15

The Samba File Server

CERTIFICATION OBJECTIVES

Samba is the Linux implementation of the networking protocols used to connect Microsoft operating systems. Microsoft networking 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, and with Samba on Linux 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. In its optional repository, RHEL 6 includes a GUI configuration tool—the Samba Web Administration Tool.

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

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 is seamless with Microsoft clients. But as you won't 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.

Of course, you can't 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. While Samba does not include every feature built into the latest Microsoft networks, I have confidence that it will in the near future.

on the job

I look forward to the final release of Samba 4.0, which will make it possible for Linux to act as an AD controller on a Microsoft-based network. RHEL 6 includes a preliminary version of Samba 4.0 and may include it in the lifetime of RHEL 6.

SMB network communication over a Microsoft-based network is also known as the Network Basic Input/Output System (NetBIOS) over TCP/IP. Through the collective works of Andrew Tridgell and the Samba team, Linux systems provide transparent and reliable SMB support over TCP/IP via a package known as Samba.

Samba emulates many of the advanced network features and functions associated with various Microsoft operating systems through the SMB protocol. Complete information can be found at the official Samba web site at www.samba.org. 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.
- Share user home directories.
- Act as a Windows Internet Name Service (WINS) client or server.
- Link to or manage a workgroup browse service.
- Act as a master browser.

- Provide user/password and share security databases locally, from another Samba server or from a Microsoft NT 4 PDC.
- Configure local directories as shared SMB filesystems.
- Synchronize passwords between Windows and Linux systems.
- Support Microsoft Access Control Lists.

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. As this file may intimidate some users, the Samba Web Administration Tool (SWAT) provides a GUI interface.

exam Watch

Study the /etc/samba/smb.conf configuration file. It includes many useful comments and suggested directives.

If you use SWAT, back up the Samba configuration file first, as it overwrites the default comments and directives.

Install Samba Services

The installation of Samba services and packages is somewhat different from other servers. Samba packages are not organized in a single package group. While there is a "CIFS file server" package group, that group includes only the samba RPM package. Although that's the only package required to set up a Samba server, you may find other Samba packages of use. Important Samba packages are described in Table 15-1.

Some Samba Background

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

The original Microsoft Windows networks were configured with computer host names, known as NetBIOS names, limited to 15 characters. These unique host names provided a simple, flat host name system for the computers on a LAN. All computer identification requests were made through broadcasts. This overall network transport system is known as the NetBIOS Extended User Interface (NetBEUI), which is not "routable." In other words, it does not allow communication between two different LANs. As a result, the original Microsoft-based PC networks were limited in size to 255 nodes.

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 shares from Samba and Microsoft servers.
samba-common	Contains common Samba commands used by both the client and the server.
samba-doc	Includes Samba documentation in both HTML and PDF formats.
samba-domainjoin-gui	Supports connections to network workgroups and domains.
samba-swat	Provides the web-based interface for Samba configuration.
samba-winbind	Supports Samba as a member server on Microsoft-based domains and supports Windows users on Linux servers.
samba-winbind-nss	Provides client connections to Winbind via PAM and the Network Switching Service (NSS).

While Microsoft networks could have used the Novell IPX/SPX protocol stack to route messages between networks, that was not good enough. As the Internet grew, so did the dominance of 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 SMB. As Microsoft has moved toward CIFS, Samba developers have adapted well. But some fairly recent changes have affected the configuration file as well as the main command line client **mount** command.

One of the nice 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 network. For instance, if the NetBEUI, IPX/SPX, and TCP/IP protocols were installed on a host, then three duplicate browse databases were required—one per protocol—as the services available may differ between protocols.

Ports, Firewalls, and Samba

Samba as a service and a client requires access through multiple network protocols. When communications with both Samba clients and servers is enabled through the

Red Hat Firewall Configuration tool, it adds the following rules to the /etc/sysconfig/iptables configuration file:

```
-A INPUT -m state --state NEW -m udp -p udp --dport 137 -j ACCEPT
-A INPUT -m state --state NEW -m udp -p udp --dport 138 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 139 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 445 -j ACCEPT
```

In other words, several services are involved, as described in Table 15-2. You'll note that three of the services use the User Datagram Protocol (UDP), a connectionless protocol. Collectively, the three associated ports specify NetBIOS communication over TCP/IP (NBT).

For Samba client systems, only ports 137 and 138 need to be opened.

Configure SELinux Booleans for Samba

There are several directives associated with making a Samba server work with SELinux in targeted mode, as described in Table 15-3. Only one of these booleans (qemu_use_cifs) is enabled by default. However, you may have to activate a number of these booleans to support different Samba functions.

For some readers, this may be getting repetitive. However, SELinux is not well understood even by many Linux experts. So for example, 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.

There are cases where it's appropriate to enable the samba_export_all_ro or samba_export_all_rw booleans, such as on directories that are shared through other servers. For example, files that are shared via an Apache web server must be labeled with the httpd_sys_content_t file type.

TABLE 15-2 Samba Communication Services

Port/Protocol	Description
137/UDP	NetBIOS name service
138/UDP	NetBIOS datagram service
139/UDP	NetBIOS session service
445/TCP	Microsoft directory services, also known as Samba over IP

TABLE 15-3 Samba Communication Services

Boolean	Description
allow_smb_anon_write	Supports the writing of files to directories configured with the **public_content_rw_t** SELinux setting.
cdrecord_read_content	Allows the **cdrecord** command to read shared Samba (and other network) directories.
qemu_use_cifs	Works with access to CIFS filesystems; enabled by default.
samba_create_home_dirs	Supports the creation of home directories, normally set up for external users.
samba_domain_controller	Allows Samba to act as a domain controller for authentication management.
samba_enable_home_dirs	Enables the sharing of 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_run_unconfined	Supports the execution of unconfined scripts from the /var/lib/samba/scripts directory.
samba_share_fusefs	Allows Samba to share filesystems mounted to fusefs, a common mount for the Microsoft NTFS filesystem.
samba_share_nfs	Enables sharing of NFS filesystems.
use_samba_home_dirs	Supports the use of a remote server for Samba home directories.
virt_use_samba	Allows a VM to access files mounted to the CIFS filesystem.

Configure SELinux File Types for Samba

Normally, Samba can only share those files and directories labeled with the samba_share_t file type. It is true, the samba_share_t file type is not required if the samba_export_all_ro or samba_export_all_rw booleans are enabled. However, that would be a security risk. So in most cases, you'll want to enable directories (and files therein) with the noted file type with a command like the following:

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

In addition, to make sure the changes survive a relabel of SELinux, you'll want to set up the file_contexts.local file in /etc/selinux/targeted/contexts/files directory 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 a substantial number of commands that run the service, as well as aid in configuration. The most important of the 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, most administrators will want to run the Winbind service (**winbindd**) for user and host name resolution. All three 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, the associated scripts in the /etc/init.d directory are **smb**, **nmb**, and **winbind**. They start the associated **smbd**, **nmbd**, and **winbindd** daemons with the following options in the /etc/sysconfig/samba file:

```
SMBDOPTIONS="-D"
NMBDOPTIONS="-D"
WINBINDOPTIONS=""
```

Yes, while no options are included for the **winbind** daemon, they can be included in quotes in the noted file. To confirm the way a daemon is running, the **ps** command can help. For example, the following output to the **ps aux | grep smb** command confirms that the Samba service is running with the **-D** switch:

```
root  12836  0.0  0.2 203612  1648 ?  S  Mar08 0:00 smbd -D
```

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

Unlike with some other services, the default Samba configuration file includes a number of commented directives other than the default. The default value of such directives can be found in the man page for the smb.conf file.

You can edit this file directly or create directory shares using SWAT. Before using any GUI tool, be brave. Study the original /etc/samba/smb.conf file. Once you see how the file is structured, back it up. Try editing the file directly. Try changing the file with the SWAT tool, described later in this chapter. Test the result by restarting the Samba server with the following command:

```
# service smb restart
```

To help you with this process, I'll analyze the default RHEL 6 version of this file. The code shown next is essentially a complete view of this file. In some cases, I've replaced the comments in the file with my 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 for 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 the disabled feature).

(Note that the physical dimensions of this book limit the lengths of lines of code. In a few cases, I'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. You should read the
# smb.conf(5) manual page in order to understand the options listed
# here. Samba has a huge number of configurable options (perhaps
# too many!) most of which are not shown in this example.

# NOTE: Whenever you modify this file you should run the command
# "testparm" to check that you have not made any basic syntactic
# 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.

In smb.conf, the global settings, which define the overall attributes of a server, follow the first set of comments, including SELinux-related comments covered earlier. The [global] 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 --------------
```

(In early releases of RHEL 6, the word "Network" is actually misspelled as "Netwrok.") With that in mind, look at each of the directives in this part of the Global Settings section. Despite the name, the **workgroup** variable specifies the name of a workgroup or more commonly, a domain. But since peer-to-peer workgroups were developed first, the default Samba **workgroup** is **WORKGROUP**, which happens to be the old name of the default peer-to-peer workgroup. It's now set to the default workgroup for Microsoft Windows 7:

```
    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 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 regular Linux **smbclient** command.

```
;   netbios name = MYSERVER
```

If the local system is connected to more than one network, you can specify them 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 networks with the 192.168.12.0 and 192.168.13.0 network IP addresses, 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.

Logging Options

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

```
#----------- 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.

```
# logs split per machine
log file = /var/log/samba/%m.log
# max 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, to configure this computer as a member server on a domain, use a password database from a DC. Strangely enough, in that case, you would substitute the following command:

```
security = domain
```

on the job

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.

To configure a system as a member server on an Active Directory network, substitute the following command:

```
security = ads
```

Alternatively, to use a database from another computer that is not a DC, you'd substitute the following command:

```
security = server
```

Finally, to configure a system on a peer-to-peer workgroup that does not require usernames, substitute the following command:

```
security = share
```

To summarize, there are five basic authentication options: **share**, **user**, **server**, **domain**, and **ads**.

Now, refocus this directive on the authentication database. The default is **security = user**; in this case, make sure the Samba usernames and passwords that you create match those on individual Windows NT/2000/XP/Vista systems on the network. If the database is local, it could be either

```
passdb backend = smbpasswd
```

or

```
passdb backend = tdbsam
```

The smbpasswd database is local, stored in the local /etc/samba directory. The tdbsam option, short for the Trivial Database Security Accounts Manager, sets up a local account database in the /var/lib/samba 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 = server** or **security = domain**, you'll also want to activate the following directive with the name or IP address of the password server. Alternatively, you could replace <NT-Server-Name> with a * to have Samba search for the password server.

```
;   password server = <NT-Server-Name>
```

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, as the system that receives requests for logins to the domain:

```
;  domain master = yes
;  domain logins = yes
```

The next command set up Microsoft command line batch files by computer and user. The command afterward stores Microsoft user profiles on the local Samba server. That means these commands can't be tested on the Red Hat exams unless you have access to a Microsoft Windows computer. Since I can't tell you what's on the Red Hat exams, I can only suggest that Red Hat might not want separate Microsoft Windows computers available during their exams. Of course, Microsoft

Windows guest VMs are included in the description for the Red Hat Enterprise Virtualization course.

```
# the login script name depends on the machine name
;   logon script = %m.bat
# the login script name depends on the unix user used
;   logon script = %U.bat
;   logon path = \\%L\Profiles\%U
```

The remaining commands 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 --------------
```

Unless a Samba server is specifically designated as a local browse master,

```
; local master = no
```

Samba participates in browser elections like any other Microsoft Windows computer, using the specified **os level**.

```
; os level = 33
```

Alternatively, if a Domain Controller isn't already elected as a browse master, you can make it easier for the local computer to win the browser election, with the **preferred master** command:

```
; 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, starting with the following comment:

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

The Windows Internet Name Service (WINS) is functionally equivalent to DNS on Microsoft-based networks such as Samba. If you activate the following command, Samba activates 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 incompatible.

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

Samba servers may not installed on every Linux system. In that case, you could enable the following directive to allow access from such systems with only Samba client software:

```
;   wins proxy = yes
```

If the answer to a name resolution request is not in a WINS server, the following directive would allow the same search through 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. Nevertheless, printing is part of the default Samba server configuration. So you should at least scan the 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 loads printers as defined by **printcap name = /etc/printcap**.

The **cups options = raw** directive means that print jobs are already processed by a service with print processors, such as the CUPS service.

```
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 older Linux systems:

```
printcap name = lpstat
```

Filesystem Options

The following section supports the configuration of extended attributes, associated with the Access Control List (ACL) settings for a Microsoft file. With the right options, such attributes can be stored. The comments within the Samba configuration file refer to the Microsoft Disk Operating System (DOS), as shorthand for how permissions and related ACL bits are specified for such shared files. The following would be the default for all shared directories, starting with the following comment:

```
#----------- Filesystem Options --------------
```

First, the **map archive** directive can control whether the DOS file archive attribute is mapped to the local Linux executable 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 Linux executable bit, if supported by the **create mask** directive.

```
;  map hidden = no
```

The **map read only** directive, also known as **map readonly** in Samba documentation, as shown is useful for shared mounted media such as DVDs:

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

The **map system** directive, if set to yes, supports the use of the Linux execute bit for DOS system files:

```
;  map system = no
```

Finally, the **store dos attributes** directive, if active, attempts to store previously configured ACLs of DOS files:

```
;       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 home directory of the logged-in user. Every user gets access to his or her own home directory; the **browseable = no** command keeps users away from each other's home directory.

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 variables 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 and generally should be spelled per the Samba defaults, not standard written English.***

exam Watch

Before a shared home directory can actually share files over Samba, the SELinux samba_enable_home_dirs boolean must be active.

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 = /usr/spool/samba
   browseable = no
# Set public = yes to allow user 'guest account' to print
   guest ok = no
   writable = no
   printable = yes
```

Domain Logons

The commands in the following stanza supports 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 profiles for Microsoft Windows workstations. As these profiles become a part of a Microsoft Windows registry when you log on to one of those 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, but read only, except for
# people in the "staff" group
;[public]
;   comment = Public Stuff
;   path = /home/samba
;   public = yes
;   writable = yes
;   printable = no
;   write list = +staff
```

To set up appropriate permissions on the shared directory, you may also want to include the following directives for creating files and directories:

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

One difference with Samba is that Microsoft authentication databases do not allow users and groups to have the same names. In addition, the /home/samba directory, along with any files contained in that directory, normally must have the proper SELinux file type, 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
```

exam Watch

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

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.

For example, the following share of the /tmp directory can share a common location where users share downloaded files. If 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
```

The following stanza is slightly different from the **[tmp]** share. Once connected, the only user that connects is a guest. Unless you've configured a guest user, 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 user, so any user with access can delete any
# 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 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, it's easy to do with the following command:

```
# net rpc join -U root
```

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 root is the administrative user on the DC. However, the administrative user on a domain governed by a Microsoft Windows computer is *administrator*. If the command 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 in /etc/passwd.

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. A template is available in /etc/samba/smbusers and is in effect if you add the following entry to the smb.conf file:

```
username map = /etc/samba/smbusers
```

If you're comfortable with the command line interface, the quickest way to set up Samba users is with the **smbpasswd** command. 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 described in Table 15-4.

TABLE 15-4 Various smbpasswd Commands

smbpasswd Switch	Description
-a *username*	Adds the specified *username* to the database.
-d *username*	Disables the specified *username*; thus disables that password from Microsoft networking.
-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.

The location of the authentication database depends on the value of the passdb backend directive. If it's set to smbpasswd, 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
```

To configure different usernames and passwords for Linux and Microsoft computers, you'll need to edit them directly into the /etc/samba/smbusers file; alternatively, such users can be configured with SWAT.

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 chapter, create the /home/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 = /home/PublicShare
     writeable = yes
     browseable = yes
     guest ok = yes
```

But that kind of security may 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 outsider1.example.org.

To make this happen, change the last command 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
```

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 IP addresses in the same format as with the **hosts allow** directive. 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 /home/PublicShare
```

The digit 1 in front of the 777 directory permission string is known as the "sticky bit." That sticky bit allows any user to do anything in the directory, courtesy of the 777 permission value. But such privileges are limited to files created by the specific user. Otherwise, any user could delete or rename any file in the /home/PublicShare directory, regardless of the file's owner.

Alternatively, a directory with permissions limited to members of a group may have 2770 permissions, with the SGID bit set and full permissions given to members of the group that owns the directory.

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 workstation. You'll connect to the Samba server from the workstation 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 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. Test the syntax of the Samba configuration file with 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 (enter an appropriate password when prompted):

   ```
   # smbpasswd -a michael
   ```

8. Make Samba reread the smb.conf file with the following command:

   ```
   # service smb reload
   ```

9. Go to a remote Linux or Microsoft Windows workstation on the same domain or workgroup.
10. 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
    ```

11. Log in as the root user on the remote RHEL 6 Samba client.
12. From that remote RHEL 6 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=root
    ```

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

The Samba Web Administration Tool

RHEL 6 no longer includes a dedicated Red Hat GUI tool to configure Samba. Instead, Red Hat has included the web-based administration tool created by Samba developers for that purpose, known as SWAT, which you can install from the samba-swat RPM.

SWAT is not available from the standard RHEL 6 DVD. So if that's all that's available on a Red Hat exam, you won't be able to use it to configure Samba for the RHCE. However, SWAT is such an excellent, well-documented tool, it's worth the trouble to install and activate that tool. It can help you learn more about Samba.

On a genuine RHEL 6 system, it's available from the RHEL Server Optional repository, which can be activated from a Red Hat Network account at https://rhn.redhat.com. On the Scientific Linux rebuild distribution (and possibly other rebuilds), it's available from standard repositories configured in the /etc/yum.repos.d directory.

on the job

The Red Hat Network web site at https://rhn.redhat.com is now an interface to "Classic Subscription Management." For the latest RHN interface, see the Knowledgebase article at https://access.redhat.com/kb/docs/DOC-47394.

SWAT is installed and run as an Extended Internet Super Server service in the /etc/xinetd.d directory, like those discussed in Chapter 10. Once installed, you can activate it with the following commands:

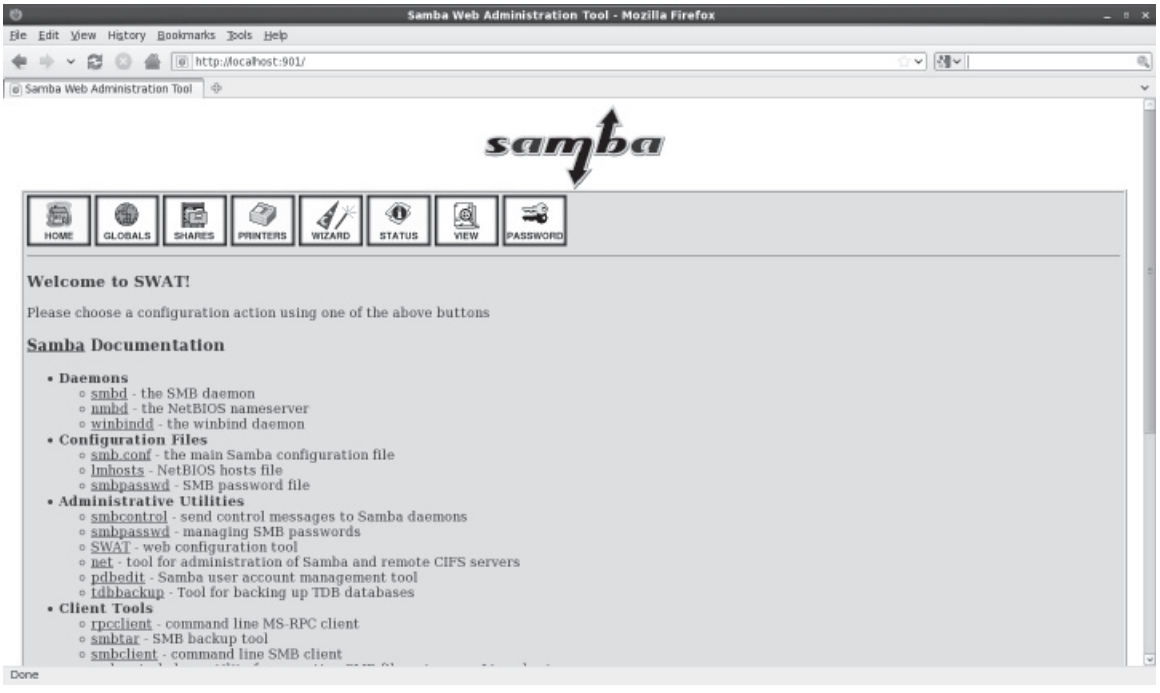
```
# chkconfig swat on
# /etc/init.d/xinetd restart
```

Next, open a web browser and navigate to http://localhost:901. You'll be prompted for a username and password; the root user account and password should work. When authentication is confirmed, you're taken to a screen similar to that shown in Figure 15-1.

If you want to access SWAT from a remote location, comment out the following directive in the /etc/xinetd.d/swat file and then open up TCP port 901 in any existing local firewall.

```
only_from = 127.0.0.1
```

FIGURE 15-1 The Samba web administration tool



Most of the hyperlinks are associated with the man page for noted commands. At the top of the menu, you should see icons for Home through Password. Figure 15-1 displays the home page for SWAT. The following sections briefly describes the options in each of the other screens.

SWAT provides a comprehensive view of what you can do with Samba. But be careful. Many of the features may be useful for a real network, especially a network mixed with Microsoft systems. However, most go beyond what's necessary for basic operation, and the Red Hat exams. Don't get lost in details. This chapter focuses on Samba directives relevant to the Red Hat exam objectives.

In most cases, in SWAT, there's a Help hyperlink associated with each directive. In most cases, it highlights the relevant portion of the smb.conf man page. Most of the discussion relates to the Global Settings page; many of the security settings on that page may also be used to enhance user- and host-based security for individually shared directories.

The Printers and Wizard options are not covered in this book. If you're interested, try them out. They're not difficult to understand.

on the job

Before making any changes, back up the /etc/samba/smb.conf configuration file. SWAT overwrites not only the file, but also any related comments. If you do overwrite the smb.conf file without a backup, move or delete that file and then run the* yum reinstall samba-common *command.

Global Settings

To see what SWAT can do to the global settings in the smb.conf configuration file, click Globals from atop the SWAT web-based menu. Changes are straightforward. Enter desired changes it in the text boxes that follow. When the process is complete, click Commit Changes. Relevant options fall into several categories.

Base Options The Base Options shown in Figure 15-2 correspond to the Network Related Options in the smb.conf file discussed earlier in this chapter. With the possible exception of **netbios aliases** and **realm**, the directives may all be important, as described in Table 15-5.

FIGURE 15-2 SWAT global settings

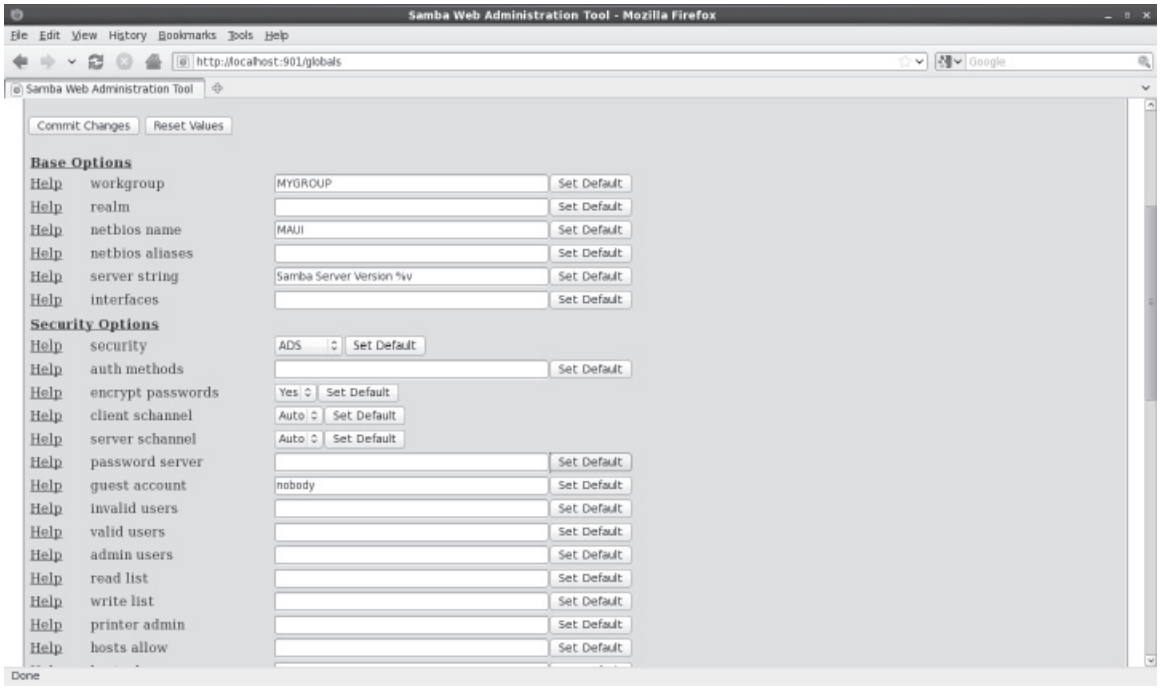


TABLE 15-5 SWAT Base Options

workgroup	The workgroup or domain to which the system belongs.
realm	The Kerberos realm for the domain, set to the DNS name of the Kerberos server such as server1.example.com. May be used if a system is configured as a Kerberos client, as discussed in Chapter 12.
netbios name	Windows host name; may be different from the DNS host name.
netbios aliases	Additional host names for the server.
server string	Description of the server shown to clients who browse this server.
interfaces	Devices and IP addresses allowed to connect.

Security Options A number of security options can be used both globally and for individual shares. Naturally, to use one of these options in a share, you should include it in the stanza associated with the share. The focus of this section is on those directives that can be used to help configure basic user- and host-based security for Samba. While these directives were discussed earlier in this chapter, the different perspective associated with SWAT may help you understand Samba better. As such, the following list is not comprehensive.

- **security** Basic directive for authentication on Samba systems; may be set to share, user, server, domain, or ads, as discussed earlier in this chapter.
- **password server** Reference to another system with the authentication database, usually a Samba or a pure Windows server.
- **guest account** Support for a nonprivileged account for connections.
- **invalid users** Users not allowed to access a system or share; for example, the following list prohibits users root, michael, and members of the project group:

```
invalid users root michael @project
```

- **valid users** Users allowed to access a system.
- **admin users** Users allowed administrative access, normally just to a share.
- **read list** Users given read-only access.
- **write list** Users given read/write access.
- **hosts allow** Hosts allowed access to a system (also known as **allow hosts**); for example, the following list supports access from all systems except one. May also use host and domain names.

```
hosts allow 192.168. EXCEPT 192.168.0.100
```

- **hosts deny** Hosts not allowed to access a system (also known as **deny hosts**).

exam Watch

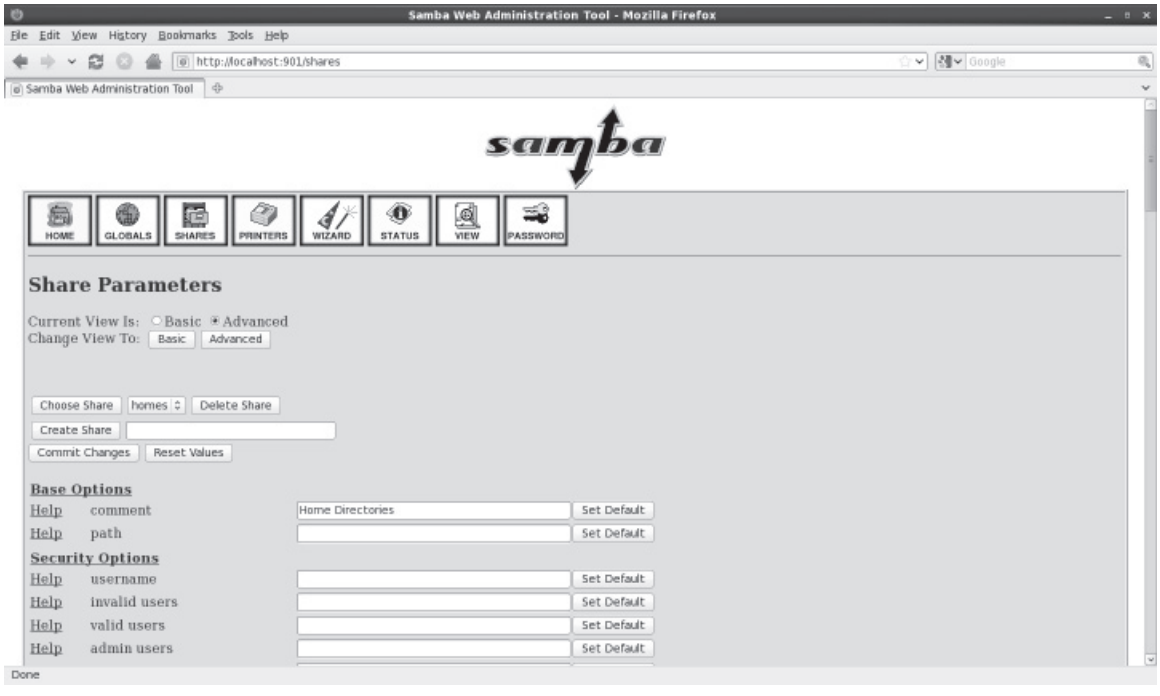
Host- and user-based security on a Samba system can be enhanced with the following directives: *invalid users, valid users, hosts allow,* ***and*** *hosts deny.*

To repeat, this section is focused only on those directives relevant to the RHCE objectives.

Share Settings

To see what SWAT can do to the share settings in the smb.conf configuration file, click Shares from atop the SWAT web-based menu. Click the drop-down text box associated with Choose Share, and then click Homes. Next to Change View To, click Advanced to reveal the screen shown in Figure 15-3. Relevant options fall into several categories.

FIGURE 15-3 SWAT share settings



Options on this page are straightforward. To create a second share, enter a name in the Create Share text box, and then click the Create Share button. You can then configure it in the text boxes that follow. When the process is complete, click Commit Changes. Many of the directives that appear were already explained in the global settings section but apply only to the local share. Those directives are not repeated here.

- **comment** Information included with the share name.
- **path** The path to the directory to be shared.
- **username** Substitute usernames where machine usernames are not available; rarely used.
- **force user** A user account assigned to all who connect to the share.
- **force group** A group account assigned to all who connect to the share.
- **read only** Shares so labeled can only be read.
- **guest only** Only guest user connections are allowed.
- **guest ok** Guest user connections are allowed; no password is required.

You'll note that most of these directives aren't covered here and in fact are rarely used.

Server Status

To see the current status of Samba on the system, click Server Status from atop the SWAT web-based menu. As shown in Figure 15-4, it supports control of the Samba, NetBIOS, and Winbind daemons. If there are active connections, active shared directories, and files in use, they're also listed in this view.

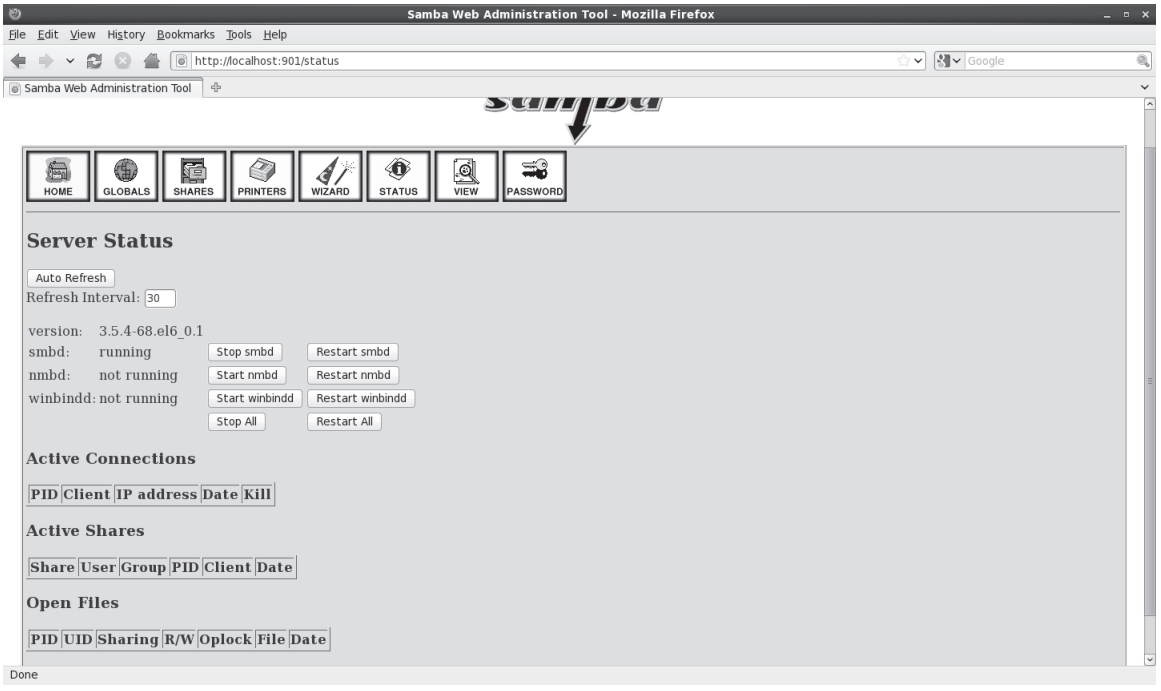
View Configuration

Any changes made with SWAT, once saved are written to the /etc/samba/smb.conf file. Click View. It includes a current read-only view of that file.

User Management

You can manage the Samba user authentication database with SWAT. To do so, click Password from atop the SWAT web-based menu. As shown in Figure 15-5, it supports password changes for current users. In addition, you can use the screen shown to add new users to the local Samba authentication database.

FIGURE 15-4 Default installed Apache home page



But these are just front ends to the **smbpasswd** command. In either case, the user has to exist in the Linux authentication database. For example, the following command takes the current user michael and prompts for a new password:

```
# smbpasswd michael
```

Alternatively, the following command adds user donna to the Samba authentication database, prompting for a password:

```
# smbpasswd -a donna
```

The **-d**, **-e**, and **-x** options can respectively disable, enable, and delete the given user from the Samba authentication database.

Test Changes to /etc/samba/smb.conf

After making any changes to /etc/samba/smb.conf, you should always test the system before putting it into production. A simple syntax check on the Samba configuration

FIGURE 15-5 SWAT server password management

file is possible with the **testparm** utility, as shown in Figure 15-6. This does not actually check to determine whether the service is running or functioning correctly; it checks only basic text syntax and command stanzas.

The directives that are displayed are share stanzas, along with associated directives. For example, the [homes] share is not read only and is not browsable to all clients.

Review User- and Host-Based Samba Security

As suggested in the RHCE objectives, you need to know how to configure "host-based and user-based security for" each service, including Samba. So while this section is repetitive, it's important.

To review, user-based security can be configured within the main Samba configuration file, smb.conf. Users specified in that file are configured in a separate database, normally in the /var/lib/samba directory, managed with the **smbpasswd** command.

FIGURE 15-6

Review the Samba configuration with testparm

```
[root@Maui ~]# testparm
Load smb config files from /etc/samba/smb.conf
rlimit_max: rlimit_max (1024) below 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
        wins server = 127.0.0.1
        cups options = raw

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

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

User-based security is enabled for the system with the following option:

```
security = user
```

Users can be specified as allowed or denied with the **invalid users** and **valid users** directives. Those directives can be applied generally in the Global Settings section or applied per share in the stanza where the shared directory is configured.

Host-based security can be configured in both the Samba configuration file and any associated **iptables**-based firewalls. Hosts can be allowed or denied with directives such as **hosts allow** and **hosts deny**. In some configurations, you'll see synonyms for those directives, such as **allow hosts** and **deny hosts**.

For example, the following **hosts allow** directive can limit access to the specified network(s):

```
hosts allow = 127. 192.168.122. 192.168.100.
```

The 127. is not required; localhost addresses are always allowed unless specifically included in a **hosts deny** directive.

Review Basic Samba Shares

Specifically for Samba, the RHCE objectives specify that you need to "provide network shares to specific clients" and to "provide network shares suitable for group collaboration."

To provide network shares to one or more specific clients, you'll need to include directives like **valid users** and **invalid users** in the stanza associated with a shared Samba directory.

To provide network shares suitable for group collaboration, you'll need to remember the following tasks:

- Set up an appropriate group and permissions on the directory to be shared.
- Configure the shared directory and files with the SELinux samba_share_t file type.
- Define a separate share stanza in the Samba configuration file, with appropriate values for **writable**, **create mask**, **directory mask**, and **write list**.

EXERCISE 15-2

Configuring Samba with Shares

In this exercise, you'll configure Samba to share a directory. For this purpose, you'll directly edit the /etc/samba/smb.conf file.

1. Create a /home/ftp/public directory. Change ownership to the ftp user and group, with full permissions for both (770).
2. Make sure to set appropriate SELinux settings for the directory with the following command:

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

   In addition, to make sure the changes survive a relabel of SELinux, you'll want to set up the file_contexts.local file in the /etc/selinux/targeted/contexts/files directory with a command such as the following:

   ```
   # semanage fcontext -a -t samba_share_t /home/ftp
   ```

3. Open the /etc/samba/smb.conf file in a text editor.

4. Configure Samba to share as public, in read-only mode, the /home/ftp/pub directory tree. In the Share Definitions section, you could add the following commands:

```
[pub]
     comment = shared FTP directory
     path = /home/ftp/pub
```

5. Allow guest access to all public shares. In smb.conf, this means adding the following line to the **[pub]** stanza:

```
     guest ok = yes
```

6. To create a guest account, you'll need to add the following command in smb.conf:

```
; guest account = pcguest
```

7. Create a guest account for pcguest, associate it with an unused UID and GID 600. (If you already have a user with this ID, substitute an unused ID number.) Set the password as "anonymous." While you can do this with the Red Hat User Manager discussed in Chapter 8, the quickest way to do this is with the following commands:

```
# useradd pcguest -u 600
# passwd pcguest
```

8. Create separate log files for each computer host that connects. This is already active by default with the following command:

```
log file = /var/log/samba/%m.log
```

9. Write and save changes to the smb.conf file.
10. You can see if Samba is already running with the **service smb status** command. If it's stopped, you can start it with the **service smb start** command. If it's running, you can make Samba reread your configuration file with the following command:

```
# service smb reload
```

This final option allows you to change your Samba configuration without disconnecting users from the Samba server.

CERTIFICATION OBJECTIVE 15.02

Samba as a Client

You can configure two types of clients through Samba. One connects to directories shared from Microsoft Windows servers or Samba servers on Linux/Unix. The second connects to shared printers from one of the same two types of servers. The Samba client commands are available from the samba-client RPM. With those commands, you should be able to find browse lists and mount shared directories locally.

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 by 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
```

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

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 3.5.4-68.el6_0.2]
      Sharename       Type      Comment
      ---------       ----      -------
      public          Disk      Public Stuff
      IPC$            IPC       IPC Service (Samba Server Version 3.5.4-68.el6_0.2)
      OfficePrinter@Maui Printer   in the office
      donna           Disk      Home Directories
Domain=[MYGROUP] OS=[Unix] Server=[Samba 3.5.4-68.el6_0.2]
```

From the displayed output, there's a share available named public. You can also use the **smbclient** command to make a client connection similar to an FTP connection with the following command:

```
$ 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. Of course, that user donna could also mount her remote home directory in a similar fashion, with a command like the following:

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

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

Automated Samba Mounts

As it certainly takes a few extra steps to set up a shared directory, it would be useful to automate the process. The standard method 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 is wrapped):

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

But 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:

```
//server/pub  /share  cifs rw,credentials=/etc/smbdonna 0 0
```

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

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

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

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. 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. Some of the parameters are misspelled; for example, **writable** is a synonym for **writeable**. But beyond those parameters, the **testparm** command can help identify problems. For example, Figure 15-7 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.
```

Sometimes, troubleshooting commands come in the output to other commands. For example, problems often appear in the output to various commands. Sometimes the output is straightforward, such as the following output to a specific **mount.cifs**

FIGURE 15-7

Review the Samba Configuration with testparm

```
[root@Maui ~]# testparm /etc/samba/smb.conf
Load smb config files from /etc/samba/smb.conf
rlimit_max: rlimit_max (1024) below 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
```

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 that 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 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.

Most Samba log files are located in the /var/log/samba directory. As shown in Figure 15-8, the log files are classified by the host or IP address of the client that connects to the server. In general, fewer messages mean success. For example, a connection to a localhost system, useful for troubleshooting, may include the following message in the log.__ffff_127.0.0.1 log file:

```
__ffff_127.0.0.1 (::ffff:127.0.0.1) connect to service michael
initially as user michael (uid=1000, gid=1000) (pid 23800)
```

The information therein suggests the use of both IPv4 and IPv6 addresses. 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.

Enable Remote Access

Network services aren't much good unless access is allowed from other systems. As with other RHEL 6 systems, each server has a firewall. The ports associated with Samba are closed by default. It's easy to set up access for a Samba server and a Samba client through the Firewall Configuration tool. All you'd need to do is to specify that the server or client is a trusted service.

As discussed in previous chapters, it's possible to limit remote access to certain IP addresses on a firewall. The **-s** switch in an **iptables**-based firewall configures source addresses. For example, the following rule would limit communication to the NetBIOS name service to any system but that on IP address 192.168.122.150:

```
-A INPUT -m state --state NEW -m udp -p udp -s !192.168.122.150 --dport 137 -j
ACCEPT
```

Remember, such specialized **iptables**-command rules can only be configured as a "Custom Rule" as described in Chapter 10.

SCENARIO & SOLUTION	
You need to set up sharing on a network with Microsoft computers	Install Samba, configure shared directories in /etc/samba/smb.conf. Make sure shared directories (except for user home directories) have the appropriate samba_share_t file 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, 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 **iptables**-based firewalls to limit access.
You want to set up user-based security for Samba	Set up appropriate **valid users** and **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 appropriate values for directory mask (2770) and create mask (2770). Set up a shared directory for a group per Chapter 8. Make sure the shared directory is set to samba_share_t.

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 ports 137, 138, 139, and 445. The key SELinux boolean is samba_enable_home_dirs. Shared directories should be set to the samba_share_t file 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 Red Hat SWAT tool, with a web-based interface, provides another way to configure smb.conf, as well as a front end to the **smbpasswd** command.

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 on the TCP/IP protocol stack.
- Samba includes a client and a server. Variations on the **mount -t cifs** or **/sbin/mount.cifs** commands support mounting of a shared Samba or even a shared Microsoft directory.
- The main Samba configuration file is /etc/samba/smb.conf. You can configure it in a text editor or a GUI tool such as SWAT.
- 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.
- The **mount.cifs** command can mount directories shared from a Samba or a Microsoft server.
- Samba shares can be mounted during the boot process with the help of the /etc/fstab configuration file.

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. A group that prefers Microsoft servers has set up a Windows 2008 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 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 service 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 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 stanza?

_____

10. What command adds user elizabeth to a smbpasswd or 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 CD that accompanies the book, in the Chapter15/ subdirectory. In case you haven't yet set up RHEL 6 on a system, refer to Chapter 1 for installation instructions.

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

SELF TEST ANSWERS

Samba Services

1. At minimum, to configure a Linux system as a DC, you need to configure the **security = user** directive. If it's on an active directory system, it's better to use the **security = ads** directive.
2. The command that forces the Samba service to reread the configuration file—without disconnecting Microsoft users or restarting the service—is **service smb reload**.
3. Open ports associated with communication to a Samba server are TCP ports 137, 138, 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 stanza is

```
guest ok = yes
```

10. The command that adds user elizabeth to either 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: Install and Start Samba

The chapter lab on Samba is designed to be easy to follow. However, you'll need explicit Linux knowledge to complete some specific steps. Answers to these steps can be found in the following:

1. You've installed the "CIFS file server" package group, which includes one RPM, samba.
2. One way to find all related Samba packages is with the **yum search samba | grep samba** command. You can then install noted packages with the **yum install packagename** command. Samba 4 packages are not supported for RHEL 6, at least not yet.
3. The Trusted Services section of the Firewall Configuration tool should make it easy to set up a local firewall to support communication to local Samba servers and clients.
4. You can use the **chkconfig smb on** command, the **ntsysv** tool, or the Service Configuration utility described in Chapter 7 to make sure Samba starts the next time you boot Linux.
5. Use the **service smb start** command to begin the Samba service. The **/etc/init.d/smb start** command is functionally equivalent.
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 service command such as **service smb status** command.

Lab 2: Review Samba Documentation

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 the following:

```
?hosts allow
```

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

From the browser, you should be able to review Samba documentation. This lab directs you to sample stanzas for shared directories. Of course, you can browse around other Samba documentation. Learn what you need as a reference for the job, or for an exam.

Lab 3: Configure Samba Global Settings

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

1. To use SWAT, you'll first need to enable it with the associated Extended Internet Super Server. One way to do so is with the following commands:

```
# /etc/init.d/xinetd.d restart
# chkconfig swat on
```

2. Next, direct a browser to http://localhost:901. SWAT should prompt you for a username and password. By default, the root username and associated password will support access to SWAT.
3. Many administrators stick with the standard Microsoft Windows **workgroup** name of **WORKGROUP**. You can find it in the output from the **smbclient -L *//clientname*** command.
4. To limit access to a Samba server, you can do so in the Globals section, with the **hosts allow** directive. Of course, you can also do so by directly editing the smb.conf file in the /etc/samba directory.
5. To limit access from a specific computer, you can do so in the Globals section, with the **hosts deny** directive. Of course, you can also do so by directly editing the smb.conf file in the /etc/samba directory.
6. Make sure to click the Commit Changes button in SWAT. Then open a command line, and make Samba read the changes with the **service smb reload** command.
7. Before committing the changes, you can test them with the **testparm** command.

8. 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 with the Trusted Services section of the Firewall Configuration tool.
9. 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: Configure a Share to a Home Directory

If successful, only one remote user will get access to his home directory via Samba, something that can be tested 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 **guest ok = no** to the **[homes]** stanza.
5. Add a **valid users =** ***username*** directive with the name of the desired user.
6. Commit the changes. Add the desired user to the Samba authentication database with the **smbpasswd -a** ***username*** command.
7. Restart or reload the Samba daemon, **smb**, under the Status menu or with the appropriate **service** command.
8. Save the changes made so far.
9. 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: Configure a Share to a Public Directory

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
```

   Create a new directory, /home/public; configure that share and call it public. Set the **hosts allow** setting, and list the domain associated with your network. Deny access to all other systems.

   The *777* setting for permissions grants 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 **service smb reload** command.
6. When testing the result from a remote system, any username in the local Samba database should work.

Lab 6: Configure a Shared Network Directory, Limited to a Group

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. Since that discussion in Chapter 8 was based on an RHCSA requirement, you'll have to repeat that process in this lab.

Once complete, you'll want to add the following directives to the stanza for the shared group directory:

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

Lab 7: Persistency Check

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. And 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 **chkconfig**. In this case, the **chkconfig smb on** command sets up the **smbd** daemon to become active when you boot Linux in a standard runlevel.
2. You can use various commands to perform an orderly shutdown, such as **shutdown**, **halt**, **init 0**, 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 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.