-side firewall interference with data connections, it can complicate administration of the server-side firewall. You can reduce the number of open ports on a server by limiting the range of unprivileged ports on the FTP server. This also simplifies the process of configuring firewall rules for the server. Refer to [Section 22.5.8, “Network Options”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ftp-vsftpd-conf.html#s2-ftp-vsftpd-conf-opt-net) for more about limiting passive ports.

## 22.2. FTP Servers

Red Hat Enterprise Linux ships with two different FTP servers:

* **Red Hat Content Accelerator** — A kernel-based Web server that delivers high performance Web server and FTP services. Since speed as its primary design goal, it has limited functionality and runs only as an anonymous FTP server. For more information about configuring and administering **Red Hat Content Accelerator**, consult the documentation available online at <http://www.redhat.com/docs/manuals/tux/>.
* vsftpd — A fast, secure FTP daemon which is the preferred FTP server for Red Hat Enterprise Linux. The remainder of this chapter focuses on vsftpd.

### 22.2.1. vsftpd

The Very Secure FTP Daemon (vsftpd) is designed from the ground up to be fast, stable, and, most importantly, secure. Its ability to handle large numbers of connections efficiently and securely is why vsftpd is the only stand-alone FTP distributed with Red Hat Enterprise Linux.

The security model used by vsftpd has three primary aspects:

* Strong separation of privileged and non-privileged processes — Separate processes handle different tasks, and each of these processes run with the minimal privileges required for the task.
* Tasks requiring elevated privileges are handled by processes with the minimal privilege necessary — By leveraging compatibilities found in the libcap library, tasks that usually require full root privileges can be executed more safely from a less privileged process.
* Most processes run in a *chroot* jail — Whenever possible, processes are change-rooted to the directory being shared; this directory is then considered a chroot jail. For example, if the directory /var/ftp/ is the primary shared directory, vsftpd reassigns /var/ftp/ to the new root directory, known as /. This disallows any potential malicious hacker activities for any directories not contained below the new root directory.

Use of these security practices has the following effect on how vsftpd deals with requests:

* The parent process runs with the least privileges required — The parent process dynamically calculates the level of privileges it requires to minimize the level of risk. Child processes handle direct interaction with the FTP clients and run with as close to no privileges as possible.
* All operations requiring elevated privileges are handled by a small parent process — Much like the Apache HTTP Server, vsftpd launches unprivileged child processes to handle incoming connections. This allows the privileged, parent process to be as small as possible and handle relatively few tasks.
* All requests from unprivileged child processes are distrusted by the parent process — Communication with child processes are received over a socket, and the validity of any information from child processes is checked before being acted on.
* Most interaction with FTP clients is handled by unprivileged child processes in a *chroot* jail — Because these child processes are unprivileged and only have access to the directory being shared, any crashed processes only allows the attacker access to the shared files.

## 22.3. Files Installed with vsftpd

The vsftpd RPM installs the daemon (/usr/sbin/vsftpd), its configuration and related files, as well as FTP directories onto the system. The following lists the files and directories related to vsftpd configuration:

* /etc/rc.d/init.d/vsftpd — The initialization script (*initscript*) used by the /sbin/service command to start, stop, or reload vsftpd. Refer to [Section 22.4, “Starting and Stopping vsftpd”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ftp-vsftpd-start.html) for more information about using this script.
* /etc/pam.d/vsftpd — The Pluggable Authentication Modules (PAM) configuration file for vsftpd. This file specifies the requirements a user must meet to login to the FTP server. For more information, refer to [Section 42.4, “Pluggable Authentication Modules (PAM)”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-pam.html).
* /etc/vsftpd/vsftpd.conf — The configuration file for vsftpd. Refer to [Section 22.5, “vsftpd Configuration Options”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ftp-vsftpd-conf.html) for a list of important options contained within this file.
* /etc/vsftpd.ftpusers — A list of users not allowed to log into vsftpd. By default, this list includes the root, bin, and daemon users, among others.
* /etc/vsftpd.user\_list — This file can be configured to either deny or allow access to the users listed, depending on whether the userlist\_deny directive is set to YES (default) or NO in /etc/vsftpd/vsftpd.conf. If /etc/vsftpd.user\_list is used to grant access to users, the usernames listed must not appear in /etc/vsftpd.ftpusers.
* /var/ftp/ — The directory containing files served by vsftpd. It also contains the /var/ftp/pub/ directory for anonymous users. Both directories are world-readable, but writable only by the root user.

## 22.4. Starting and Stopping vsftpd

The vsftpd RPM installs the /etc/rc.d/init.d/vsftpd script, which can be accessed using the /sbin/service command.

To start the server, as root type:

/sbin/service vsftpd start

To stop the server, as root type:

/sbin/service vsftpd stop

The restart option is a shorthand way of stopping and then starting vsftpd. This is the most efficient way to make configuration changes take effect after editing the configuration file for vsftpd.

To restart the server, as root type:

/sbin/service vsftpd restart

The condrestart (*conditional restart*) option only starts vsftpd if it is currently running. This option is useful for scripts, because it does not start the daemon if it is not running.

To conditionally restart the server, as root type:

/sbin/service vsftpd condrestart

By default, the vsftpd service does not start automatically at boot time. To configure the vsftpd service to start at boot time, use an initscript utility, such as /sbin/chkconfig, **/usr/sbin/ntsysv**, or the **Services Configuration Tool** program. Refer to [Chapter 15, *Controlling Access to Services*](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-services.html) for more information regarding these tools.

### 22.4.1. Starting Multiple Copies of vsftpd

Sometimes one computer is used to serve multiple FTP domains. This is a technique called *multihoming*. One way to multihome using vsftpd is by running multiple copies of the daemon, each with its own configuration file.

To do this, first assign all relevant IP addresses to network devices or alias network devices on the system. Refer to [Chapter 14, *Network Configuration*](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-network-config.html) for more information about configuring network devices and device aliases. Additional information can be found about network configuration scripts in [Chapter 13, *Network Interfaces*](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-networkscripts.html).

Next, the DNS server for the FTP domains must be configured to reference the correct machine. For information about BIND and its configuration files, refer to [Chapter 16, *Berkeley Internet Name Domain (BIND)*](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-bind.html).

For vsftpd to answer requests on different IP addresses, multiple copies of the daemon must be running. The first copy must be run using the vsftpd initscripts, as outlined in [Section 22.4, “Starting and Stopping vsftpd”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ftp-vsftpd-start.html). This copy uses the standard configuration file, /etc/vsftpd/vsftpd.conf.

Each additional FTP site must have a configuration file with a unique name in the /etc/vsftpd/ directory, such as /etc/vsftpd/vsftpd-site-2.conf. Each configuration file must be readable and writable only by root. Within each configuration file for each FTP server listening on an IPv4 network, the following directive must be unique:

listen\_address=*N.N.N.N*

Replace *N.N.N.N* with the unique IP address for the FTP site being served. If the site is using IPv6, use the listen\_address6 directive instead.

Once each additional server has a configuration file, the vsftpd daemon must be launched from a root shell prompt using the following command:

vsftpd /etc/vsftpd/*<configuration-file>* [amp ]

In the above command, replace *<configuration-file>* with the unique name for the server's configuration file, such as /etc/vsftpd/vsftpd-site-2.conf.

Other directives to consider altering on a per-server basis are:

* anon\_root
* local\_root
* vsftpd\_log\_file
* xferlog\_file

For a detailed list of directives available within vsftpd's configuration file, refer to [Section 22.5, “vsftpd Configuration Options”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ftp-vsftpd-conf.html).

To configure any additional servers to start automatically at boot time, add the above command to the end of the /etc/rc.local file.

## 22.5. vsftpd Configuration Options

Although vsftpd may not offer the level of customization other widely available FTP servers have, it offers enough options to fill most administrator's needs. The fact that it is not overly feature-laden limits configuration and programmatic errors.

All configuration of vsftpd is handled by its configuration file, /etc/vsftpd/vsftpd.conf. Each directive is on its own line within the file and follows the following format:

*<directive>*=*<value>*

For each directive, replace *<directive>* with a valid directive and *<value>* with a valid value.

## Important

There must not be any spaces between the *<directive>*, equal symbol, and the *<value>* in a directive.

Comment lines must be preceded by a hash mark (#) and are ignored by the daemon.

For a complete list of all directives available, refer to the man page for vsftpd.conf.

## Important

For an overview of ways to secure vsftpd, refer to [Section 42.2, “Server Security”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-server.html).

The following is a list of some of the more important directives within /etc/vsftpd/vsftpd.conf. All directives not explicitly found within vsftpd's configuration file are set to their default value.

### 22.5.1. Daemon Options

The following is a list of directives which control the overall behavior of the vsftpd daemon.

* listen — When enabled, vsftpd runs in stand-alone mode. Red Hat Enterprise Linux sets this value to YES. This directive cannot be used in conjunction with the listen\_ipv6 directive.

The default value is NO.

* listen\_ipv6 — When enabled, vsftpd runs in stand-alone mode, but listens only to IPv6 sockets. This directive cannot be used in conjunction with the listen directive.

The default value is NO.

* session\_support — When enabled, vsftpd attempts to maintain login sessions for each user through Pluggable Authentication Modules (PAM). Refer to [Section 42.4, “Pluggable Authentication Modules (PAM)”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-pam.html) for more information. If session logging is not necessary, disabling this option allows vsftpd to run with less processes and lower privileges.

The default value is YES.

### 22.5.2. Log In Options and Access Controls

The following is a list of directives which control the login behavior and access control mechanisms.

* anonymous\_enable — When enabled, anonymous users are allowed to log in. The usernames anonymous and ftp are accepted.

The default value is YES.

Refer to [Section 22.5.3, “Anonymous User Options”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ftp-vsftpd-conf.html#s2-ftp-vsftpd-conf-opt-anon) for a list of directives affecting anonymous users.

* banned\_email\_file — If the deny\_email\_enable directive is set to YES, this directive specifies the file containing a list of anonymous email passwords which are not permitted access to the server.

The default value is /etc/vsftpd.banned\_emails.

* banner\_file — Specifies the file containing text displayed when a connection is established to the server. This option overrides any text specified in the ftpd\_banner directive.

There is no default value for this directive.

* cmds\_allowed — Specifies a comma-delimited list of FTP commands allowed by the server. All other commands are rejected.

There is no default value for this directive.

* deny\_email\_enable — When enabled, any anonymous user utilizing email passwords specified in the /etc/vsftpd.banned\_emails are denied access to the server. The name of the file referenced by this directive can be specified using the banned\_email\_file directive.

The default value is NO.

* ftpd\_banner — When enabled, the string specified within this directive is displayed when a connection is established to the server. This option can be overridden by the banner\_file directive.

By default vsftpd displays its standard banner.

* local\_enable — When enabled, local users are allowed to log into the system.

The default value is YES.

Refer to [Section 22.5.4, “Local User Options”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ftp-vsftpd-conf.html#s2-ftp-vsftpd-conf-opt-usr) for a list of directives affecting local users.

* pam\_service\_name — Specifies the PAM service name for vsftpd.

The default value is ftp. Note, in Red Hat Enterprise Linux, the value is set to vsftpd.

* The default value is NO. Note, in Red Hat Enterprise Linux, the value is set to YES.
* userlist\_deny — When used in conjunction with the userlist\_enable directive and set to NO, all local users are denied access unless the username is listed in the file specified by the userlist\_file directive. Because access is denied before the client is asked for a password, setting this directive to NO prevents local users from submitting unencrypted passwords over the network.

The default value is YES.

* userlist\_enable — When enabled, the users listed in the file specified by the userlist\_file directive are denied access. Because access is denied before the client is asked for a password, users are prevented from submitting unencrypted passwords over the network.

The default value is NO, however under Red Hat Enterprise Linux the value is set to YES.

* userlist\_file — Specifies the file referenced by vsftpd when the userlist\_enable directive is enabled.

The default value is /etc/vsftpd.user\_list and is created during installation.

* cmds\_allowed — Specifies a comma separated list of FTP commands that the server allows. Any other commands are rejected.

There is no default value for this directive.

### 22.5.3. Anonymous User Options

The following lists directives which control anonymous user access to the server. To use these options, the anonymous\_enable directive must be set to YES.

* anon\_mkdir\_write\_enable — When enabled in conjunction with the write\_enable directive, anonymous users are allowed to create new directories within a parent directory which has write permissions.

The default value is NO.

* anon\_root — Specifies the directory vsftpd changes to after an anonymous user logs in.

There is no default value for this directive.

* anon\_upload\_enable — When enabled in conjunction with the write\_enable directive, anonymous users are allowed to upload files within a parent directory which has write permissions.

The default value is NO.

* anon\_world\_readable\_only — When enabled, anonymous users are only allowed to download world-readable files.

The default value is YES.

* ftp\_username — Specifies the local user account (listed in /etc/passwd) used for the anonymous FTP user. The home directory specified in /etc/passwd for the user is the root directory of the anonymous FTP user.

The default value is ftp.

* no\_anon\_password — When enabled, the anonymous user is not asked for a password.

The default value is NO.

* secure\_email\_list\_enable — When enabled, only a specified list of email passwords for anonymous logins are accepted. This is a convenient way to offer limited security to public content without the need for virtual users.

Anonymous logins are prevented unless the password provided is listed in /etc/vsftpd.email\_passwords. The file format is one password per line, with no trailing white spaces.

The default value is NO.

### 22.5.4. Local User Options

The following lists directives which characterize the way local users access the server. To use these options, the local\_enable directive must be set to YES.

* chmod\_enable — When enabled, the FTP command SITE CHMOD is allowed for local users. This command allows the users to change the permissions on files.

The default value is YES.

* chroot\_list\_enable — When enabled, the local users listed in the file specified in the chroot\_list\_file directive are placed in a chroot jail upon log in.

If enabled in conjunction with the chroot\_local\_user directive, the local users listed in the file specified in the chroot\_list\_file directive are not placed in a chroot jail upon log in.

The default value is NO.

* chroot\_list\_file — Specifies the file containing a list of local users referenced when the chroot\_list\_enable directive is set to YES.

The default value is /etc/vsftpd.chroot\_list.

* chroot\_local\_user — When enabled, local users are change-rooted to their home directories after logging in.

The default value is NO.

## Warning

Enabling chroot\_local\_user opens up a number of security issues, especially for users with upload privileges. For this reason, it is not recommended.

* guest\_enable — When enabled, all non-anonymous users are logged in as the user guest, which is the local user specified in the guest\_username directive.

The default value is NO.

* guest\_username — Specifies the username the guest user is mapped to.

The default value is ftp.

* local\_root — Specifies the directory vsftpd changes to after a local user logs in.

There is no default value for this directive.

* local\_umask — Specifies the umask value for file creation. Note that the default value is in octal form (a numerical system with a base of eight), which includes a "0" prefix. Otherwise the value is treated as a base-10 integer.

The default value is 022.

* passwd\_chroot\_enable — When enabled in conjunction with the chroot\_local\_user directive, vsftpd change-roots local users based on the occurrence of the /./ in the home directory field within /etc/passwd.

The default value is NO.

* user\_config\_dir — Specifies the path to a directory containing configuration files bearing the name of local system users that contain specific setting for that user. Any directive in the user's configuration file overrides those found in /etc/vsftpd/vsftpd.conf.

There is no default value for this directive.

### 22.5.5. Directory Options

The following lists directives which affect directories.

* dirlist\_enable — When enabled, users are allowed to view directory lists.

The default value is YES.

* dirmessage\_enable — When enabled, a message is displayed whenever a user enters a directory with a message file. This message resides within the current directory. The name of this file is specified in the message\_file directive and is .message by default.

The default value is NO. Note, in Red Hat Enterprise Linux, the value is set to YES.

* force\_dot\_files — When enabled, files beginning with a dot (.) are listed in directory listings, with the exception of the . and .. files.

The default value is NO.

* hide\_ids — When enabled, all directory listings show ftp as the user and group for each file.

The default value is NO.

* message\_file — Specifies the name of the message file when using the dirmessage\_enable directive.

The default value is .message.

* text\_userdb\_names — When enabled, test usernames and group names are used in place of UID and GID entries. Enabling this option may slow performance of the server.

The default value is NO.

* use\_localtime — When enabled, directory listings reveal the local time for the computer instead of GMT.

The default value is NO.

### 22.5.6. File Transfer Options

The following lists directives which affect directories.

* download\_enable — When enabled, file downloads are permitted.

The default value is YES.

* chown\_uploads — When enabled, all files uploaded by anonymous users are owned by the user specified in the chown\_username directive.

The default value is NO.

* chown\_username — Specifies the ownership of anonymously uploaded files if the chown\_uploads directive is enabled.

The default value is root.

* write\_enable — When enabled, FTP commands which can change the file system are allowed, such as DELE, RNFR, and STOR.

The default value is YES.

### 22.5.7. Logging Options

The following lists directives which affect vsftpd's logging behavior.

* dual\_log\_enable — When enabled in conjunction with xferlog\_enable, vsftpd writes two files simultaneously: a wu-ftpd-compatible log to the file specified in the xferlog\_file directive (/var/log/xferlog by default) and a standard vsftpd log file specified in the vsftpd\_log\_file directive (/var/log/vsftpd.log by default).

The default value is NO.

* log\_ftp\_protocol — When enabled in conjunction with xferlog\_enable and with xferlog\_std\_format set to NO, all FTP commands and responses are logged. This directive is useful for debugging.

The default value is NO.

* syslog\_enable — When enabled in conjunction with xferlog\_enable, all logging normally written to the standard vsftpd log file specified in the vsftpd\_log\_file directive (/var/log/vsftpd.log by default) is sent to the system logger instead under the FTPD facility.

The default value is NO.

* vsftpd\_log\_file — Specifies the vsftpd log file. For this file to be used, xferlog\_enable must be enabled and xferlog\_std\_format must either be set to NO or, if xferlog\_std\_format is set to YES, dual\_log\_enable must be enabled. It is important to note that if syslog\_enable is set to YES, the system log is used instead of the file specified in this directive.

The default value is /var/log/vsftpd.log.

* xferlog\_enable — When enabled, vsftpd logs connections (vsftpd format only) and file transfer information to the log file specified in the vsftpd\_log\_file directive (/var/log/vsftpd.log by default). If xferlog\_std\_format is set to YES, file transfer information is logged but connections are not, and the log file specified in xferlog\_file (/var/log/xferlog by default) is used instead. It is important to note that both log files and log formats are used if dual\_log\_enable is set to YES.

The default value is NO. Note, in Red Hat Enterprise Linux, the value is set to YES.

* xferlog\_file — Specifies the wu-ftpd-compatible log file. For this file to be used, xferlog\_enable must be enabled and xferlog\_std\_format must be set to YES. It is also used if dual\_log\_enable is set to YES.

The default value is /var/log/xferlog.

* xferlog\_std\_format — When enabled in conjunction with xferlog\_enable, only a wu-ftpd-compatible file transfer log is written to the file specified in the xferlog\_file directive (/var/log/xferlog by default). It is important to note that this file only logs file transfers and does not log connections to the server.

The default value is NO. Note, in Red Hat Enterprise Linux, the value is set to YES.

## Important

To maintain compatibility with log files written by the older wu-ftpd FTP server, the xferlog\_std\_format directive is set to YES under Red Hat Enterprise Linux. However, this setting means that connections to the server are not logged.

To both log connections in vsftpd format and maintain a wu-ftpd-compatible file transfer log, set dual\_log\_enable to YES.

If maintaining a wu-ftpd-compatible file transfer log is not important, either set xferlog\_std\_format to NO, comment the line with a hash mark (#), or delete the line entirely.

### 22.5.8. Network Options

The following lists directives which affect how vsftpd interacts with the network.

* accept\_timeout — Specifies the amount of time for a client using passive mode to establish a connection.

The default value is 60.

* anon\_max\_rate — Specifies the maximum data transfer rate for anonymous users in bytes per second.

The default value is 0, which does not limit the transfer rate.

* connect\_from\_port\_20 When enabled, vsftpd runs with enough privileges to open port 20 on the server during active mode data transfers. Disabling this option allows vsftpd to run with less privileges, but may be incompatible with some FTP clients.

The default value is NO. Note, in Red Hat Enterprise Linux, the value is set to YES.

* connect\_timeout — Specifies the maximum amount of time a client using active mode has to respond to a data connection, in seconds.

The default value is 60.

* data\_connection\_timeout — Specifies maximum amount of time data transfers are allowed to stall, in seconds. Once triggered, the connection to the remote client is closed.

The default value is 300.

* ftp\_data\_port — Specifies the port used for active data connections when connect\_from\_port\_20 is set to YES.

The default value is 20.

* idle\_session\_timeout — Specifies the maximum amount of time between commands from a remote client. Once triggered, the connection to the remote client is closed.

The default value is 300.

* listen\_address — Specifies the IP address on which vsftpd listens for network connections.

There is no default value for this directive.

## Tip

If running multiple copies of vsftpd serving different IP addresses, the configuration file for each copy of the vsftpd daemon must have a different value for this directive. Refer to [Section 22.4.1, “Starting Multiple Copies of vsftpd”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ftp-vsftpd-start.html#s2-ftp-vsftpd-start-multi) for more information about multihomed FTP servers.

* listen\_address6 — Specifies the IPv6 address on which vsftpd listens for network connections when listen\_ipv6 is set to YES.

There is no default value for this directive.

## Tip

If running multiple copies of vsftpd serving different IP addresses, the configuration file for each copy of the vsftpd daemon must have a different value for this directive. Refer to [Section 22.4.1, “Starting Multiple Copies of vsftpd”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-ftp-vsftpd-start.html#s2-ftp-vsftpd-start-multi) for more information about multihomed FTP servers.

* listen\_port — Specifies the port on which vsftpd listens for network connections.

The default value is 21.

* local\_max\_rate — Specifies the maximum rate data is transferred for local users logged into the server in bytes per second.

The default value is 0, which does not limit the transfer rate.

* max\_clients — Specifies the maximum number of simultaneous clients allowed to connect to the server when it is running in standalone mode. Any additional client connections would result in an error message.

The default value is 0, which does not limit connections.

* max\_per\_ip — Specifies the maximum of clients allowed to connected from the same source IP address.

The default value is 0, which does not limit connections.

* pasv\_address — Specifies the IP address for the public facing IP address of the server for servers behind Network Address Translation (NAT) firewalls. This enables vsftpd to hand out the correct return address for passive mode connections.

There is no default value for this directive.

* pasv\_enable — When enabled, passive mode connects are allowed.

The default value is YES.

* pasv\_max\_port — Specifies the highest possible port sent to the FTP clients for passive mode connections. This setting is used to limit the port range so that firewall rules are easier to create.

The default value is 0, which does not limit the highest passive port range. The value must not exceed 65535.

* pasv\_min\_port — Specifies the lowest possible port sent to the FTP clients for passive mode connections. This setting is used to limit the port range so that firewall rules are easier to create.

The default value is 0, which does not limit the lowest passive port range. The value must not be lower 1024.

* pasv\_promiscuous — When enabled, data connections are not checked to make sure they are originating from the same IP address. This setting is only useful for certain types of tunneling.

## Caution

Do not enable this option unless absolutely necessary as it disables an important security feature which verifies that passive mode connections originate from the same IP address as the control connection that initiates the data transfer.

The default value is NO.

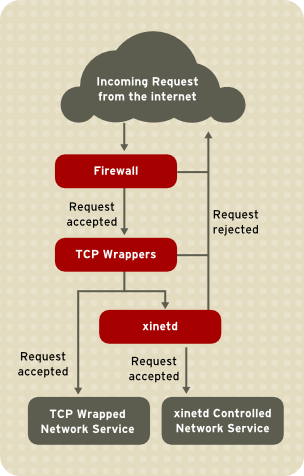
* port\_enable — When enabled, active mode connects are allowed.

The default value is YES.

## 42.5. TCP Wrappers and xinetd

Controlling access to network services is one of the most important security tasks facing a server administrator. Red Hat Enterprise Linux provides several tools for this purpose. For example, an iptables-based firewall filters out unwelcome network packets within the kernel's network stack. For network services that utilize it, *TCP Wrappers* add an additional layer of protection by defining which hosts are or are not allowed to connect to "wrapped" network services. One such wrapped network service is the xinetd super server. This service is called a super server because it controls connections to a subset of network services and further refines access control.

[Figure 42.9, “Access Control to Network Services”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-tcpwrappers.html#fig-access-flowchart) is a basic illustration of how these tools work together to protect network services.



**Figure 42.9. Access Control to Network Services**

This chapter focuses on the role of TCP Wrappers and xinetd in controlling access to network services and reviews how these tools can be used to enhance both logging and utilization management. Refer to [Section 42.9, “IPTables”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-iptables.html) for information about using firewalls with iptables.

### 42.5.1. TCP Wrappers

The TCP Wrappers package (tcp\_wrappers) is installed by default and provides host-based access control to network services. The most important component within the package is the /usr/lib/libwrap.a library. In general terms, a TCP-wrapped service is one that has been compiled against the libwrap.a library.

When a connection attempt is made to a TCP-wrapped service, the service first references the host's access files (/etc/hosts.allow and /etc/hosts.deny) to determine whether or not the client is allowed to connect. In most cases, it then uses the syslog daemon (syslogd) to write the name of the requesting client and the requested service to /var/log/secure or /var/log/messages.

If a client is allowed to connect, TCP Wrappers release control of the connection to the requested service and take no further part in the communication between the client and the server.

In addition to access control and logging, TCP Wrappers can execute commands to interact with the client before denying or releasing control of the connection to the requested network service.

Because TCP Wrappers are a valuable addition to any server administrator's arsenal of security tools, most network services within Red Hat Enterprise Linux are linked to the libwrap.a library. Some such applications include /usr/sbin/sshd, /usr/sbin/sendmail, and /usr/sbin/xinetd.

## Note

To determine if a network service binary is linked to libwrap.a, type the following command as the root user:

ldd <binary-name> | grep libwrap

Replace *<binary-name>* with the name of the network service binary.

If the command returns straight to the prompt with no output, then the network service is not linked to libwrap.a.

The following example indicates that /usr/sbin/sshd is linked to libwrap.a:

[root@myserver ~]# ldd /usr/sbin/sshd | grep libwrap

libwrap.so.0 => /usr/lib/libwrap.so.0 (0x00655000)

[root@myserver ~]#

#### 42.5.1.1. Advantages of TCP Wrappers

TCP Wrappers provide the following advantages over other network service control techniques:

* Transparency to both the client and the wrapped network service — Both the connecting client and the wrapped network service are unaware that TCP Wrappers are in use. Legitimate users are logged and connected to the requested service while connections from banned clients fail.
* Centralized management of multiple protocols — TCP Wrappers operate separately from the network services they protect, allowing many server applications to share a common set of access control configuration files, making for simpler management.

### 42.5.2. TCP Wrappers Configuration Files

To determine if a client is allowed to connect to a service, TCP Wrappers reference the following two files, which are commonly referred to as *hosts access* files:

* /etc/hosts.allow
* /etc/hosts.deny

When a TCP-wrapped service receives a client request, it performs the following steps:

1. It references */etc/hosts.allow*. — The TCP-wrapped service sequentially parses the /etc/hosts.allow file and applies the first rule specified for that service. If it finds a matching rule, it allows the connection. If not, it moves on to the next step.
2. It references */etc/hosts.deny*. — The TCP-wrapped service sequentially parses the /etc/hosts.deny file. If it finds a matching rule, it denies the connection. If not, it grants access to the service.

The following are important points to consider when using TCP Wrappers to protect network services:

* Because access rules in hosts.allow are applied first, they take precedence over rules specified in hosts.deny. Therefore, if access to a service is allowed in hosts.allow, a rule denying access to that same service in hosts.deny is ignored.
* The rules in each file are read from the top down and the first matching rule for a given service is the only one applied. The order of the rules is extremely important.
* If no rules for the service are found in either file, or if neither file exists, access to the service is granted.
* TCP-wrapped services do not cache the rules from the hosts access files, so any changes to hosts.allow or hosts.deny take effect immediately, without restarting network services.

## Warning

If the last line of a hosts access file is not a newline character (created by pressing the **Enter** key), the last rule in the file fails and an error is logged to either /var/log/messages or /var/log/secure. This is also the case for a rule that spans multiple lines without using the backslash character. The following example illustrates the relevant portion of a log message for a rule failure due to either of these circumstances:

warning: /etc/hosts.allow, line 20: missing newline or line too long

#### 42.5.2.1. Formatting Access Rules

The format for both /etc/hosts.allow and /etc/hosts.deny is identical. Each rule must be on its own line. Blank lines or lines that start with a hash (#) are ignored.

Each rule uses the following basic format to control access to network services:

*<daemon list>*: *<client list>* [: *<option>*: *<option>*: ...]

* *<daemon list>* — A comma-separated list of process names (not service names) or the ALL wildcard. The daemon list also accepts operators (refer to [Section 42.5.2.1.4, “Operators”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-tcpwrappers.html#s3-tcpwrappers-access-rules-op)) to allow greater flexibility.
* *<client list>* — A comma-separated list of hostnames, host IP addresses, special patterns, or wildcards which identify the hosts affected by the rule. The client list also accepts operators listed in [Section 42.5.2.1.4, “Operators”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-tcpwrappers.html#s3-tcpwrappers-access-rules-op) to allow greater flexibility.
* *<option>* — An optional action or colon-separated list of actions performed when the rule is triggered. Option fields support expansions, launch shell commands, allow or deny access, and alter logging behavior.

## Note

More information on the specialist terms above can be found elsewhere in this Guide:

* [Section 42.5.2.1.1, “Wildcards”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-tcpwrappers.html#s3-tcpwrappers-access-rules-wild)
* [Section 42.5.2.1.2, “Patterns”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-tcpwrappers.html#s3-tcpwrappers-access-rules-pat)
* [Section 42.5.2.2.4, “Expansions”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-tcpwrappers.html#s3-tcpwrappers-access-rules-options-exp)
* [Section 42.5.2.2, “Option Fields”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-tcpwrappers.html#s2-tcpwrappers-access-rules-options)

The following is a basic sample hosts access rule:

vsftpd : .example.com

This rule instructs TCP Wrappers to watch for connections to the FTP daemon (vsftpd) from any host in the example.com domain. If this rule appears in hosts.allow, the connection is accepted. If this rule appears in hosts.deny, the connection is rejected.

The next sample hosts access rule is more complex and uses two option fields:

sshd : .example.com \ : spawn /bin/echo `/bin/date` access denied>>/var/log/sshd.log \ : deny

Note that each option field is preceded by the backslash (\). Use of the backslash prevents failure of the rule due to length.

This sample rule states that if a connection to the SSH daemon (sshd) is attempted from a host in the example.com domain, execute the echo command to append the attempt to a special log file, and deny the connection. Because the optional deny directive is used, this line denies access even if it appears in the hosts.allow file. Refer to [Section 42.5.2.2, “Option Fields”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-tcpwrappers.html#s2-tcpwrappers-access-rules-options) for a more detailed look at available options.

##### 42.5.2.1.1. Wildcards

Wildcards allow TCP Wrappers to more easily match groups of daemons or hosts. They are used most frequently in the client list field of access rules.

The following wildcards are available:

* ALL — Matches everything. It can be used for both the daemon list and the client list.
* LOCAL — Matches any host that does not contain a period (.), such as localhost.
* KNOWN — Matches any host where the hostname and host address are known or where the user is known.
* UNKNOWN — Matches any host where the hostname or host address are unknown or where the user is unknown.
* PARANOID — Matches any host where the hostname does not match the host address.

## Caution

The KNOWN, UNKNOWN, and PARANOID wildcards should be used with care, because they rely on functioning DNS server for correct operation. Any disruption to name resolution may prevent legitimate users from gaining access to a service.

##### 42.5.2.1.2. Patterns

Patterns can be used in the client field of access rules to more precisely specify groups of client hosts.

The following is a list of common patterns for entries in the client field:

* Hostname beginning with a period (.) — Placing a period at the beginning of a hostname matches all hosts sharing the listed components of the name. The following example applies to any host within the example.com domain:
* ALL : .example.com
* IP address ending with a period (.) — Placing a period at the end of an IP address matches all hosts sharing the initial numeric groups of an IP address. The following example applies to any host within the 192.168.x.x network:
* ALL : 192.168.
* IP address/netmask pair — Netmask expressions can also be used as a pattern to control access to a particular group of IP addresses. The following example applies to any host with an address range of 192.168.0.0 through 192.168.1.255:
* ALL : 192.168.0.0/255.255.254.0

## Important

When working in the IPv4 address space, the address/prefix length (*prefixlen*) pair declarations (CIDR notation) are not supported. Only IPv6 rules can use this format.

* [IPv6 address]/prefixlen pair — [net]/prefixlen pairs can also be used as a pattern to control access to a particular group of IPv6 addresses. The following example would apply to any host with an address range of 3ffe:505:2:1:: through 3ffe:505:2:1:ffff:ffff:ffff:ffff:
* ALL : [3ffe:505:2:1::]/64
* The asterisk (\*) — Asterisks can be used to match entire groups of hostnames or IP addresses, as long as they are not mixed in a client list containing other types of patterns. The following example would apply to any host within the example.com domain:
* ALL : \*.example.com
* The slash (/) — If a client list begins with a slash, it is treated as a file name. This is useful if rules specifying large numbers of hosts are necessary. The following example refers TCP Wrappers to the /etc/telnet.hosts file for all Telnet connections:
* in.telnetd : /etc/telnet.hosts

Other, lesser used, patterns are also accepted by TCP Wrappers. Refer to the hosts\_access man 5 page for more information.

## Warning

Be very careful when using hostnames and domain names. Attackers can use a variety of tricks to circumvent accurate name resolution. In addition, disruption to DNS service prevents even authorized users from using network services. It is, therefore, best to use IP addresses whenever possible.

##### 42.5.2.1.3. Portmap and TCP Wrappers

Portmap's implementation of TCP Wrappers does not support host look-ups, which means portmap can not use hostnames to identify hosts. Consequently, access control rules for portmap in hosts.allow or hosts.deny must use IP addresses, or the keyword ALL, for specifying hosts.

Changes to portmap access control rules may not take effect immediately. You may need to restart the portmap service.

Widely used services, such as NIS and NFS, depend on portmap to operate, so be aware of these limitations.

##### 42.5.2.1.4. Operators

At present, access control rules accept one operator, EXCEPT. It can be used in both the daemon list and the client list of a rule.

The EXCEPT operator allows specific exceptions to broader matches within the same rule.

In the following example from a hosts.allow file, all example.com hosts are allowed to connect to all services except cracker.example.com:

ALL: .example.com EXCEPT cracker.example.com

In another example from a hosts.allow file, clients from the 192.168.0.*x* network can use all services except for FTP:

ALL EXCEPT vsftpd: 192.168.0.

## Note

Organizationally, it is often easier to avoid using EXCEPT operators. This allows other administrators to quickly scan the appropriate files to see what hosts are allowed or denied access to services, without having to sort through EXCEPT operators.

#### 42.5.2.2. Option Fields

In addition to basic rules that allow and deny access, the Red Hat Enterprise Linux implementation of TCP Wrappers supports extensions to the access control language through *option fields*. By using option fields in hosts access rules, administrators can accomplish a variety of tasks such as altering log behavior, consolidating access control, and launching shell commands.

##### 42.5.2.2.1. Logging

Option fields let administrators easily change the log facility and priority level for a rule by using the severity directive.

In the following example, connections to the SSH daemon from any host in the example.com domain are logged to the default authpriv syslog facility (because no facility value is specified) with a priority of emerg:

sshd : .example.com : severity emerg

It is also possible to specify a facility using the severity option. The following example logs any SSH connection attempts by hosts from the example.com domain to the local0 facility with a priority of alert:

sshd : .example.com : severity local0.alert

## Note

In practice, this example does not work until the syslog daemon (syslogd) is configured to log to the local0 facility. Refer to the syslog.conf man page for information about configuring custom log facilities.

##### 42.5.2.2.2. Access Control

Option fields also allow administrators to explicitly allow or deny hosts in a single rule by adding the allow or deny directive as the final option.

For example, the following two rules allow SSH connections from client-1.example.com, but deny connections from client-2.example.com:

sshd : client-1.example.com : allow

sshd : client-2.example.com : deny

By allowing access control on a per-rule basis, the option field allows administrators to consolidate all access rules into a single file: either hosts.allow or hosts.deny. Some administrators consider this an easier way of organizing access rules.

##### 42.5.2.2.3. Shell Commands

Option fields allow access rules to launch shell commands through the following two directives:

* spawn — Launches a shell command as a child process. This directive can perform tasks like using /usr/sbin/safe\_finger to get more information about the requesting client or create special log files using the echo command.

In the following example, clients attempting to access Telnet services from the example.com domain are quietly logged to a special file:

in.telnetd : .example.com \

: spawn /bin/echo `/bin/date` from %h>>/var/log/telnet.log \

: allow

* twist — Replaces the requested service with the specified command. This directive is often used to set up traps for intruders (also called "honey pots"). It can also be used to send messages to connecting clients. The twist directive must occur at the end of the rule line.

In the following example, clients attempting to access FTP services from the example.com domain are sent a message using the echo command:

vsftpd : .example.com \

: twist /bin/echo "421 This domain has been black-listed. Access denied!"

For more information about shell command options, refer to the hosts\_options man page.

##### 42.5.2.2.4. Expansions

Expansions, when used in conjunction with the spawn and twist directives, provide information about the client, server, and processes involved.

The following is a list of supported expansions:

* %a — Returns the client's IP address.
* %A — Returns the server's IP address.
* %c — Returns a variety of client information, such as the username and hostname, or the username and IP address.
* %d — Returns the daemon process name.
* %h — Returns the client's hostname (or IP address, if the hostname is unavailable).
* %H — Returns the server's hostname (or IP address, if the hostname is unavailable).
* %n — Returns the client's hostname. If unavailable, unknown is printed. If the client's hostname and host address do not match, paranoid is printed.
* %N — Returns the server's hostname. If unavailable, unknown is printed. If the server's hostname and host address do not match, paranoid is printed.
* %p — Returns the daemon's process ID.
* %s —Returns various types of server information, such as the daemon process and the host or IP address of the server.
* %u — Returns the client's username. If unavailable, unknown is printed.

The following sample rule uses an expansion in conjunction with the spawn command to identify the client host in a customized log file.

When connections to the SSH daemon (sshd) are attempted from a host in the example.com domain, execute the echo command to log the attempt, including the client hostname (by using the %h expansion), to a special file:

sshd : .example.com \

: spawn /bin/echo `/bin/date` access denied to %h>>/var/log/sshd.log \

: deny

Similarly, expansions can be used to personalize messages back to the client. In the following example, clients attempting to access FTP services from the example.com domain are informed that they have been banned from the server:

vsftpd : .example.com \

: twist /bin/echo "421 %h has been banned from this server!"

For a full explanation of available expansions, as well as additional access control options, refer to section 5 of the man pages for hosts\_access (man 5 hosts\_access) and the man page for hosts\_options.

Refer to [Section 42.5.5, “Additional Resources”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-tcpwrappers.html#s1-tcpwrappers-additional-resources) for more information about TCP Wrappers.

### 42.5.3. xinetd

The xinetd daemon is a TCP-wrapped *super service* which controls access to a subset of popular network services, including FTP, IMAP, and Telnet. It also provides service-specific configuration options for access control, enhanced logging, binding, redirection, and resource utilization control.

When a client attempts to connect to a network service controlled by xinetd, the super service receives the request and checks for any TCP Wrappers access control rules.

If access is allowed, xinetd verifies that the connection is allowed under its own access rules for that service. It also checks that the service can have more resources allotted to it and that it is not in breach of any defined rules.

If all these conditions are met (that is, access is allowed to the service; the service has not reached its resource limit; and the service is not in breach of any defined rule), xinetd then starts an instance of the requested service and passes control of the connection to it. After the connection has been established, xinetd takes no further part in the communication between the client and the server.

### 42.5.4. xinetd Configuration Files

The configuration files for xinetd are as follows:

* /etc/xinetd.conf — The global xinetd configuration file.
* /etc/xinetd.d/ — The directory containing all service-specific files.

#### 42.5.4.1. The /etc/xinetd.conf File

The /etc/xinetd.conf file contains general configuration settings which affect every service under xinetd's control. It is read when the xinetd service is first started, so for configuration changes to take effect, you need to restart the xinetd service. The following is a sample /etc/xinetd.conf file:

defaults

{

instances = 60

log\_type = SYSLOG authpriv

log\_on\_success = HOST PID

log\_on\_failure = HOST

cps = 25 30

}

includedir /etc/xinetd.d

These lines control the following aspects of xinetd:

* instances — Specifies the maximum number of simultaneous requests that xinetd can process.
* log\_type — Configures xinetd to use the authpriv log facility, which writes log entries to the /var/log/secure file. Adding a directive such as FILE /var/log/xinetdlog would create a custom log file called xinetdlog in the /var/log/ directory.
* log\_on\_success — Configures xinetd to log successful connection attempts. By default, the remote host's IP address and the process ID of the server processing the request are recorded.
* log\_on\_failure — Configures xinetd to log failed connection attempts or if the connection was denied.
* cps — Configures xinetd to allow no more than 25 connections per second to any given service. If this limit is exceeded, the service is retired for 30 seconds.
* includedir /etc/xinetd.d/ — Includes options declared in the service-specific configuration files located in the /etc/xinetd.d/ directory. Refer to [Section 42.5.4.2, “The /etc/xinetd.d/ Directory”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-tcpwrappers.html#s2-tcpwrappers-xinetd-config-files) for more information.

## Note

Often, both the log\_on\_success and log\_on\_failure settings in /etc/xinetd.conf are further modified in the service-specific configuration files. More information may therefore appear in a given service's log file than the /etc/xinetd.conf file may indicate. Refer to [Section 42.5.4.3.1, “Logging Options”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-tcpwrappers.html#s3-tcpwrappers-xinetd-alt-log) for further information.

#### 42.5.4.2. The /etc/xinetd.d/ Directory

The /etc/xinetd.d/ directory contains the configuration files for each service managed by xinetd and the names of the files correlate to the service. As with xinetd.conf, this directory is read only when the xinetd service is started. For any changes to take effect, the administrator must restart the xinetd service.

The format of files in the /etc/xinetd.d/ directory use the same conventions as /etc/xinetd.conf. The primary reason the configuration for each service is stored in a separate file is to make customization easier and less likely to affect other services.

To gain an understanding of how these files are structured, consider the /etc/xinetd.d/krb5-telnet file:

service telnet

{

flags = REUSE

socket\_type = stream

wait = no

user = root

server = /usr/kerberos/sbin/telnetd

log\_on\_failure += USERID

disable = yes

}

These lines control various aspects of the telnet service:

* service — Specifies the service name, usually one of those listed in the /etc/services file.
* flags — Sets any of a number of attributes for the connection. REUSE instructs xinetd to reuse the socket for a Telnet connection.

## Note

The REUSE flag is deprecated. All services now implicitly use the REUSE flag.

* socket\_type — Sets the network socket type to stream.
* wait — Specifies whether the service is single-threaded (yes) or multi-threaded (no).
* user — Specifies which user ID the process runs under.
* server — Specifies which binary executable to launch.
* log\_on\_failure — Specifies logging parameters for log\_on\_failure in addition to those already defined in xinetd.conf.
* disable — Specifies whether the service is disabled (yes) or enabled (no).

Refer to the xinetd.conf man page for more information about these options and their usage.

#### 42.5.4.3. Altering xinetd Configuration Files

A range of directives is available for services protected by xinetd. This section highlights some of the more commonly used options.

##### 42.5.4.3.1. Logging Options

The following logging options are available for both /etc/xinetd.conf and the service-specific configuration files within the /etc/xinetd.d/ directory.

The following is a list of some of the more commonly used logging options:

* ATTEMPT — Logs the fact that a failed attempt was made (log\_on\_failure).
* DURATION — Logs the length of time the service is used by a remote system (log\_on\_success).
* EXIT — Logs the exit status or termination signal of the service (log\_on\_success).
* HOST — Logs the remote host's IP address (log\_on\_failure and log\_on\_success).
* PID — Logs the process ID of the server receiving the request (log\_on\_success).
* USERID — Logs the remote user using the method defined in RFC 1413 for all multi-threaded stream services (log\_on\_failure andlog\_on\_success).

For a complete list of logging options, refer to the xinetd.conf man page.

##### 42.5.4.3.2. Access Control Options

Users of xinetd services can choose to use the TCP Wrappers hosts access rules, provide access control via the xinetd configuration files, or a mixture of both. Refer to [Section 42.5.2, “TCP Wrappers Configuration Files”](https://www.centos.org/docs/5/html/Deployment_Guide-en-US/ch-tcpwrappers.html#s1-tcpwrappers-access) for more information about TCP Wrappers hosts access control files.

This section discusses using xinetd to control access to services.

## Note

Unlike TCP Wrappers, changes to access control only take effect if the xinetd administrator restarts the xinetd service.

Also, unlike TCP Wrappers, access control through xinetd only affects services controlled by xinetd.

The xinetd hosts access control differs from the method used by TCP Wrappers. While TCP Wrappers places all of the access configuration within two files, /etc/hosts.allow and /etc/hosts.deny, xinetd's access control is found in each service's configuration file in the /etc/xinetd.d/ directory.

The following hosts access options are supported by xinetd:

* only\_from — Allows only the specified hosts to use the service.
* no\_access — Blocks listed hosts from using the service.
* access\_times — Specifies the time range when a particular service may be used. The time range must be stated in 24-hour format notation, HH:MM-HH:MM.

The only\_from and no\_access options can use a list of IP addresses or host names, or can specify an entire network. Like TCP Wrappers, combining xinetd access control with the enhanced logging configuration can increase security by blocking requests from banned hosts while verbosely recording each connection attempt.

For example, the following /etc/xinetd.d/telnet file can be used to block Telnet access from a particular network group and restrict the overall time range that even allowed users can log in:

service telnet

{

disable = no

flags = REUSE

socket\_type = stream

wait = no

user = root

server = /usr/kerberos/sbin/telnetd

log\_on\_failure += USERID

no\_access = 172.16.45.0/24

log\_on\_success += PID HOST EXIT

access\_times = 09:45-16:15

}

In this example, when a client system from the 10.0.1.0/24 network, such as 10.0.1.2, tries to access the Telnet service, it receives the following message:

Connection closed by foreign host.

In addition, their login attempts are logged in /var/log/messages as follows:

Sep 7 14:58:33 localhost xinetd[5285]: FAIL: telnet address from=172.16.45.107

Sep 7 14:58:33 localhost xinetd[5283]: START: telnet pid=5285 from=172.16.45.107

Sep 7 14:58:33 localhost xinetd[5283]: EXIT: telnet status=0 pid=5285 duration=0(sec)

When using TCP Wrappers in conjunction with xinetd access controls, it is important to understand the relationship between the two access control mechanisms.

The following is the sequence of events followed by xinetd when a client requests a connection:

1. The xinetd daemon accesses the TCP Wrappers hosts access rules using a libwrap.a library call. If a deny rule matches the client, the connection is dropped. If an allow rule matches the client, the connection is passed to xinetd.
2. The xinetd daemon checks its own access control rules both for the xinetd service and the requested service. If a deny rule matches the client, the connection is dropped. Otherwise, xinetd starts an instance of the requested service and passes control of the connection to that service.

## Important

Care should be taken when using TCP Wrappers access controls in conjunction with xinetd access controls. Misconfiguration can cause undesirable effects.

##### 42.5.4.3.3. Binding and Redirection Options

The service configuration files for xinetd support binding the service to an IP address and redirecting incoming requests for that service to another IP address, hostname, or port.

Binding is controlled with the bind option in the service-specific configuration files and links the service to one IP address on the system. When this is configured, the bind option only allows requests to the correct IP address to access the service. You can use this method to bind different services to different network interfaces based on requirements.

This is particularly useful for systems with multiple network adapters or with multiple IP addresses. On such a system, insecure services (for example, Telnet), can be configured to listen only on the interface connected to a private network and not to the interface connected to the Internet.

The redirect option accepts an IP address or hostname followed by a port number. It configures the service to redirect any requests for this service to the specified host and port number. This feature can be used to point to another port number on the same system, redirect the request to a different IP address on the same machine, shift the request to a totally different system and port number, or any combination of these options. A user connecting to a certain service on a system may therefore be rerouted to another system without disruption.

The xinetd daemon is able to accomplish this redirection by spawning a process that stays alive for the duration of the connection between the requesting client machine and the host actually providing the service, transferring data between the two systems.

The advantages of the bind and redirect options are most clearly evident when they are used together. By binding a service to a particular IP address on a system and then redirecting requests for this service to a second machine that only the first machine can see, an internal system can be used to provide services for a totally different network. Alternatively, these options can be used to limit the exposure of a particular service on a multi-homed machine to a known IP address, as well as redirect any requests for that service to another machine especially configured for that purpose.

For example, consider a system that is used as a firewall with this setting for its Telnet service:

service telnet

{

socket\_type = stream

wait = no

server = /usr/kerberos/sbin/telnetd

log\_on\_success += DURATION USERID

log\_on\_failure += USERID

bind = 123.123.123.123

redirect = 10.0.1.13 23

}

The bind and redirect options in this file ensure that the Telnet service on the machine is bound to the external IP address (123.123.123.123), the one facing the Internet. In addition, any requests for Telnet service sent to 123.123.123.123 are redirected via a second network adapter to an internal IP address (10.0.1.13) that only the firewall and internal systems can access. The firewall then sends the communication between the two systems, and the connecting system thinks it is connected to 123.123.123.123 when it is actually connected to a different machine.

This feature is particularly useful for users with broadband connections and only one fixed IP address. When using Network Address Translation (NAT), the systems behind the gateway machine, which are using internal-only IP addresses, are not available from outside the gateway system. However, when certain services controlled by xinetd are configured with the bind and redirect options, the gateway machine can act as a proxy between outside systems and a particular internal machine configured to provide the service. In addition, the various xinetd access control and logging options are also available for additional protection.

##### 42.5.4.3.4. Resource Management Options

The xinetd daemon can add a basic level of protection from Denial of Service (DoS) attacks. The following is a list of directives which can aid in limiting the effectiveness of such attacks:

* per\_source — Defines the maximum number of instances for a service per source IP address. It accepts only integers as an argument and can be used in both xinetd.conf and in the service-specific configuration files in the xinetd.d/ directory.
* cps — Defines the maximum number of connections per second. This directive takes two integer arguments separated by white space. The first argument is the maximum number of connections allowed to the service per second. The second argument is the number of seconds that xinetd must wait before re-enabling the service. It accepts only integers as arguments and can be used in either the xinetd.conf file or the service-specific configuration files in the xinetd.d/ directory.
* max\_load — Defines the CPU usage or load average threshold for a service. It accepts a floating point number argument.

The load average is a rough measure of how many processes are active at a given time. See the uptime, who, and procinfo commands for more information about load average.

There are more resource management options available for xinetd. Refer to the xinetd.conf man page for more information.

### 42.5.5. Additional Resources

More information about TCP Wrappers and xinetd is available from system documentation and on the Internet.

#### 42.5.5.1. Installed Documentation

The documentation on your system is a good place to start looking for additional configuration options for TCP Wrappers, xinetd, and access control.

* /usr/share/doc/tcp\_wrappers-*<version>*/ — This directory contains a README file that discusses how TCP Wrappers work and the various hostname and host address spoofing risks that exist.
* /usr/share/doc/xinetd-*<version>*/ — This directory contains a README file that discusses aspects of access control and a sample.conf file with various ideas for modifying service-specific configuration files in the /etc/xinetd.d/ directory.
* TCP Wrappers and xinetd-related man pages — A number of man pages exist for the various applications and configuration files involved with TCP Wrappers and xinetd. The following are some of the more important man pages:

Server Applications

* + man xinetd — The man page for xinetd.

Configuration Files

* + man 5 hosts\_access — The man page for the TCP Wrappers hosts access control files.
  + man hosts\_options — The man page for the TCP Wrappers options fields.
  + man xinetd.conf — The man page listing xinetd configuration options.

#### 42.5.5.2. Useful Websites

* [http://www.xinetd.org/](http://www.xinetd.org) — The home of xinetd, containing sample configuration files, a full listing of features, and an informative FAQ.
* <http://www.macsecurity.org/resources/xinetd/tutorial.shtml> — A thorough tutorial that discusses many different ways to optimize default xinetd configuration files to meet specific security goals.

#### 42.5.5.3. Related Books

* *Hacking Linux Exposed* by Brian Hatch, James Lee, and George Kurtz; Osbourne/McGraw-Hill — An excellent security resource with information about TCP Wrappers and xinetd.