Linux Academy

Hands-on Lab

Create and Mount Samba and CIFS Fileshares

Contents

Related Courses

Linux Academy RHCSA Prep Course

Related Videos

Mount and Unmount CIFS and NFS Network File Systems

Need Help?

Linux Academy Community

... and you can always send in a support ticket on our website to talk to an instructor!

Lab Connection Information

- Labs may take up to five minutes to build
- The IP address of your server is located on the Hands-on Lab page
- Username: linuxacademy
- Password: 123456
- Root Password: 123456

In this lab, we deploy a Samba share for exchanging and sharing files across multiple systems. Using two CentOS 7 servers we set up and configure the share, add a user, and then set up the client server so it can access the share.

Install and Configure Samba Server

SSH into the first server and install the needed packages:

```
[linuxacademy@ip]$ sudo yum install samba samba-client samba-common cifs-utils
```

Note that the cifs-utils package may not be necessary if using Samba 3.

We now need to make edits to the Samba configuration file, which is located at /etc/samba.

```
[linuxacademy@ip]$ cd /etc/samba
[linuxacademy@ip]$ ll
total 20
-rw-r-r-. 1 root root    20 Jan 17 19:07 lmhosts
-rw-r-r-. 1 root root   667 Jan 17 19:07 smb.conf
-rw-r-r-. 1 root root 11327 Jan 17 19:07 smb.conf.example
```

Review the example configuration file, smb.conf.example; as we can see, this is a large file containing a variety of options, and it would not necessarily be the best option to copy this file directly to create our own configuration. Instead, open smb.conf in your chosen text editor.

This configuration is largely acceptable for most Samba installs. That said, we still have to make some changes and ensure we understand what our settings are doing.

Notice that each setting has its own section within the configuration file ([homes], [printers], etc.), with a global section at the top:

```
[global]
        workgroup = Samba
        security = user
        passdb backend = tdbsam
        printing = cups
        printcap name = cups
        load printers = yes
        cups options = raw
```

The workgroup line defines the name of the workgroup we are using.

security denotes who is able to access what; user is the Samba user security system, but we can also use Active Directory or a domain controller. In this case, we are leaving it as user.

passdb backend is for security on the database-level and is very rarely changed – any other options for this setting are external to Samba.

The next segment – printing, printcap name, load printers, and cups options – lets us define what service we are using for printing. By default, we use CUPS. This can be left as-is.

Next, the [homes] portion denotes whether users can share their home directories over Samba. This makes it easy for users to share files, as needed.

[printers] defines which printers to use, with the path defining the cups spool for printing. Because CUPS is not installed on this system, /var/tmp is used instead.

[print$] allows for printer drivers to be shared across accounts.

We can also build our own sections in the Samba configuration, which we want to do now. Add a new section, [myshare]. This allows us to create a Samba share and determine who has access to that share.

```
[myshare]
        comment = This is our test share
        path = /myshare
        guest ok = no
        writeable = yes
```

We first want to add a comment, which helps us identify this share. The path defines where the share is located, and guest ok determines whether guest accounts can be used. writable defines whether or not our share is read only for those who have access to the account.

Save and exit the file.

Create the Samba share:

```
[linuxacademy@ip]$ cd /
[linuxacademy@ip]$ sudo mkdir /myshare
[linuxacademy@ip]$ sudo chmod 777 /myshare
[linuxacademy@ip]$ echo "test file" > /myshare/testfile.txt
```

Notice that we set the permissions to 777, because we expect the Samba share to take care of any permissions issues.

Since we are using a CentOS 7 server, and SELinux is enabled by default, we want to set SELinux to permissive mode:

```
[linuxacademy@ip]$ sudo setenforce 0
```

In actual practice, properly configuring SELinux to work with the Samba share may be more appropriate;

however, this is out of scope for this lab.

With our share created, open up the smb.conf file again, and edit the load printers directive, since our system does not have printers:

```
load printers = no
```

Save and exit the file.

Test the file to see if the configuration is accurate:

```
[linuxacademy@ip]$ 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]"
Processing section "[print$]"
Processing section "[myshare]"
Loaded services file OK.
Server role: ROLE_STANDALONE
Press enter to see a dump of your service definitions
```

Everything should be configured correctly.

Start the Samba and nmb services. nmb allows Windows user to access the share.

```
[linuxacademy@ip]$ sudo systemctl start smb
[linuxacademy@ip]$ sudo systemctl start nmb
```

Confirm that both services are ready:

```
[linuxacademy@ip]$ sudo systemctl status -l smb
● smb.service - Samba SMB Daemon
   Loaded: loaded (/usr/lib/systemd/system/smb.service; disabled; vendor
preset: disabled)
   Active: active (running) since Mon 2017-03-27 17:15:45 UTC; 8s ago
 Main PID: 9714 (smbd)
   Status: "smbd: ready to serve connections..."
   CGroup: /system.slice/smb.service
           ├─9714 /usr/sbin/smbd
           ├─9715 /usr/sbin/smbd
           ├─9716 /usr/sbin/smbd
           └─9717 /usr/sbin/smbd
Mar 27 17:15:45 ip-10-0-1-92 systemd[1]: Starting Samba SMB Daemon...
Mar 27 17:15:45 ip-10-0-1-92 smbd[9714]: [2017/03/27 17:15:45.506325,  0] ../
lib/util/become_daemon.c:124(daemon_ready)
Mar 27 17:15:45 ip-10-0-1-92 systemd[1]: Started Samba SMB Daemon.
Mar 27 17:15:45 ip-10-0-1-92 smbd[9714]:   STATUS=daemon 'smbd' finished
starting up and ready to serve connections
```

```
[linuxacademy@ip]$ sudo systemctl status -l nmb
● nmb.service - Samba NMB Daemon
   Loaded: loaded (/usr/lib/systemd/system/nmb.service; disabled; vendor
preset: disabled)
   Active: active (running) since Mon 2017-03-27 17:15:47 UTC; 27s ago
 Main PID: 9727 (nmbd)
   Status: "nmbd: ready to serve connections..."
   CGroup: /system.slice/nmb.service
           └─9727 /usr/sbin/nmbd
Mar 27 17:15:47 ip-10-0-1-92 systemd[1]: Starting Samba NMB Daemon...
Mar 27 17:15:47 ip-10-0-1-92 nmbd[9727]: [2017/03/27 17:15:47.873762,  0] ../
lib/util/become_daemon.c:124(daemon_ready)
Mar 27 17:15:47 ip-10-0-1-92 nmbd[9727]:   STATUS=daemon 'nmbd' finished
starting up and ready to serve connections
See what shares are available for the Samba workgroup:
[linuxacademy@ip]$ nmblookup Samba
```

The IP address returned should be the one for the lab server.

Create Samba User Accounts

We want to be able to use user authentication to determine who has access to our Samba share. To do this, we cannot just use regular system accounts but instead need to create Samba accounts and passwords for our users or create a username map so users without system accounts can access the share. This is especially important for Windows users.

Use the smbpasswd command to create a user, *user*:

```
[linuxacademy@ip]$ sudo smbpasswd -a user
```

When prompted, enter a password. The user *user* is now added to the Samba database.

Alternatively, we can create a usermap file. Navigate to the /etc/samba directory, and create the file usermap. This file uses a key/pair style of formatting.

Since *user* is already an available account, we can set it up so the Windows user, *jsmith*, can access the share:

```
jsmith = user
```

Save and exit the file. We want to reference this file in our configuration, so Samba knows to use it. Open smb.conf, and add the following to the [global] section of the configuration:

```
username map = /etc/samba/usermap
```

Save and exit the file. Test the configuration:

```
[linuxacademy@ip]$ 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]"
Processing section "[print$]"
Processing section "[myshare]"
Loaded services file OK.
Server role: ROLE_STANDALONE
```

Since we installed the client tools on this server, we can also practically test our Samba share. Retrieve your server's IP address (use ifconfig or pull it from the Hands-on Lab page), and test as the *user* user. Note that for the Hands-on Lab we want to use the server's private IP.

```
[linuxacademy@ip]$ smbclient -U user -L <PRIVATEIP>
Domain=[Samba] OS=[Windows 6.1] Server=[Samba 4.4.4]
    Sharename       Type      Comment
    -----       --      ----
    print$          Disk      Printer Drivers
    myshare         Disk      This is our test share
    IPC$            IPC       IPC Service (Samba 4.4.4)
    user            Disk      Home Directories
Domain=[Samba] OS=[Windows 6.1] Server=[Samba 4.4.4]
    Server               Comment
    -----            ----
    IP-10-0-1-94         Samba 4.4.4
    Workgroup            Master
    -----            ----
    Samba                IP-10-0-1-94
```

Enter the password when prompted.

Here we can see our available shares. To further test and see if we can access the share itself:

```
[linuxacademy@ip]$ smbclient -U user //<PRIVATEIP>/myshare
Domain=[Samba] OS=[Windows 6.1] Server=[Samba 4.4.4]
smb: \>
```

Enter the password when prompted. This gives us a Samba prompt (smb: \>), which allows us to access our share. Run the ls command to ensure the test file is there, then quit

```
smb: \> ls
  .                         D        0  Tue Mar 28 18:38:00 2017
  ..                        DR        0  Tue Mar 28 18:37:45 2017
  testfile.txt              N       10  Tue Mar 28 18:38:00 2017
        8377344 blocks of size 1024. 6817160 blocks available
smb: \> quit
```

Finally, run smbstatus to see who is connected and what shares they are using.

```
[linuxacademy@ip]$ sudo smbstatus
```

This will output a mostly-blank response, since we have no connected clients and no users are on our share.

Configure Client and Mount Shares

SSH into your second provided server. Install the needed utilities for using Samba:

```
[linuxacademy@ip2]$ sudo yum install samba samba-client samba-common cifs-utils
```

Connect to the Samba share, using the IP address of the serving instance:

```
[linuxacademy@ip2]$ smbclient -U user -L <PRIVATEIP>
Domain=[Samba] OS=[Windows 6.1] Server=[Samba 4.4.4]
    Sharename       Type      Comment
    -----       --      ----
    print$          Disk      Printer Drivers
    myshare         Disk      This is our test share
    IPC$            IPC       IPC Service (Samba 4.4.4)
    user            Disk      Home Directories
Domain=[Samba] OS=[Windows 6.1] Server=[Samba 4.4.4]
    Server               Comment
    -----            ----
    IP-10-0-1-94         Samba 4.4.4
    Workgroup            Master
    -----            ----
    Samba                IP-10-0-1-94
```

Enter the password when prompted. The output should match the output from earlier in the lab. We can also view the share:

```
[linuxacademy@ip2]$ smbclient -U user //<PRIVATEIP>/myshare
Domain=[Samba] OS=[Windows 6.1] Server=[Samba 4.4.4]
smb: \>
```

quit the Samba prompt.

However, we want to mount the share to our system, not only access it through the Samba prompt.

To avoid SELinux issues, cd into the /mnt directory. Create a directory:

```
[linuxacademy@ip2]$ cd /mnt
[linuxacademy@ip2]$ sudo mkdir sambashare
```

Mount the file system:

```
[linuxacademy@ip2]$ sudo mount -t cifs -o username=user //IP/myshare /mnt/
sambashare
```

Note that for Samba 3, `username=user` should instead be `user=user`. Input your password when prompted. Confirm its mount:

```
[linuxacademy@ip2]$ df -h
Filesystem           Size  Used Avail Use% Mounted on
/dev/xvda1           8.0G  1.5G  6.6G  19% /
devtmpfs             477M     0  477M   0% /dev
tmpfs                496M     0  496M   0% /dev/shm
tmpfs                496M   13M  483M   3% /run
tmpfs                496M     0  496M   0% /sys/fs/cgroup
tmpfs                100M     0  100M   0% /run/user/0
tmpfs                100M     0  100M   0% /run/user/1001
//10.0.1.94/myshare  8.0G  1.5G  6.6G  19% /mnt/sambashare
```

As a test, create a second file in the Samba share:

```
[linuxacademy@ip2]$ sudo sh -c "echo "Another file" > /mnt/sambashare/another.
txt"
```

Return to the base `/mnt` directory and unmount the share:

```
[linuxacademy@ip2]$ cd /mnt
[linuxacademy@ip2]$ sudo umount sambashare
```

But what if we want to mount the system on boot? Open `/etc/fstab`, and add the mount as you would another file system but with included username and password information.

```
# samba mount for share - using username and password
//<PRIVATEIP>/myshare    /mnt/sambashare cifs    username=user,password=password
0 0
```

Save and edit.

Mount the file system with `mount -a`. Confirm:

```
[linuxacademy@ip2]$ mount -a
[linuxacademy@ip2]$ df -h
Filesystem           Size  Used Avail Use% Mounted on
/dev/xvda1           8.0G  1.5G  6.6G  19% /
devtmpfs             477M     0  477M   0% /dev
tmpfs                496M     0  496M   0% /dev/shm
tmpfs                496M   13M  483M   3% /run
tmpfs                496M     0  496M   0% /sys/fs/cgroup
tmpfs                100M     0  100M   0% /run/user/0
tmpfs                100M     0  100M   0% /run/user/1001
//10.0.1.94/myshare  8.0G  1.5G  6.6G  19% /mnt/sambashare
```

Unmount the share:

```
[linuxacademy@ip2]$ sudo umount /mnt/sambashare
```

While this works, it is not the most secure option for mounting the system – notably, it involves inputting a password in plain text into our /etc/fstab file. Remove the line from /etc/fstab. Instead, we can to use a credentials file:

```
# samba mount for share - user credentials file
//ip/myshare    /mnt/sambashare cifs    credentials=/etc/samba/creds.txt 0   0
```

Save and exit, then create the creds.txt file in /etc/samba.

```
username=user
password=password
```

Save and exit.

Mount the file system:

```
[linuxacademy@ip2]$ mount -a
[linuxacademy@ip2]$ df -h
Filesystem           Size  Used Avail Use% Mounted on
/dev/xvda1           8.0G  1.5G  6.6G  19% /
devtmpfs             477M     0  477M   0% /dev
tmpfs                496M     0  496M   0% /dev/shm
tmpfs                496M   13M  483M   3% /run
tmpfs                496M     0  496M   0% /sys/fs/cgroup
tmpfs                100M     0  100M   0% /run/user/0
tmpfs                100M     0  100M   0% /run/user/1001
//10.0.1.94/myshare  8.0G  1.5G  6.6G  19% /mnt/sambashare
```

Return to the first server, and run smbstatus.

```
[linuxacademy@ip]$ sudo smbstatus
```

Here we can see that we're connected from the second server. You can now complete the lab!