Chapter 15

The Samba File Server

CERTIFICATION OBJECTIVES

Samba is the Linux implementation of the networking protocols used to connect Microsoft operating systems. In a network of Microsoft Windows computers, file sharing is based on the Common Internet File System (CIFS), which was developed from the Server Message Block (SMB) protocol. Samba was developed as a freely available SMB server for all Unix-related operating systems, including Linux, and has been upgraded to support CIFS.

Samba interacts with CIFS so transparently that Microsoft clients cannot tell your Linux server from a genuine Windows server. With Samba running only on Linux machines, there are no server, client, or client access licenses to purchase. If you can learn to edit the main Samba configuration file from the command-line interface, you can configure Samba quickly.

Learn to test network services such as Samba. These are services that you might configure and/or troubleshoot on the Red Hat exams. Take some time to understand the configuration files associated with each of these services, and practice making them work on different Linux systems. In some cases, two or more systems running Linux will be useful to practice what you learn in this chapter.

INSIDE THE EXAM

Inside the Exam

This chapter directly addresses two RHCE objectives related to Samba File System services. When you're finished with this chapter, you'll know how to

- Provide network shares to specific clients
- Provide network shares suitable for group collaboration

With Samba, communications are seamless with Microsoft clients. However, because you might not have access to Microsoft Windows during the Red Hat exams, you'll see how Samba communications are also seamless with other Linux clients. Shares can be limited to specific clients with Samba and other security options.

Samba also provides support for group collaboration, as does Apache in Chapter 14. The principles are the same as the way group directories were configured on Linux in Chapter 8.

Do not forget the standard requirements for all network services, discussed in Chapters 10 and 11. To review, you need to install the service, make it work with SELinux, make sure it starts on boot, configure the service for basic operation, and set up user- and host-based security.

CERTIFICATION OBJECTIVE 15.01

Samba Services

Microsoft's CIFS was built on the Server Message Block (SMB) protocol. SMB was developed in the 1980s by IBM, Microsoft, and Intel as a way to share files and printers over a network.

As Microsoft developed SMB into CIFS, the Samba developers have upgraded Samba accordingly. Samba services provide a stable, reliable, fast, and highly compatible file and print sharing service that allows your computer to act as a client, a member server,

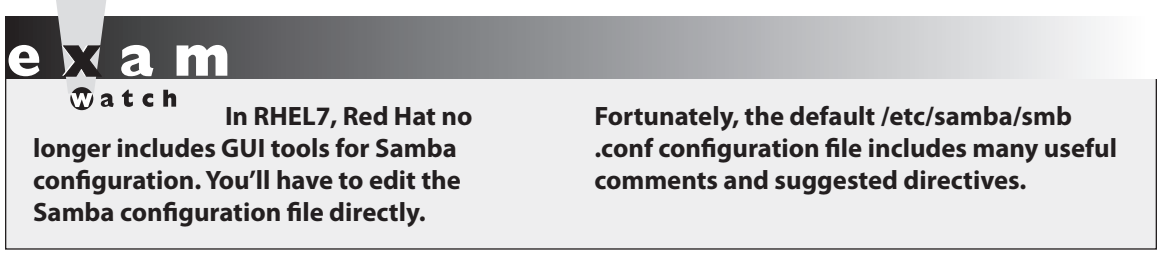
a Primary Domain Controller (PDC), or a member of an Active Directory (AD) service on Microsoft-based networks.

With the release of Samba version 4.0, you can now configure Samba to act as an AD Domain Controller (DC) on a Microsoft-based network. Although the configuration of a DC is outside the scope of the RHCE exam, it is an important skill for many systems administrators.

It is easy to configure Samba to do a number of things on a Microsoft-based network. Here are some examples:

- Participate in a Microsoft Windows Workgroup or a domain as a client, member server, or even a PDC.
- Provide user/password and share directory databases locally, either from another Samba server or from a Windows PDC.
- Configure local directories as shared SMB filesystems.

Samba can do more, but you get the idea. Samba features are configured through one very big file, smb.conf, in the /etc/samba directory. Although this file may intimidate some users, read the comments therein. You might be surprised at what you can do with Samba!



In RHEL7, Red Hat no longer includes GUI tools for Samba configuration. You'll have to edit the Samba configuration file directly.

Fortunately, the default /etc/samba/smb.conf configuration file includes many useful comments and suggested directives.

Install Samba Services

The process of installing Samba differs from other servers. Samba packages are not organized in a single package group. While all you need to set up a Samba server is the Samba RPM package (and dependencies), you'll find other helpful Samba packages. Important Samba packages are described in Table 15-1.

Some Samba Background

Samba services provide interoperability between Microsoft Windows and Linux/Unix systems. Before configuring Samba, you need a basic understanding of how Microsoft Windows networking works with TCP/IP.

TABLE 15-1 Samba Packages

RPM Package	Description
samba	Includes the basic SMB server software for sharing files and printers.
samba-client	Provides the utilities needed to connect to SMB/CIFS shares and printers.
samba-common	Contains common Samba files used by both the client and the server.
samba-dc	Included in the RHEL Server Optional repository; provides integration with Active Directory.
samba-libs	Contains the libraries needed by the other Samba software packages.
samba-python	Contains the Python modules needed by the other Samba software packages.
samba-winbind	Provides the Winbind daemon, which enables Samba to be a member server on Microsoft-based domains and supports Windows users on Linux servers.
samba-winbind-krb5-locator	Included in the RHEL Server Optional repository; allows the local Kerberos library to use the same KDC as Samba.
samba-winbind-modules	Provides a client connection to the Winbind daemon via PAM and the Network Service Switch (NSS).

The original Microsoft Windows networks were configured with computer hostnames, known as NetBIOS names, limited to 15 characters. These unique hostnames provided a simple, flat hostname system for the computers on a LAN.

All computer identification requests were made through broadcasts. This overall network transport system was known as the NetBIOS Extended User Interface (NetBEUI), and was not "routable." In other words, it did not allow communication between different networks. As a result, administrators could configure only 100–200 nodes on the original Microsoft-based PC networks.

Microsoft needed a solution to route information between networks. They could have used the Novell IPX/SPX protocol stack, but that was not good enough. As the Internet grew, Microsoft needed a standard compatible with TCP/IP. Microsoft adapted its NetBIOS system to TCP/IP with SMB. Since Microsoft published SMB as an industry-wide standard, anyone could set up their own service to work with it. As Microsoft has moved toward CIFS, Samba developers have adapted to the changes.

One of the features of Windows networks is the browser service. All computers register their NetBIOS names with one "elected" master browser, the keeper of the database of network-wide services. In fact, a browse database is maintained by some elected host for every protocol running on the Microsoft-based network. For instance, if the NetBEUI and TCP/IP protocols were installed on a host, then two duplicate browse databases were required—one per protocol—because the services available might differ between protocols.

Ports, Firewalls, and Samba

Samba as a service and a client requires access through multiple network ports. To enable communications through the local firewall for a Samba server, run the following commands:

```
# firewall-cmd --permanent --add-service=samba
# firewall-cmd --reload
```

For Samba clients, you need to enable the following firewalld service:

```
# firewall-cmd --permanent --add-service=samba-client
# firewall-cmd --reload
```

The configuration of the corresponding firewalld services is defined in the files samba.xml and samba-client.xml, located in the /usr/lib/firewalld/services directory. As shown in Figure 15-1 and Figure 15-2, the Samba server requires four open ports, whereas the Samba Client service requires only two open ports.

You'll note that two of the open ports are associated with the User Datagram Protocol (UDP), which is not connection based. Therefore, to track locally originating NetBIOS requests and allow packets matching the corresponding returning traffic through the firewall, both the Samba and the Samba Client services use the firewalld helper module **ns_conntrack_netbios_ns**.

FIGURE 15-1

Configuration file for the Samba firewalld service

```
[root@server1 ~]# cat /usr/lib/firewalld/services/samba.xml
<?xml version="1.0" encoding="utf-8"?>
<service>
  <short>Samba</short>
  <description>This option allows you to access and participate in Windows file
and printer sharing networks. You need the samba package installed for this opti
on to be useful.</description>
  <port protocol="udp" port="137"/>
  <port protocol="udp" port="138"/>
  <port protocol="tcp" port="139"/>
  <port protocol="tcp" port="445"/>
  <module name="nf_conntrack_netbios_ns"/>
</service>
[root@server1 ~]#
```

FIGURE 15-2

Configuration file for the Samba Client firewalld service

```
[root@server1 ~]# cat /usr/lib/firewalld/services/samba-client.xml
<?xml version="1.0" encoding="utf-8"?>
<service>
  <short>Samba Client</short>
  <description>This option allows you to access Windows file and printer sharing
 networks. You need the samba-client package installed for this option to be use
ful.</description>
  <port protocol="udp" port="137"/>
  <port protocol="udp" port="138"/>
  <module name="nf_conntrack_netbios_ns"/>
</service>
[root@server1 ~]#
```

TABLE 15-2

Samba Communication Services

Port/Protocol	Description
137/UDP	NetBIOS name service
138/UDP	NetBIOS datagram service
139/TCP	NetBIOS session service
445/TCP	Samba over TCP/IP

The TCP and UDP ports needed for the Samba client are shown in Table 15-2. Collectively, the services that listen on ports 137–139 are associated with NetBIOS communication over TCP/IP (NBT).

Configure SELinux Booleans for Samba

Several directives are associated with making a Samba server work with SELinux in targeted mode, as described in Table 15-3. You may have to activate one or more booleans to support different Samba functions. While this may seem repetitive for some readers, SELinux is not well understood even by many Linux experts.

If you want to allow Samba to share local home directories with others on the network, run the following command:

```
# setsebool -P samba_enable_home_dirs 1
```

The **-P** makes sure the change survives a reboot.

If you need to configure directories to be shared through other services, it's appropriate to enable the samba_export_all_ro or samba_export_all_rw boolean. For example, files that are shared via an Apache web server must be labeled with the httpd_sys_content_t file context. However, Samba requires a different SELinux context for files and directories, samba_share_t. To allow Samba to access files that don't have the samba_share_t label, you must enable the samba_export_all_ro or samba_export_all_rw boolean.

Configure SELinux File Contexts for Samba

Normally, Samba can only share those files and directories labeled with the samba_share_t file type. However, the samba_share_t file type is not required if the samba_export_all_ro or samba_export_all_rw booleans is enabled. It is also not required if the files are labeled with public_content_rw_t or public_content_t and the smbd_anon_write boolean is enabled.

However, enabling samba_export_all_rw may be a security risk because it would allow all directories to be shared by Samba, regardless of their SELinux context. So in most cases, you'll want to label directories (and the files therein) with the samba_share_t SELinux context by running a command such as the following:

```
# chcon -R -t samba_share_t /share
```

TABLE 15-3 Samba SELinux Booleans

Boolean	Description
cdrecord_read_content	Allows the **cdrecord** command to read shared Samba (and other network) directories
ksmtuned_use_cifs	Allows KSM (a service that reduces identical memory pages into a single page) to access CIFS filesystems
samba_create_home_dirs	Supports the creation of home directories, normally via the pam_mkhomedir.so PAM module
samba_domain_controller	Allows Samba to act as a Domain Controller for authentication management
samba_enable_home_dirs	Enables the sharing of user home directories
samba_export_all_ro	Sets up read-only access to any directory, even those without the samba_share_t file type label
samba_export_all_rw	Sets up read/write access to any directory, even those without the samba_share_t file type label
samba_portmapper	Allows Samba to act as a portmapper
samba_run_unconfined	Supports the execution of unconfined scripts from the /var/lib/samba/scripts directory
samba_share_fusefs	Allows Samba to share filesystems mounted via fusefs, such as GlusterFS filesystems
samba_share_nfs	Enables sharing of NFS filesystems from Samba
smbd_anon_write	Allows Samba to modify files on public directories configured with the **public_content_rw_t** and **public_content_r_t** SELinux contexts
sanlock_use_samba	Allows sanlock (a shared storage lock manager) to access CIFS filesystems
use_samba_home_dirs	Supports the use of a remote Samba server for local home directories
virt_sandbox_use_samba	Allows sandbox containers to access CIFS filesystems
virt_use_samba	Allows a VM to access files mounted to the CIFS filesystem

In addition, to make sure the changes survive a relabel of SELinux, you'll want to change the default file context policy defined in the file_contexts.local file in the /etc/selinux/targeted/contexts/files directory. This can be achieved with a command such as the following:

```
# semanage fcontext -a -t samba_share_t '/share(/.*)?'
```

Samba Daemons

The sharing of directories and printers on a Microsoft-style network requires several daemons and a number of related commands. Working together, the commands can help configure Samba, and the daemons help it communicate through the different communication ports described earlier in this chapter.

Samba includes commands that run the service, as well as aid in configuration. The most important of these commands are the binary files in the /usr/sbin directory that start the various Samba services.

You need two daemons to run Samba: the main Samba service (**smbd**) and the NetBIOS name service (**nmbd**). In addition, some administrators may want to run the Winbind service (**winbindd**) for user and hostname resolution. This is provided by the samba-winbind RPM package. All three daemons are configured through the /etc/samba/smb.conf configuration file.

If you want to make sure the services are running the next time Linux is booted, execute the following command:

```
# systemctl enable smb nmb winbind
```

You can start the associated **smbd**, **nmbd**, and **winbindd** daemons with the following command:

```
# systemctl start smb nmb winbind
```

The **systemctl** command can help you confirm the way a daemon is running. For example, the command listed in Figure 15-3 confirms that the Samba service is running with main PID 14691.

Samba Server Global Configuration

You can configure a Samba server through the main Samba configuration file, /etc/samba/smb.conf. This file is long and includes a number of directives that require some understanding of

FIGURE 15-3

Showing status information about the smb service unit

```
[root@server1 rhel7]# systemctl status smb
smb.service - Samba SMB Daemon
   Loaded: loaded (/usr/lib/systemd/system/smb.service; disabled)
   Active: active (running) since Mon 2015-05-04 11:29:46 BST; 1min 20s ago
 Main PID: 14691 (smbd)
   Status: "smbd: ready to serve connections..."
   CGroup: /system.slice/smb.service
           ├─14691 /usr/sbin/smbd
           └─14692 /usr/sbin/smbd

May 04 11:29:46 server1.example.net systemd[1]: Starting Samba SMB Daemon...
May 04 11:29:46 server1.example.net smbd[14691]: [2015/05/04 11:29:46.609245,...)
May 04 11:29:46 server1.example.net systemd[1]: Started Samba SMB Daemon.
Hint: Some lines were ellipsized, use -l to show in full.
[root@server1 rhel7]#
```

the concepts associated with Microsoft Windows networking. Fortunately, the default version of this file also includes helpful documentation with suggestions and useful options.

The default Samba configuration file includes a number of non-default commented directives. You can find the default value of such directives in the man page for the smb.conf file.

Before editing this file, study its contents. Once you understand how the file is structured, back it up and then try editing the file directly. Check the syntax of smb.conf using the **testparm** command. Test the result of your changes by reloading the Samba server with the following command:

```
# systemctl reload smb
```

To help you with this process, we'll analyze the default RHEL 7 version of this file. The excerpts shown next are essentially a complete view of this file. In some cases, we've replaced the comments in the file with our own explanations. You might want to browse your own /etc/samba/smb.conf file as well.

The smb.conf file includes two types of comment lines. The hash symbol (#) is used as a general text comment. This is typically verbiage that describes a feature. The second comment symbol is the semicolon (;), used to comment out Samba directives (which you may later wish to uncomment to enable a feature).

(Note that the physical dimensions of this book limit the lengths of lines of code. In a few cases, we've modified the code lines slightly to meet this limitation without changing the intent of any command in this configuration file.)

```
# This is the main Samba configuration file. For detailed information
# about the options listed here, refer to the smb.conf(5) manual page.
# Samba has a huge number of configurable options, most of which are
# not shown in this example.

# Note: Run the "testparm" command after modifying this file to check
# for basic syntax errors.
```

exam Watch

As stated in the Red Hat Exam Prep guide, RHCEs must be able to configure various services, including Samba, for basic operation. Some of the details of the default version of the main Samba configuration go beyond basic operation.

While you need to know what can be done with different global settings, you should change as little as possible. The less you change, the less can go wrong. Perfect configuration files are not required. Configuration files that meet the specific requirements of an exam or a job are.

The smb.conf file can be split into different sections. After the first set of comments, there is another commented block with a description of the most important SELinux booleans and file contexts related to Samba. Due to lack of

space, we only show the first lines of this section. Do read through all the comments of the smb.conf file, because they are a very useful source of information:

```
# Security-Enhanced Linux (SELinux) Notes:
#
# Turn the samba_enable_home_dirs Boolean on if you want to share home
# directories via Samba. Run the following command as the root user to turn
# this boolean on:
# setsebool -P samba_enable_home_dirs on
#
# If you create a new directory, such as a new top-level directory, label it
# with samba_share_t so that SELinux allows Samba to read and write to it. Do
# not label system directories, such as /etc/ and /home/, with samba_share_t,
# as such directories should already have an SELinux label.
```

Then, there is a global settings section that defines the overall attributes of a server. This section starts with the following two lines:

```
#======================= Global Settings=======================
[global]
```

Now examine the global settings that follow. First, if you see the line

```
#--authconfig--start-line--
```

this means the configuration file has been modified by the **authconfig** or the **system-config-authentication** tool.

Network-Related Options

Scroll down to the subsection entitled

```
#----------- Network Related Options ---------------
```

Examine each of the directives in this part of the Global Settings section. Despite the name, the **workgroup** variable may specify the name of a workgroup or, more commonly, a domain. However, because peer-to-peer workgroups were developed first, the default Samba **workgroup** is **WORKGROUP**, which happens to be the default peer-to-peer workgroup on older Microsoft Windows systems. On RHEL 7, it's now set to the following value:

```
workgroup = MYGROUP
```

The **server string** directive that follows becomes the comment shown with the NetBIOS name of the system in the visible browse list, where Samba substitutes the version number for the **%v** variable:

```
server string = Samba Server Version %v
```

It's a good idea to add a NetBIOS name for the local system to this file. While limited to 15 characters, it can be the same as the hostname used for the system. This becomes what other clients see in network browse lists, such as those shown from a Microsoft **net view** command or a Linux **smbclient** command.

```
;   netbios name = MYSERVER
```

If the local system is connected to more than one network, you can specify the interfaces Samba should listen to with the **interfaces** directive, as shown here. Of course, the devices and network addresses should be changed appropriately.

```
;   interfaces = lo eth0 192.168.12.2/24 192.168.13.2/24
```

If you activate the **hosts allow** directive, that action can limit access to the specified network(s). The following default would limit access to the hosts within the 192.168.12.0/24 and 192.168.13.0/24 networks, as well as the local computer (127.):

```
;   hosts allow = 127. 192.168.12. 192.168.13.
```

It's possible to configure a **hosts deny** directive in a similar fashion. With such directives, you can set up host-based security for Samba. In the global section, such security would apply server-wide. You can also use the **hosts allow** and **hosts deny** directives in the definitions for individual shared directories, as described later in this chapter.

Next, the **max protocol** directive specifies the highest protocol level that the Samba server will support. By default, this is set to the Windows 7 SMB2 version:

```
;   max protocol = SMB2
```

Logging Options

The next section sets up logging options, as indicated by the following line:

```
#----------- Logging Options ---------------
```

The **log file** directive, as shown, sets up separate log files for every machine that connects to this Samba server, based on its machine name (**%m**). By default, the log file is limited to 50KB. As suggested by the comment, log files that exceed the given size are rotated. If logs exceed that size, you'll still see them in the /var/log/samba directory with the .old extension.

```
# log files split per-machine
log file = /var/log/samba/log.%m
# maximum size of 50KB per log file, then rotate:
max log size = 50
```

Standalone Server Options

The following section sets up security options, based on configuration as a standalone server:

```
#----------- Standalone Server Options --------------
```

The **security** directive may be a bit confusing. The standard value of the directive, as shown here, means that connections check the local password database. It is appropriate when configuring this computer as a Domain Controller (DC), specifically a Primary Domain Controller (PDC).

```
security = user
```

Alternatively, you can configure this computer as a member server on a domain to use a password database from a DC. In that case, you would substitute the following command:

```
security = domain
```

To set up a Linux system as a workstation that happens to share directories on a Microsoft domain, you'll need to set up the computer as a member server on that domain.

Alternatively, to configure a system as a domain member on an Active Directory Services realm, substitute the following command:

```
security = ads
```

Finally, note that the **server** and **share** values are deprecated. To summarize, there are three basic authentication options: **user**, **domain**, and **ads**.

Now, refocus this directive on the authentication database. The default is **security = user**; in this case, make sure to create Samba usernames and passwords to populate the user database. If the database is local, it could be either

```
passdb backend = smbpasswd
```

or

```
passdb backend = tdbsam
```

The smbpasswd database is stored in the local /etc/samba directory. The tdbsam option, short for the Trivial Database Security Accounts Manager, is the recommended database format and is stored in the /var/lib/samba/private directory.

Alternatively, for a remote database such as LDAP, you could activate the following directive. If the LDAP server is located on a remote system, that Uniform Resource Identifier (URI) address can be included here.

```
passdb backend = ldapsam
```

If you've set up **security = ads**, you'll also want to activate the following directive to specify the Active Directory (AD) realm, substituting the actual AD realm for *MY_REALM:*

```
;   realm = MY_REALM
```

Domain Controller Options

The following section supports the configuration of a system as a Domain Controller, starting with the following comment:

```
#----------- Domain Controller Options --------------
```

Additional configuration is required for a Samba server configured as a Domain Controller. In brief, these options specify the role of the system as the domain master and as the system that receives requests for logins to the domain:

```
;  domain master = yes
;  domain logons = yes
```

The next directive sets up Microsoft command-line logon scripts by computer and user. The directive afterward stores Microsoft user profiles on the local Samba server:

```
# the following login script name is determined by the machine name
# (%m):
;   logon script = %m.bat
# the following login script name is determined by the UNIX user
# used:
;   logon script = %u.bat
;   logon path = \\%L\Profiles\%u
```

The remaining configuration options are fairly self-explanatory as scripts that add and delete users, groups, and machine accounts:

```
;  add user script = /usr/sbin/useradd "%u" -n -g users
;  add group script = /usr/sbin/groupadd "%g"
;  add machine script = /usr/sbin/adduser -n -c ↵
      "Workstation (%u)" -M -d /nohome -s /bin/false "%u"
;  delete user script = /usr/sbin/userdel "%u"
;  delete user from group script = /usr/sbin/userdel "%u" "%g"
;  delete group script = /usr/sbin/groupdel "%g"
```

Browse Control Options

The following section controls whether and how a system may be configured as a browse master, which maintains a list of resources on the network. Related directives start with the following comment:

```
#----------- Browser Control Options --------------
```

If the **local master** option is enabled, Samba participates in browser elections like any other Microsoft Windows computer, using the specified **os level** priority:

```
;  local master = no
;  os level = 33
```

You can force a local browser election when **nmbd** starts, using the **preferred master** directive:

```
;  preferred master = yes
```

Name Resolution

The following section allows you to set up a Samba server with a database of NetBIOS names and IP addresses. The section starts with the following comment:

```
#----------- Name Resolution --------------
```

The Windows Internet Name Service (WINS) is functionally similar to DNS on Microsoft-based networks. If you activate the following configuration option, the **nmbd** daemon will enable a WINS server on the local computer:

```
;   wins support = yes
```

Alternatively, you can point the local computer to a remote WINS server on the network; of course, you'd have to substitute the IP address for *w.x.y.z.* Do not activate both the **wins support** and **wins server** directives on the same system, as they are mutually exclusive.

```
;   wins server = w.x.y.z
```

WINS may not be supported on some clients. In that case, you could enable the following directive to allow Samba to answer resolution queries on behalf of those clients:

```
;   wins proxy = yes
```

If Samba is acting as a WINS server and a NetBIOS name has not been registered, the following directive would enable a DNS name lookup through the locally configured DNS servers:

```
;   dns proxy = yes
```

Printing Options

Printers were included in the RHCT exam objectives for RHEL 5. However, they are not listed in either the RHCSA or the RHCE objectives for RHEL 6 and RHEL 7. Nevertheless,

printing is part of the default Samba server configuration, so you should at least scan the corresponding section in the Samba configuration file, starting with the following comment:

```
#----------- Printing Options --------------
```

These default printer settings are required to share printers from this Samba server. The following three directives load printers as defined by **printcap name = /etc/printcap**. The **cups options = raw** directive means that CUPS sends data to the printer in "raw" mode—that is, without any additional filtering. This setting is used when the printer driver is installed on the Windows clients so that no additional data processing is required by CUPS:

```
load printers = yes
cups options = raw
printcap name = /etc/printcap
```

Alternatively, it's possible to configure a different print server. The following option obtains information from printers configured on System V hosts, which rely on the **lpstat** command to list available printers:

```
printcap name = lpstat
```

Filesystem Options

The following section configures how Microsoft Disk Operating System (DOS) filesystem attributes are stored. The best option to provide support for DOS attributes is to store the Samba shares on a filesystem that supports extended attributes, such as XFS. Alternatively, Samba can create maps between DOS attributes and Unix permissions bits. The following directives define defaults for all shared directories:

```
#----------- File System Options --------------
```

First, the **map archive** directive can control whether the DOS file archive attribute is mapped to the local file owner execute bit, if supported by the **create mask** directive:

```
;  map archive = no
```

The **map hidden** directive can control whether DOS hidden files are mapped to the local file world execute bit:

```
;  map hidden = no
```

The **map read only** directive, also known as **map readonly** in Samba documentation, controls how the DOS read-only attribute is mapped on Linux:

```
;  map read only = no
```

The **map system** directive, if set to yes, supports the mapping of DOS-style system files to the file group execute bit:

```
;  map system = no
```

Finally, the **store dos attributes** directive, if active, tells Samba to attempt to read DOS attributes from the filesystem-extended attributes, rather than mapping DOS attributes to permission bits. This is the default setting:

```
;       store dos attributes = yes
```

Shared Samba Directories

The second part of the main Samba configuration file, /etc/samba/smb.conf, is used to set up shared directories and printers via Samba. This section includes an analysis of the default version of the file.

In Samba, settings for shared directories are organized into *stanzas,* which are groups of commands associated with a share name. (*Stanza* doesn't seem like a technical term, but some believe that well-constructed configuration code is like good poetry.)

Shared Home Directories

The first four lines in this section define the **[homes]** share, which automatically shares the users' home directories. The **browseable = no** command prevents the share from being shown when a client searches the servers for available shares.

The "homes" section is a special one, as there is no default /homes directory. It's just a label. You don't need to supply a home directory because Samba will read the user's account record in /etc/passwd to determine the directory to be shared.

By default, this does not allow access to unknown users (**guest ok = no**). In addition, you can limit the systems that can use this share with directives such as **hosts allow** and **hosts deny**, described earlier. The effects of the **hosts allow** and **hosts deny** directives are limited to the share stanza where they are used.

```
#============================ Share Definitions =============
[homes]
   comment = Home Directories
   browseable = no
   writable = yes
```

on the job

There are a number of directives in smb.conf **that are not spelled correctly, such as browseable. In some cases, the correct spelling (browsable) also works. Even if misspelled, they are still accepted Samba variables. Check the spelling in the man page of** smb.conf **if in doubt.**

exam Watch

To allow Samba to share files from a home directory, make sure that the SELinux samba_enable_home_dirs **boolean is enabled.**

Shared Printers

The **[printers]** stanza normally works as is to allow access by all users with accounts on a computer or domain. Even though the spool directory (/var/spool/samba) is not browsable, the associated printers are browsable by their NetBIOS names. While changes are straightforward, the standard options mean that guest users aren't allowed to print, related print spools are not writable, and **printable = yes** is a prerequisite for loading associated configuration files, such as for CUPS.

```
# NOTE: If you have a BSD-style print system there is no need to
# specifically define each individual printer
[printers]
   comment = All Printers
   path = /var/spool/samba
   browseable = no
# Set guest ok = yes to allow users to connect without a password
   guest ok = no
   writable = no
   printable = yes
```

Domain Logons

The commands in the following stanza support the configuration of a **[netlogon]** share for Microsoft Windows workstations. As there are no **[netlogon]** shares, even for Samba-enabled Linux workstations, this section requires a Microsoft Windows computer to verify functionality. If you believe that you'll have access to a Microsoft Windows computer during the Red Hat exams, study this section carefully.

```
# Un-comment the following and create the netlogon directory for
# Domain Logons
; [netlogon]
;   comment = Network Logon Service
;   path = /var/lib/samba/netlogon
;   guest ok = yes
;   writable = no
;   share modes = no
```

Workstation Profiles

This next stanza configures a specific roaming profile share for Microsoft Windows workstations. As these profiles are used by Windows workstations, you're unlikely to

configure this section in a network of Linux-only computers. Make your own judgment on whether this section might apply during an RHCE exam.

```
# Un-comment the following to provide a specific roving profile
# share. The default is to use the user's home directory:
;[Profiles]
;    path = /var/lib/samba/profiles
;    browseable = no
;    guest ok = yes
```

Group Directories

The following stanza, as suggested by the comment, configures the /home/samba directory to be shared by the group named staff. You can configure this common group of users to share this directory. To configure special ownership and permissions for /home/samba, you'll need also to configure appropriate permissions. Both processes are described in Chapter 8.

```
# A publicly accessible directory that is read only, except for
# users in the "staff" group (which have write permissions):
;[public]
;   comment = Public Stuff
;   path = /home/samba
;   public = yes
;   writable = yes
;   printable = no
;   write list = +staff
```

The staff group can be labeled +staff or @staff. To set up appropriate permissions on the shared directory, you might also want to include the following directives for creating files and directories:

```
create mask = 0770
directory mask = 2770
```

Then, create a unique group name, preferably with a higher GID number, with the **groupadd** command. Make sure appropriate users are members of that Linux group, and that those users are also present in the Samba user database, as described later. In addition, the /home/samba directory, along with any files contained in that directory, normally must have the proper SELinux file context, something made possible with the following command:

```
# chcon -R -t samba_share_t /home/samba
```

Of course, you'll want to make sure such a change survives a SELinux relabel, and that can be configured for the noted directory with the following command:

```
# semanage fcontext -a -t samba_share_t '/home/samba(/.*)?'
```

Other Sample Stanzas

To learn more about Samba, it may be helpful to examine other stanzas for shared directories. The following examples were included in earlier Red Hat releases of Samba. While they're not included in the comments for Samba, they still can be included in the smb.conf configuration file, and therefore are still useful at least for learning purposes.

exam Watch

The RHCE objectives specify a requirement to "provide network shares suitable for group collaboration."

For example, the following share of the /tmp directory can provide a common location where users share downloaded files. If it's activated, all users (**public = yes**) get write access (**read only = no**) to this share.

```
# This one is useful for people to share files
;[tmp]
;   comment = Temporary file space
;   path = /tmp
;   read only = no
;   public = yes
```

This stanza configures a directory for Fred's exclusive use. It allows that user exclusive access to his home directory via Samba. A better location for the **path** would be within the /home directory.

```
# A private directory, usable only by fred. Note that fred
# requires write access to the directory.
;[fredsdir]
;   comment = Fred's Service
;   path = /usr/somewhere/private
;   valid users = fred
;   public = no
;   writable = yes
;   printable = no
```

on the Job

Some parameters in smb.conf **are inverted synonyms, such as** writable = yes **and** read only = no**. You can specify either one form or the other because they have exactly the same effect.**

The following stanza is slightly different from the [**tmp**] share. The only user that is allowed to connect is a guest, without authentication. Unless you've configured a guest user in Samba, this defaults to the user named nobody.

```
# A publicly accessible directory, read/write to all users. Note
# that all files created in the directory by users will be owned
# by the default guest user, so any user with access can delete
# other user's files. Obviously this directory must be writable
# by the default user. Another user could of course be specified,
# in which case all files would be owned by that user instead.
;[public]
;   path = /usr/somewhere/else/public
;   public = yes
;   only guest = yes
;   writable = yes
;   printable = no
```

Finally, this is another variation on the User Private Group scheme, which creates a group directory. Unlike the [**public**] stanza, this share is private.

```
# The following two entries demonstrate how to share a directory so
# that two users can place files there that will be owned by the
# specific users. In this setup, the directory should be writable
# by both users and should have the sticky bit set on it to prevent
# abuse. Obviously this could be extended to as many users as
required.
;[myshare]
;   comment = Mary's and Fred's stuff
;   path = /usr/somewhere/shared
;   valid users = mary fred
;   public = no
;   writable = yes
;   printable = no
;   create mask = 0765
```

Let Samba Join a Domain

If you've configured a Samba server and it's not the DC for the network, you may need to configure it as a member of the domain. To do so, you can configure an account on the DC for the network. As long as there's one domain on this network, you can join a domain with the following command:

```
# net rpc join -U Administrator
```

If there is more than one domain available, substitute the name of the controller for *DC* in the **net rpc join -S *DC* -U root** command. This assumes that the user named Administrator

is the administrative user on the DC. If the command is successful, it prompts for that user's password on the remote DC. The result adds an account for the local computer to the DC's user database.

The Samba User Database

You could set up identical usernames and passwords for both the Microsoft Windows and Samba-enabled Linux computers on a network. However, this is not always possible, especially when there are preexisting databases. In that case, you can set up a database of Samba users and passwords that correspond to current Microsoft usernames and passwords on your network.

The quickest way to set up Samba users is with the **smbpasswd** command, which is provided by the **samba-client** RPM package. Remember that you can create a new Samba user only from valid accounts on a Linux computer.

However, you can configure such an account without login privileges on the Linux system. For example, the following command adds the noted user without a valid login shell:

```
# useradd winuser1 -s /sbin/nologin
```

You can then configure that user with a Samba password with the **smbpasswd -a winuser1** command. The **smbpasswd** command is powerful; it includes a number of useful switches, as described in Table 15-4.

The location of the authentication database depends on the value of the **passdb backend** directive. If it's set to the old **smbpasswd** format, you'll find it in the /etc/samba/smbpasswd file. If it's set to **tdbsam**, you'll find it in the passwd.tdb file in the /var/lib/samba/private directory. To read the list of current users, run the following command:

```
# pdbedit -L
```

TABLE 15-4

Various smbpasswd Command Options

smbpasswd Switch	Description
-a *username*	Adds the specified *username* to the database.
-d *username*	Disables the specified *username*, thus preventing that account from authenticating with SMB.
-e *username*	Enables the specified *username*; opposite of **-d**.
-r *computername*	Allows changes to a Windows or Samba password on a remote computer. Normally goes with **-U**.
-U *username*	Normally changes the *username* on a remote computer, if specified with the **-r** switch.
-x *username*	Deletes the specified *username* from the database.

Create a Public Share

With this information, you should now know how to create a public access share for use with the entire network. For the purpose of this example, create the /publicshare directory. The following sample stanza in the /etc/samba/smb.conf configuration file reflects a directory available to all users:

```
[PublicShare]
     comment = Shared Public Directory
     path = /publicshare
     writeable = yes
     browseable = yes
     guest ok = yes
```

But that kind of security might not be appropriate. For example, assume the following limits are desirable:

- Access to the **[PublicShare]** should be limited to users with a regular local Linux account (or a user who can log in locally based on a remote authentication database such as LDAP).
- Denied access to guest users and others.
- Access to all users in the local example.com domain.
- Denied access to all users from a suspect computer such as evil.example.com.

To make this happen, change the last directive in this stanza. As **guest ok = no** is the default, you can just erase the **guest ok = yes** directive. To provide access to all users in the given domain, add the following command:

```
hosts allow = .example.com
```

Then, to deny access to one specific computer on that network, you could add **EXCEPT**; for example, the following line specifically excludes the noted **evil.example.com** system from the list:

```
hosts allow = .example.com EXCEPT evil.example.com
```

Alternatively, if this domain is on the 192.168.122.0 network, either of the following directives supports access to all systems on that network:

```
hosts allow = 192.168.122.
hosts allow = 192.168.122.0/255.255.255.0
```

You could specifically deny access to computers with a command such as the following:

```
hosts deny = evil.example.com
```

Alternatively, you could substitute the IP addresses for evil.example.com.

You've defined the share attributes in the Samba smb.conf configuration file, but you need to modify the directory associated with the share with the following command:

```
# chmod 1777 /publicshare
```

The digit (1) in front of the 777 directory permission string is known as the "sticky bit." While 777 permissions allow read, write, and execute permissions to all users, the sticky bit restricts the ability to delete or rename a file only to the owner.

Alternatively, you can limit directory permissions to members of a group, with the SGID bit. For example, take a directory with 2770 permissions. The digit (2) in front of user read/write/execute permissions (770) ensures that all files created inherit the group ownership of the directory.

Finally, don't forget to set the appropriate SELinux context on the shared directory:

```
# chcon -R -t samba_share_t /publicshare
# semanage fcontext -a -t samba_share_t '/publicshare(/.*)?'
```

Test Changes to /etc/samba/smb.conf

Before putting a new version of the /etc/samba/smb.conf file in production, test those changes. The first step in such a test is a syntax check. For that purpose, you can use the **testparm** utility, shown in Figure 15-4. A syntax utility does not check functionality; it only checks the syntax of the configuration file.

The directives shown in the output are share stanzas, along with associated directives. In this output, the **[homes]** share is not read-only and is not browsable by all clients.

FIGURE 15-4

Review the Samba configuration with testparm.

```
[root@server1 ~]# testparm
Load smb config files from /etc/samba/smb.conf
rlimit_max: increasing rlimit_max (1024) to minimum Windows limit (16384)
Processing section "[homes]"
Processing section "[printers]"
Loaded services file OK.
Server role: ROLE_STANDALONE
Press enter to see a dump of your service definitions

[global]
        workgroup = MYGROUP
        server string = Samba Server Version %v
        log file = /var/log/samba/log.%m
        max log size = 50
        idmap config * : backend = tdb
        cups options = raw

[homes]
        comment = Home Directories
        read only = No
        browseable = No

[printers]
        comment = All Printers
        path = /var/spool/samba
        printable = Yes
        print ok = Yes
        browseable = No
[root@server1 ~]#
```

EXERCISE 15-1

Configure a Samba Home Directory Share

In this exercise, you'll learn about the basic home directory share. You'll need at least two computers, one of which should be a Samba server. The other can be a Linux or Microsoft Windows client. You'll connect to the Samba server from the client and access the files in your home directory on the Samba server. These steps assume that the user account is michael; substitute your regular user account name as appropriate.

1. Install and configure Samba to start at boot using the methods described earlier in this chapter.
2. Open the /etc/samba/smb.conf configuration file. Look for the current value of **workgroup**.
3. Make sure that the computers on the local network have the same value for **workgroup**. If the local network is a Windows-style domain, set **workgroup** to the name of the domain.
4. Run the **testparm** command.
5. Read and address any problems that appear in the output from the **testparm** command. Fix any smb.conf syntax problems defined in the output.
6. Activate the samba_enable_home_dirs boolean on the Samba server with the following command:

   ```
   # setsebool -P samba_enable_home_dirs on
   ```

7. Set up a user account on the Samba server in the authentication database with the following command. Note that a corresponding account must already exist in /etc/passwd; otherwise, the command will fail (enter an appropriate password when prompted):

   ```
   # smbpasswd -a michael
   ```

8. Start the Samba services with the following command:

   ```
   # systemctl start smb nmb
   ```

   Alternatively, if Samba is already running, reload the configuration in the smb.conf file with the following command:

   ```
   # systemctl reload smb nmb
   ```

9. Open appropriate ports in the firewall for the Samba server:

   ```
   # firewall-cmd --permanent --add-service=samba
   # firewall-cmd --reload
   ```

On clients, the following commands open the required ports in firewalld:

```
# firewall-cmd --permanent --add-service=samba-client
# firewall-cmd --reload
```

10. Go to a remote Linux or Microsoft Windows workstation on the same domain or workgroup. On a Linux machine, ensure that the **samba-client** RPM is installed.
11. If you can browse the list of computers from the Samba server with the following command, browsing is working. Substitute the name of the configured Samba server host for *sambaserver.*

    ```
    # smbclient -L sambaserver -U michael
    ```

12. Log in as the root user on the remote RHEL 7 client. Ensure that the **cifs-utils** RPM is installed.
13. From that remote RHEL 7 client, use the **mount.cifs** command to configure the remote **[homes]** directory share on an empty local directory. For example, as the root user, you could mount on the local /share directory (create it if required) with the following command:

    ```
    # mount //sambaserver/michael /share -o username=michael
    ```

14. Test the result. Can you browse the home directory on the remote computer?
15. Bonus: disable the **samba_enable_home_dirs** boolean and try again. What happens?

CERTIFICATION OBJECTIVE 15.02

Samba as a Client

You can configure two types of clients through Samba. One is an FTP-like client that can browse and access directories and printers, shared from Microsoft Windows servers or Samba servers on Linux/Unix. The second adds support to the **mount** command for the SMB and CIFS protocols. The Samba client commands are available from the **samba-client** and **cifs-utils** RPMs.

Command-Line Tools

To browse shared directories from a Linux computer, you should know how to use the **smbclient** command. This can test connectivity to any SMB host on a Windows- or

Samba-based Linux/Unix computer. Assuming it's allowed through a firewall, you can use **smbclient** to check the shared directories and printers from other systems on at least the local network. For example, the following **smbclient** command checks shared directories and printers:

```
# smbclient -L server1.example.com -U donna
```

We've specified two arguments with the **smbclient** command: the **-L** argument tells the name of the Samba server, and the **-U** argument specifies a username on the remote computer. When the command reaches the Samba server, you're prompted for the appropriate password.

A list of shares will appear; for example, the following output reveals shares named public and donna, as well as a printer named OfficePrinter on a remote system named Maui:

```
Domain=[MYGROUP] OS=[Unix] Server=[Samba 4.1.1]
      Sharename       Type      Comment
      ---------       ----      -------
      public          Disk      Public Stuff
      IPC$            IPC       IPC Service (Samba Server Version 4.1.1)
      OfficePrinter   Printer   Printer in the office
      donna           Disk      Home Directories
```

From the displayed output, there's a share available named **public**. You can also use the **smbclient** command to browse and copy files, in a similar way to an FTP client. The required command to make a connection is shown next:

```
$ smbclient //server1.example.com/public -U michael
```

Of course, most administrators would prefer to mount that share on a local directory. That's where options to the **mount** command are helpful.

Mount Options

Shares can be mounted by the root administrative user. The standard is with the **mount.cifs** command, functionally equivalent to the **mount -t cifs** command. For example, the following command mounts the share named "public" on the local /home/shared directory:

```
# mount.cifs //server1.example.com/public /home/shared -o username=donna
```

This command prompts for user donna's password on the remote server. That password should be part of the Samba user authentication database on the server1.example.com system, normally different from the standard Linux authentication database.

While there is no longer an **umount.cifs** command for shared Samba directories, you can still use the **umount** command to unmount such directories.

Automated Samba Mounts

As it takes a few extra steps to set up a shared directory, you may want to automate the process. One way to automate mounts is through the /etc/fstab configuration file discussed in Chapter 6. To review the essential elements of that chapter, you could set up the public share in /etc/fstab by adding the following line (which can be wrapped in that file):

```
//server1.example.com/public /home/shared cifs username=donna,password=pass,↵
 0 0
```

However, that can be a risk, as the /etc/fstab file is world-readable. To that end, you can configure a dedicated credentials file with the username and password, as follows:

```
//server1.example.com/public /home/share cifs credentials=/etc/smbdonna 0 0
```

As suggested in Chapter 6, you can then set up the username and password in a file with a name such as /etc/smbdonna:

```
username=donna
password=donnaspassword
```

While the contents of that file must still exist in clear text, you can configure the /etc/smbdonna file as readable only by the root administrative user. It's also possible to configure the automounter with similar information. However, because the automounter is an RHCSA skill, you'll have to refer to Chapter 6 for that information.

EXERCISE 15-2

Configuring a Samba Share for Group Collaboration

In this exercise, you'll configure Samba to share a directory for collaboration so that only users in the "editors" group have write access to the share.

1. Create the Samba-only users michael and alex, and add them to the "editors" group:

```
# groupadd editors
# useradd -s /sbin/nologin -G editors michael
# useradd -s /sbin/nologin -G editors alex
# smbpasswd -a michael
# smbpasswd -a alex
```

2. Create the /editors directory and assign appropriate ownership and permissions for collaboration (2770):

```
# mkdir /editors
# chgrp editors /editors
# chmod 2770 /editors
```

3. Make sure to set the appropriate SELinux context for the directory with the following command:

```
# chcon -R -t samba_share_t /editors
```

   In addition, to make sure the changes survive a relabel of SELinux, you'll want to update the SELinux policy with a command such as the following:

```
# semanage fcontext -a -t samba_share_t '/editors(/.*)?'
```

4. Open the /etc/samba/smb.conf file in a text editor.
5. Configure Samba to share the /editors directory to users who belong to the "editors" group. In the Share Definitions section, you could add the following commands:

```
[Editors]
    comment = shared directory for McGraw-Hill editors
    path = /editors
    valid users = @editors
    writable = yes
```

6. Write and save changes to the smb.conf file.
7. Ensure that the firewall grants access to the Samba service. If you don't remember how to allow this traffic through the firewall, review Exercise 15-1.

```
# firewall-cmd --list-all
```

8. You can see if Samba is already running with the **systemctl status smb nmb** command. If it's stopped, you can start it with the **systemctl start smb nmb** command. If it's running, you can make Samba reread your configuration file with the following command:

```
# systemctl reload smb nmb
```

   This option allows you to change your Samba configuration without disconnecting users from the Samba server.
9. On a different machine, browse the share using the smbclient and michael's credentials:

```
# smbclient //server1.example.com/Editors -U michael
```

10. Mount the share to a directory, using alex's credentials, and verify that you can create a file:

```
# mkdir /mnt/alex
# mount -t cifs -o username=alex //server1/Editors /mnt/alex
# touch /mnt/alex/testfile
```

11. As a bonus, create another Samba user named "evil." Try remounting the share as user evil. What happens?

Multiuser Samba Mounts

When mounting a Samba share via the **mount.cifs** command as root, you need to specify the credentials of a valid Samba account that has access permissions to the share. Those credentials are required to validate the mount, as well as file and directory permissions. In a multiuser environment, this solution is not ideal. That's because whenever a file is created on the Samba share, that file is owned by the user specified with the **mount** command.

To address this problem, the **mount.cifs** command includes a **multiuser** option. As an example, you can modify the **mount** command from Exercise 15-2 as shown:

```
# mount.cifs -o multiuser,username=alex //server1/Editors /mnt/editors
```

You can see how this works when you go through Lab 6 at the end of the chapter. Unless Samba is configured with Kerberos authentication (which is outside the scope of the RHCE exam), users must enter their SMB credentials to access the mounted share. The **cifscreds** tool, included with the **cifs-utils** RPM package, serves this purpose. The command adds a user's credentials to the kernel keyring, as shown here:

```
$ cifscreds add server1.example.com
```

Credentials are only stored for the current user's session. When a new session is established, they must be reentered by running the preceding command again.

CERTIFICATION OBJECTIVE 15.03

Samba Troubleshooting

Samba is complex. With a complex service, simple mistakes may be difficult to diagnose. Fortunately, Samba includes excellent tools for troubleshooting. The basic **testparm** command tests syntax, but log files can tell you more. Of course, unless appropriate changes are made in local firewalls, Samba might not even be accessible from remote systems.

Samba Problem Identification

Samba is a forgiving service. It includes synonyms for a number of parameters. But beyond those parameters, the **testparm** command, which we have already discussed, can help identify problems. For example, Figure 15-5 illustrates a number of problems. Unrecognized parameters are highlighted with the "unknown parameter" message.

Some parameters don't work with each other. For example, the following message in the **testparm** output highlights two incompatible directives:

```
ERROR: both 'wins support = true' and 'wins server = <server list>' cannot
be set in the smb.conf file. nmbd will abort with this setting.
```

FIGURE 15-5

Some syntax problems identified by testparm

```
[root@server1 ~]# testparm
Load smb config files from /etc/samba/smb.conf
rlimit_max: increasing rlimit_max (1024) to minimum Windows limit (16384)
Unknown parameter encountered: "ecurity"
Ignoring unknown parameter "ecurity"
Unknown parameter encountered: "assdb backend"
Ignoring unknown parameter "assdb backend"
Processing section "[homes]"
Processing section "[printers]"
Processing section "[public]"
Unknown parameter encountered: "rite list"
Ignoring unknown parameter "rite list"
Loaded services file OK.
ERROR: both 'wins support = true' and 'wins server = <server list>' cannot be se
t in the smb.conf file. nmbd will abort with this setting.
Server role: ROLE_STANDALONE
Press enter to see a dump of your service definitions
```

Sometimes, troubleshooting commands come in the output to other commands. For example, often the output message is straightforward, such as the following output to a specific **mount.cifs** command, associated with an incorrect share name. It also suggests that the case of share names is less important on networks associated with Microsoft operating systems.

```
Retrying with upper case share name
```

Sometimes, messages may appear to be more straightforward, such as

```
mount error(13): Permission denied
```

But that message could refer to an incorrect password or a user who has not been configured in the Samba database.

Sometimes problems may seem more annoying. For example, if you mount a remote home directory and no files show up in that directory, it could mean that the SELinux **samba_enable_home_dirs** boolean has not been enabled. If you mount a remote directory other than a user home directory, it could mean that the directory and associated files are not properly labeled with the **samba_share_t** file context type.

Local Log File Checks

Problems associated with Samba may appear in the /var/log/messages file, or they may appear in different files in the /var/log/samba directory. First, syntax errors revealed in the output to the **testparm** command may also appear in the /var/log/messages file. As the Samba services are started, errors in the configuration file are problems worth reporting in the standard system log file.

In addition, when an attempted mount of a shared Samba directory fails, associated messages also appear in the /var/log/messages file. Sometimes the messages are straightforward, such as "cifs_mount failed" or NT_STATUS_LOGON_FAILURE.

The systemd journal is a very useful resource for searching and browsing Samba log entries. All messages specific to the **smb** and **nmb** systemd units can be displayed by the following command:

```
# journalctl -u smb -u nmb
```

Samba server log files are stored in the /var/log/samba directory. The log files are classified by the host or IP address of the client that connects to the server. By default, Samba is not very verbose. You may want to set the **log level** entry to 1 or 2 in /etc/samba/smb.conf, and look at how the detail of the log messages has changed. As an example, a connection to a localhost system, useful for troubleshooting, may include the following message in the log.127.0.0.1 log file:

```
127.0.0.1 (ipv4:127.0.0.1:44218) connect to service michael initially ↵
 as user michael (uid=1001, gid=1001) (pid 23800)
```

The connected user is identified by UID, GID, and PID numbers. If an unauthorized user connects, these numbers can help identify a problem user and/or a compromised account, along with an associated process ID number.

Most of the other files in this directory relate to various services as named; for example, the log.smbd, log.nmbd, and log.winbindd files collect messages associated with the daemons named in each respective log file.

SCENARIO & SOLUTION

You need to set up sharing on a network with Microsoft computers.	Install Samba and configure shared directories in /etc/samba/smb.conf. Make sure shared directories (except for user home directories) have the appropriate **samba_share_t** SELinux context type.
You want to set up sharing of user home directories via Samba.	Activate the **[homes]** stanza, set up appropriate users in the Samba authentication database, and turn on the **samba_enable_home_dirs** boolean.
You want to set up host-based security for Samba.	Set up appropriate **hosts allow** and **hosts deny** directives in smb.conf or configure firewalld to limit access.
You want to set up user-based security for Samba.	Set up appropriate **valid users** and/or **invalid users** directives in smb.conf.
You need to set up a share for group collaboration.	Set up a share stanza with **valid users** set to a specific group, along with **writeable = yes** and appropriate permissions for group collaboration on the shared directory. Make sure the shared directory and its content have the **samba_share_t** SELinux context type.
You have mounted a CIFS share in /etc/fstab and want users to access the content using their own Samba credentials, rather than those specified by the **mount** command.	Mount the Samba share with the **multiuser** option. Instruct users to run the **cifscreds** command and provide their SMB credentials before accessing the contents of the share.

CERTIFICATION SUMMARY

Samba allows a Linux computer to appear like any other Microsoft computer on a Microsoft Windows–based network. Samba is based on the Server Message Block protocol, which allows Microsoft computers to communicate on a TCP/IP network. It has evolved as Microsoft has adapted SMB to the Common Internet File System. Network communication to Samba works through UDP ports 137 and 138, as well as TCP ports 139 and 445. The key SELinux boolean is **samba_enable_home_dirs**. Shared directories should be set to the **samba_share_t** file context type.

The main Samba configuration file, /etc/samba/smb.conf, includes separate sections for global settings and share definitions. The **smbpasswd** command can be used to set up existing Linux users in a local Samba authentication database. The **multiuser** mount option and the **cifscreds** command authenticate with the corresponding user's credentials when accessing a share.

As for troubleshooting, changes to smb.conf can be easily tested with the **testparm** utility. Samba includes a number of synonyms for directives; some proper directives are based on spelling mistakes. While basic Samba service log messages can be found in the /var/log/messages file, most Samba log information can be found in the /var/log/samba directory. Many of the files in that directory include the client name or IP address.

TWO-MINUTE DRILL

Here are some of the key points from the certification objectives in Chapter 15.

Samba Services

- Samba allows Microsoft Windows computers to share files and printers across networks, using the Server Message Block (SMB) protocol and NetBIOS over the TCP/IP protocol stack.
- Samba includes a client and a server. Variations on the **mount -t cifs** and **mount.cifs** commands support mounting of a shared Samba folder or even joining a Microsoft domain.
- The main Samba configuration file is /etc/samba/smb.conf.
- Samba supports configuration of a Linux computer as a Microsoft Windows server. It can also provide Microsoft browsing, WINS, and Domain Controller services, even on an Active Directory network.

Samba as a Client

- The **smbclient** command can display shared directories and printers from specified remote Samba and Microsoft servers, using an FTP-like interface.
- The **mount.cifs** command can mount directories shared from a Samba or Microsoft server.
- Samba shares can be mounted during the boot process with the help of the /etc/fstab configuration file.
- The **multiuser** mount option and the **cifscreds** command regulate access and permissions to a share using the credentials of each user, rather than those specified by the **mount** command.

Samba Troubleshooting

- The **testparm** command performs a syntax check on the main Samba configuration file, /etc/samba/smb.conf.
- Logs of Samba daemons may be written to the /var/log/messages file.
- Most Samba log files can be found in the /var/log/samba directory. Different log files can be found by client and by daemon.

SELF TEST

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

Samba Services

1. An IT department that runs Microsoft servers has set up a Windows Server to handle file and print sharing services. This server correctly refers to a WINS server on 192.168.55.3 for name resolution and configures all user logins through the DC on 192.168.55.8. If you're configuring the

local Linux system as a DC, what directive, at a minimum, do you have to configure in the local Samba configuration file?

_____

2. You've recently revised the Samba configuration file and do not want to disconnect any current users. What command forces the Samba and NetBIOS services to reread the configuration file—without having to disconnect Microsoft users or restart the service?

_____

3. What ports must be open for a Samba server to work with remote systems?

_____

4. What SELinux setting is appropriate for sharing home directories over Samba?

_____

5. What SELinux file context type is appropriate for shared directories on Samba?

_____

6. What Samba directive limits access to systems on the example.org network?

_____

7. What Samba directive limits access to users tim and stephanie?

_____

8. What Samba directive limits access in a shared stanza to a configured group named ilovelinux?

_____

9. What Samba directive supports access to all users in a shared directory?

_____

10. What command adds user elizabeth to a tdbsam Samba authentication database?

_____

Samba as a Client

11. What command can be used to mount remotely shared Microsoft directories?

_____

Samba Troubleshooting

12. You made a couple of quick changes to a Samba configuration file and need to test it quickly for syntax errors. What command tests smb.conf for syntax errors?

_____

LAB QUESTIONS

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

Red Hat presents its exams electronically. For that reason, the labs in this and future chapters are available from the media that accompanies the book, in the Chapter15/ subdirectory. In case you haven't yet set up RHEL 7 on a system, refer to Chapter 1 for installation instructions.

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

A SELF TEST ANSWERS

Samba Services

1. At a minimum, to configure a Linux system as a DC, you need to change the **security = user** directive. If it's on an Active Directory system, you should use the **security = ads** directive.
2. The command that forces the Samba and NetBIOS services to reread the configuration file—without disconnecting Microsoft users or restarting the service—is **systemctl reload smb nmb**.
3. Open ports associated with communication to a Samba server are UDP ports 137 and 138, as well as TCP 139 and 445.
4. The SELinux boolean associated with the sharing of home directories on Samba is **samba_enable_home_dirs**.
5. The SELinux file type appropriate for shared Samba directories is **samba_share_t**.
6. The Samba directive that limits access to systems on the example.org network is

```
hosts allow = .example.org
```

   The following directive is also an acceptable answer:

```
allow hosts = .example.org
```

7. One Samba directive that limits access to the noted users is

```
valid users = tim stephanie
```

8. One Samba directive that limits access to the noted group is

```
valid users = @ilovelinux
```

   The **+ilovelinux** group would also be acceptable.

9. One Samba directive that supports access to all users in a shared directory is

```
guest ok = yes
```

10. The command that adds user elizabeth to a tdbsam Samba authentication database is

```
# smbpasswd -a elizabeth
```

Samba as a Client

11. The command that can be used to mount remotely shared Microsoft directories is **mount.cifs**. The **mount -t cifs** command is also an acceptable answer.

Samba Troubleshooting

12. The command that can test a Samba configuration file for errors is **testparm**.

LAB ANSWERS

Lab 1

The chapter lab on Samba is designed to be easy to follow. However, you'll need explicit Linux knowledge to complete several steps.

1. You've installed the "File and Print Server" environment group, or as an alternative the "File and Storage Server" group, which includes one Samba-related RPM, samba. Dependent packages, such as samba-common and samba-libs, will also be installed.
2. One way to find all related Samba packages is with the **yum search samba** command. You can then install noted packages with the **yum install *packagename*** command.
3. To support communications to a local Samba server, run the following:

```
# firewall-cmd --permanent --add-service=samba
# firewall-cmd --reload
```

On clients, run this:

```
# firewall-cmd --permanent --add-service=samba-client
# firewall-cmd --reload
```

4. You can use the **systemctl enable smb nmb** command to make sure that Samba and NetBIOS start the next time you boot Linux.
5. Use the **systemctl start smb nmb** command to begin the Samba and NetBIOS services.
6. One way to verify that Samba is running is to look for the existence of the **smbd** and **nmbd** processes in the process table. Use **ps aux | grep mbd** to see if these processes are present. Another way is with a systemd command such as the **systemctl status smb nmb** command.

Lab 2

This lab should familiarize you with the available documentation for the Samba file server. When you run the **man smb.conf** command, it will open the manual for the main Samba configuration file. You should be able to search through the file with vi-style commands. For example, to search for the **hosts allow** directive, going forward in the file, type in

```
/hosts allow
```

and press N to see the next instance of that directive. Alternatively, to search backward, type in

```
?hosts allow
```

and press N to see the previous instance of that directive in the man page.

From the command line, browse around other Samba man pages. Learn what you need as a reference for the job, or for an exam.

Lab 3

This lab assumes that you've backed up the smb.conf file from the /etc/samba directory.

1. Some administrators stick with the standard Microsoft Windows **workgroup** name of **WORKGROUP**. You can find it in the output from the **smbclient -L //*clientname*** command.
2. To limit access to a Samba server, you can do so in the global section, with the **hosts allow** directive.
3. To limit access from a specific computer, you can do so in the global section, with the **hosts deny** directive.
4. You can make Samba read the changes with the **systemctl reload smb** command. Before committing the changes, you can test them with the **testparm** command.

5. When testing the connection from another system, use the **smbclient** command. You'll need to allow access through UDP ports 137 and 138 for that purpose, something possible by enabling the samba-client service with the Firewall Configuration tool.
6. If you need a fresh version of the smb.conf file, delete or move the existing version of the file from the /etc/samba directory and run the **yum reinstall samba-common** command.

Lab 4

If successful, only one remote user will get access to his home directory via Samba, something that you can test with appropriate **smbclient** and **mount.cifs** commands. One way to implement the requirements of this lab is with the following steps.

1. Open the main Samba configuration file, /etc/samba/smb.conf, in a text editor.
2. Navigate to the **[homes]** share in the last part of this file.
3. Unless there is already an appropriate limitation in the **[global]** section in this file, you can limit the **[homes]** share with the **hosts allow = .example.com**.
4. Add a **valid users =** ***username*** directive with the name of the desired user in the **[homes]** stanza.
5. Add the desired user to the Samba authentication database with the **smbpasswd -a** ***username*** command.
6. Restart or reload the Samba service with the appropriate **systemctl** command.
7. Save the changes made so far.
8. Test the result from a remote system with the **smbclient** command. You should also be able to use the **mount.cifs** command from a client root account, with the **-o username=***username* switch, to mount the shared user home directory.

Lab 5

This lab can be a continuation of Lab 4. You're just adding another stanza to the main Samba configuration file.

1. At the end of the file, start a **[public]** stanza. Add an appropriate comment for the stanza.
2. Set **path = /home/public**.
3. Make sure to set **hosts allow = .example.com**. Save your changes to the smb.conf file.
4. Set permissions for the public share with the following commands:

```
# mkdir /home/public
# chmod 1777 /home/public
```

The 777 setting for permissions grant read, write, and execute/search permissions to all users (root, root's group, and everyone else). The 1 at the beginning of the permission value sets the sticky bit. This bit, when set on directories, keeps users from deleting or renaming files they don't own.

5. Commit the changes to the currently running Samba service with the **systemctl reload smb** command.
6. When testing the result from a remote system, any username in the local Samba database should work.

Lab 6

This lab may take a significant amount of work. You'll need to set up a group of users, with group ownership of a dedicated directory. Because that discussion in Chapter 8 was based on an RHCSA requirement, you may have to repeat that process in this lab.

Set up a user account for mounting the Samba share:

```
# useradd -M -s /sbin/nologin sambashare
# smbpasswd -a sambashare
```

Add the credentials of this user account in a file (for example, /etc/sambashare). The content of the file should include the following directives:

```
username=sambashare
password=sambasharepssword
```

Then, add a line like the following to /etc/fstab:

```
//server1.example.com/public /home/share cifs multiuser,↵
credentials=/etc/sambashare 0 0
```

Note that the mount options include the **multiuser** directive. Once the filesystem is mounted, users must run the following command to store their credentials into the system keyring and access the contents of the share:

```
$ cifscreds add server1.example.com
```

Lab 7

It's important to make sure that the configured service actually runs after a reboot. In fact, it's best to make sure the configured service works after a SELinux relabel, but that process can take several minutes or more. Also, it's quite possible that you won't have that kind of time during an exam.

1. To complete many Linux configuration changes, you need to make sure that the service will start automatically when you reboot your computer. In general, the key command is **systemctl**. In this case, the **systemctl enable smb nmb** command sets up the **smbd** and **nmbd** daemons to become active when you boot Linux in the default target.

2. You can use various commands to perform an orderly shutdown, such as **shutdown**, **halt**, and more.
3. After the reboot, you should verify at least one appropriate change to the Samba SELinux settings with the following command:

```
# getsebool samba_enable_home_dirs
```

4. In addition, you should confirm appropriate directories are configured with the samba_share_t file context type, not only with the **ls -Z** command in the noted directories, but also in the file_contexts.local file, in the /etc/selinux/targeted/contexts/files directory.