

File Sharing

Sharing files with Samba and NFS.

- [Linux - Samba Setup Rocky 9](#)
- [Linux - Samba Setup No Authentication](#)

Linux - Samba Setup Rocky

9

Last verified: October 2026 for RHEL / Rocky Linux 9 against the Red Hat Samba guide, the smb.conf manual and Microsoft's SMB guest access documentation.

Step 1: Install Samba on Linux

To get started out with **Samba**, install the **Samba** core packages including the client package:

```
dnf install -y samba samba-common samba-client
```

Install-Samba in Linux

```
[tecmint@rocky-8 ~]$  
[tecmint@rocky-8 ~]$ sudo dnf install samba samba-common samba-client  
Last metadata expiration check: 0:27:23 ago on Sat 11 Dec 2021 11:51:53 PM EAT.  
Package samba-common-4.13.3-5.el8_4.noarch is already installed.  
Dependencies resolved.
```

Package	Architecture	Version	Repository	Size
Installing:				
samba	x86_64	4.14.5-2.el8	baseos	846 k
samba-client	x86_64	4.14.5-2.el8	baseos	700 k
Upgrading:				
libsmbclient	x86_64	4.14.5-2.el8	baseos	146 k
libwbclient	x86_64	4.14.5-2.el8	baseos	120 k
samba-client-libs	x86_64	4.14.5-2.el8	baseos	5.4 M
samba-common	noarch	4.14.5-2.el8	baseos	219 k

The command installs the packages specified along with the dependencies as displayed on the output. After the installation is complete, you will get a summary of all the packages that have been installed.

Samba Installation Completes

```
Verifying      : samba-common-4.14.5-2.el8.noarch      11/14
Verifying      : samba-common-4.13.3-5.el8_4.noarch   12/14
Verifying      : samba-common-libs-4.14.5-2.el8.x86_64 13/14
Verifying      : samba-common-libs-4.13.3-5.el8_4.x86_64 14/14
Installed products updated.

Upgraded:
  libsmbclient-4.14.5-2.el8.x86_64      libwbclient-4.14.5-2.el8.x86_64
  samba-client-libs-4.14.5-2.el8.x86_64  samba-common-4.14.5-2.el8.noarch
  samba-common-libs-4.14.5-2.el8.x86_64

Installed:
  samba-4.14.5-2.el8.x86_64      samba-client-4.14.5-2.el8.x86_64
  samba-common-tools-4.14.5-2.el8.x86_64  samba-libs-4.14.5-2.el8.x86_64

Complete!
[tecmint@rocky-8 ~]$
```

Step 2: Create and Configure Samba Shares

Once all the **samba** packages have been installed, the next step is to configure the **samba shares**. A samba share is simply a directory that is going to be shared across client systems in the network.

Here, we are going to create a samba share called **/data** in the **/srv/tecmint/** directory path.

```
mkdir -p /srv/tecmint/data
```

Next, we will assign permissions and ownership as follows.

```
chmod -R 755 /srv/tecmint/data
chown -R nobody:nobody /srv/tecmint/data
semanage fcontext -a -t samba_share_t "/srv/tecmint/data(/.*)?"
restorecon -Rv /srv/tecmint/data
```

The **semanage + restorecon** pair sets the SELinux label permanently, so it survives a file system relabel (**chcon** does not). If **semanage** is missing, install it with `dnf install policycoreutils-python-utils`.

Create Samba Share Directory

```
[tecmint@rocky-8 ~]$ sudo mkdir -p /srv/tecmint/data
[tecmint@rocky-8 ~]$
[tecmint@rocky-8 ~]$
[tecmint@rocky-8 ~]$ sudo chmod -R 755 /srv/tecmint/data
[tecmint@rocky-8 ~]$
[tecmint@rocky-8 ~]$
[tecmint@rocky-8 ~]$ sudo chown -R nobody:nobody /srv/tecmint/data
[tecmint@rocky-8 ~]$
[tecmint@rocky-8 ~]$
[tecmint@rocky-8 ~]$ sudo chcon -t samba_share_t /srv/tecmint/data
[tecmint@rocky-8 ~]$
```

Next, we are going to make some configurations in the **smb.conf** configuration file which is Samba's main configuration file. But before we do so, we will back up the file by renaming it with a different file extension.

```
mv /etc/samba/smb.conf /etc/samba/smb.conf.bak
```

Next, we are going to create a new configuration file.

```
vim /etc/samba/smb.conf
```

We will define policies on who can access the samba share by adding the lines shown in the configuration file.

```
[global]
workgroup = WORKGROUP
server string = Samba Server %v
netbios name = samba-server
security = user
map to guest = bad user
dns proxy = no
```

```
[Public]
path = /srv/tecmint/data
browsable = yes
writable = yes
guest ok = yes
read only = no
```

Save and exit the configuration file.

Do not add `ntlm auth = true` (older guides include it). It is an alias for `ntlmv1-permitted` and re-enables insecure NTLMv1. The default, `ntlmv2-only`, works with all modern clients.

To verify the configurations made, run the command:

```
testparm
```

Verify Samba Configuration

```
[tecmint@rocky-8 ~]$  
[tecmint@rocky-8 ~]$ sudo testparm  
Load smb config files from /etc/samba/smb.conf  
Loaded services file OK.  
Weak crypto is allowed  
  
Server role: ROLE_STANDALONE  
  
Press enter to see a dump of your service definitions  
  
# Global parameters  
[global]  
    dns proxy = No  
    map to guest = Bad User  
    security = USER  
    server string = Samba Server %v  
    idmap config * : backend = tdb  
  
[Public]  
    guest ok = Yes  
    path = /srv/tecmint/data  
    read only = No
```

Next, start and enable Samba daemons as shown.

```
systemctl enable --now smb;systemctl enable --now nmb
```

Be sure to confirm that both the **smb** and **nmb** daemons are running. (nmb only provides NetBIOS name resolution and network browsing; it is optional if your clients connect by DNS name or IP.)

```
systemctl status smb;systemctl status nmb
```

Verify Samba Status

```
[tecmint@rocky-8 ~]$ sudo systemctl status smb ←
● smb.service - Samba SMB Daemon
   Loaded: loaded (/usr/lib/systemd/system/smb.service; enabled; vendor preset: d
   Active: active (running) since Sun 2021-12-12 00:58:43 EAT; 11min ago
     Docs: man:smbd(8)
           man:samba(7)
           man:smb.conf(5)
  Main PID: 8007 (smbd)
    Status: "smbd: ready to serve connections..."
     Tasks: 4 (limit: 11090)

[tecmint@rocky-8 ~]$ sudo systemctl status nmb ←
● nmb.service - Samba NMB Daemon
   Loaded: loaded (/usr/lib/systemd/system/nmb.service; enabled; vendor preset: di
   Active: active (running) since Sun 2021-12-12 00:59:00 EAT; 11min ago
     Docs: man:nmbd(8)
           man:samba(7)
           man:smb.conf(5)
  Main PID: 8105 (nmbd)
    Status: "nmbd: ready to serve connections..."
     Tasks: 1 (limit: 11090)
  Memory: 2.8M
```

To enable access to samba share from remote Windows systems, you need to open the samba protocol on the firewall.

```
firewall-cmd --permanent --add-service=samba
firewall-cmd --reload
firewall-cmd --list-services
```

Step 3: Accessing Samba Share from Windows

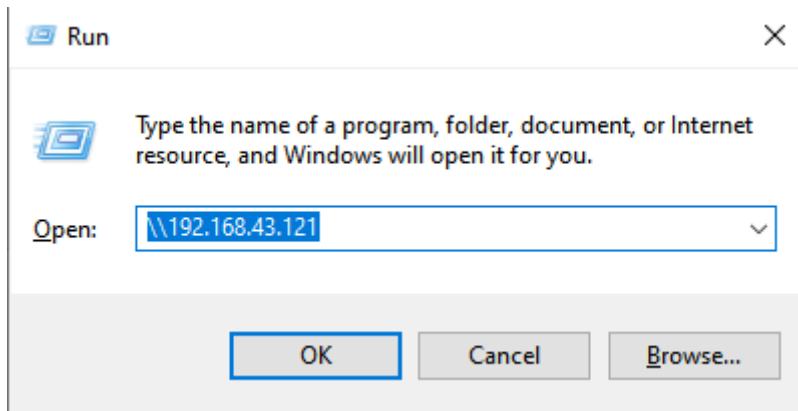
Thus far, we have installed **samba** and configured our **samba share**. We are now ready to access it remotely. To do this on a Windows client, press the Windows logo **key + R** to launch the **Run** dialog.

Current Windows versions block guest (no password) access to SMB shares by default: Windows 10 1709+ and Windows Server 2019+ disable insecure guest logons, and Windows 11 24H2 / Windows Server 2025 also require SMB signing, which guest sessions cannot do. If the **Public** share will not open, use the authenticated share from Step 4 rather than weakening the client security settings.

In the textfield provided, enter the samba server's IP address as shown:

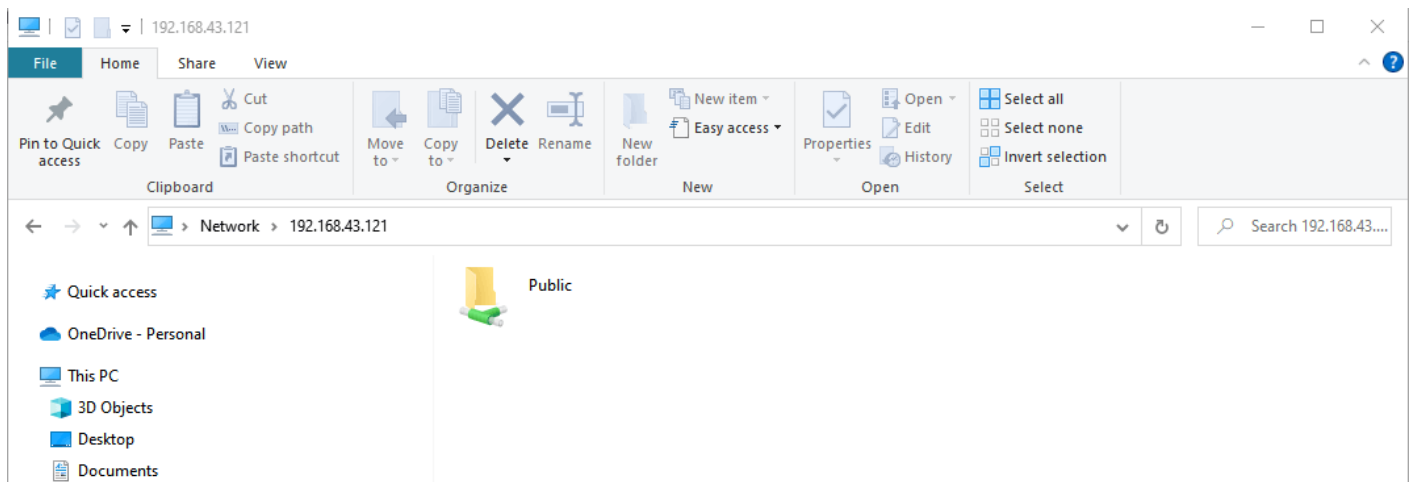
\\server-ip

Access Samba Share from Windows



The following window labeled '**Public**' will pop up. Remember, this is the directory that points to our samba share in the **/srv/tecmint/data** directory.

Access Samba Share Directory on Windows

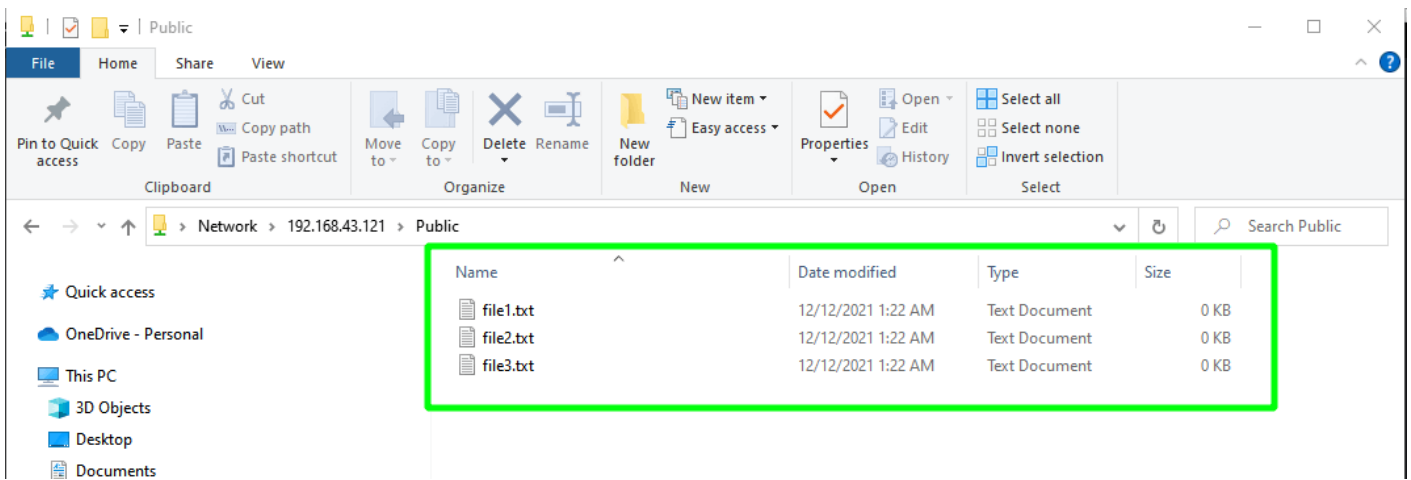


Currently, our directory is empty as we have not created any files. So, we will head back to our terminal and create a few files in the samba share directory.

```
cd /srv/tecmint/data
touch file{1..3}.txt
```

Now, we will navigate to the '**Public**' folder where the files we created earlier will be displayed.

Access Samba Share Files on Windows



Perfect. We have successfully managed to access our **samba share**. However, our directory is accessible to anyone and everybody can edit and delete files at will, which is not recommended especially if you plan to host sensitive files.

In the next step, we will demonstrate how you can create and configure a secure samba share directory.

Step 4: Secure Samba Share Directory

First, we will create a new samba user.

```
useradd smbuser
```

Next, we will configure a password for the samba user. This is the password that will be used during authentication.

```
smbpasswd -a smbuser
```

Create Samba User

```
[tecmint@rocky-8 ~]$ sudo useradd smbuser  
[tecmint@rocky-8 ~]$  
[tecmint@rocky-8 ~]$  
[tecmint@rocky-8 ~]$ sudo smbpasswd -a smbuser  
New SMB password:  
Retype new SMB password:  
Added user smbuser.  
[tecmint@rocky-8 ~]$
```

Next, we will create a new group for our secure samba share and add the new samba user.

```
groupadd smb_group  
usermod -aG smb_group smbuser
```

Thereafter, create yet another samba share which will be securely accessed. In our case, we have created another directory in the same path as the

```
mkdir -p /srv/tecmint/private
```

Then configure the file permissions for the samba share

```
chmod -R 770 /srv/tecmint/private  
semanage fcontext -a -t samba_share_t "/srv/tecmint/private(/.*)?"  
restorecon -Rv /srv/tecmint/private  
chown -R root:smb_group /srv/tecmint/private
```

Once again, access the Samba configuration file.

```
$ sudo vim /etc/samba/smb.conf
```

Add these lines to define to secure samba share.

```
[Private]  
path = /srv/tecmint/private  
valid users = @smb_group  
guest ok = no  
writable = no  
browsable = yes
```

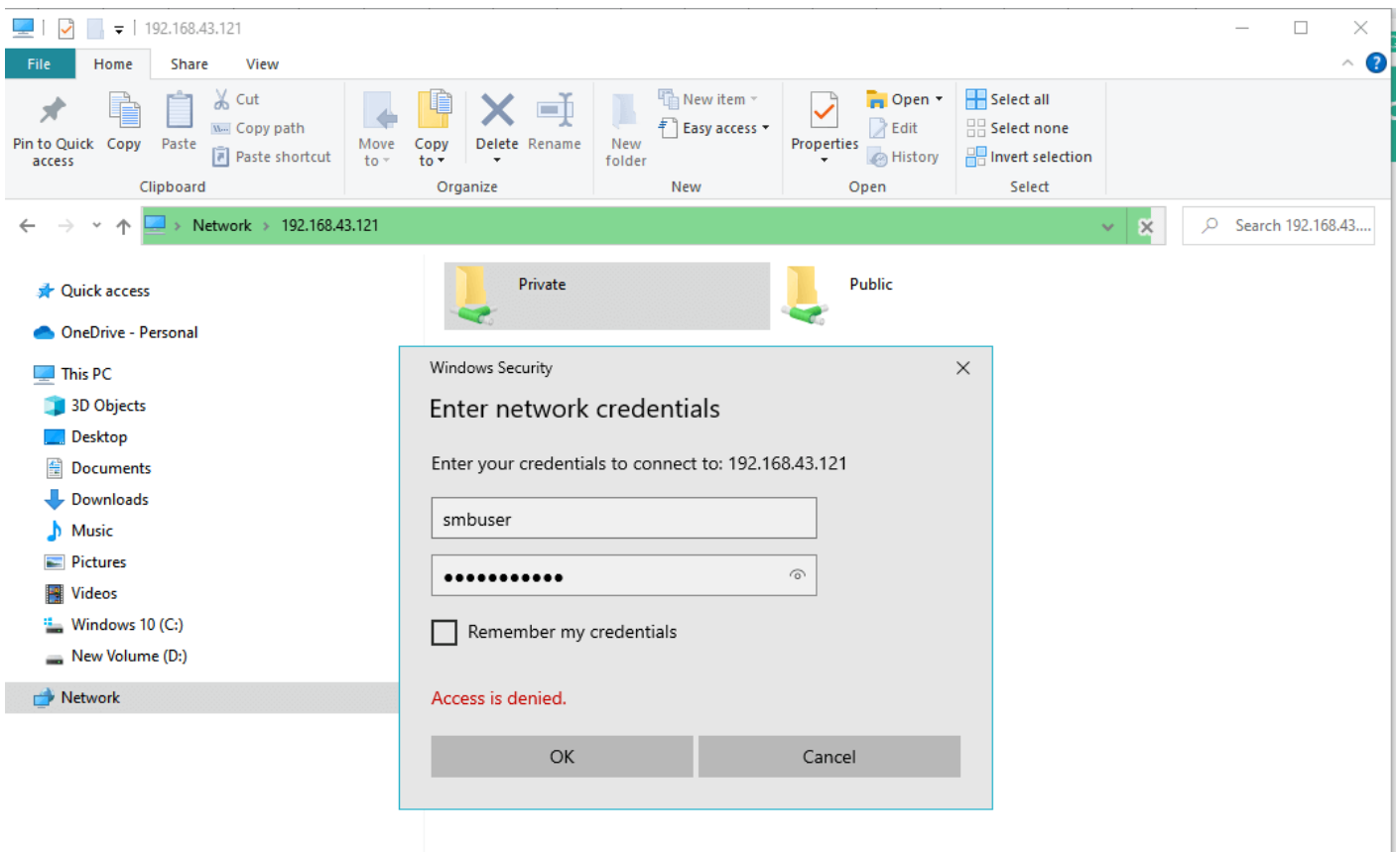
Save the changes and exit.

Finally, restart all the samba daemons as shown.

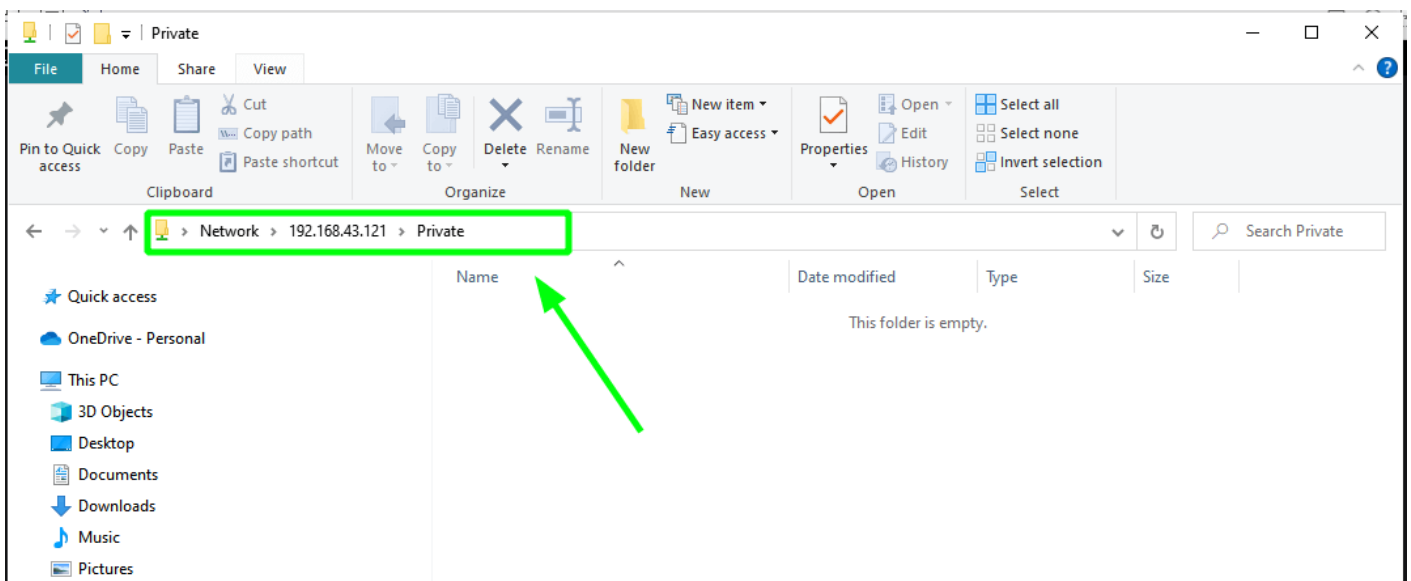
```
systemctl restart smb;systemctl restart nmb
```

When you access your server this time around, you will notice an additional '**Private**' folder. To access the folder, you will be required to authenticate with the Samba user's credentials. Provide the username and password of the user you created in the previous step and click '**OK**'.

Samba User Authentication



Samba Private Share Directory



Step 5: Accessing Samba Share from Linux Client

To access the share from a Linux client, first, ensure that the Samba client package is installed.

```
$ dnf install -y samba-client
```

Then use the **smbclient** command as follows

```
smbclient //server-ip/private -U smbuser
```

Access Samba Share from Linux

```
[james@rocky-8 ~]$  
[james@rocky-8 ~]$ smbclient '\\192.168.43.121\private' -U smbuser  
Enter WORKGROUP\smbuser's password:  
Try "help" to get a list of possible commands.  
smb: \>  
smb: \> ls  
.  
..  
D 0 Sun Dec 12 01:53:13 2021  
D 0 Sun Dec 12 01:53:13 2021  
36682240 blocks of size 1024. 23871400 blocks available  
smb: \> 
```

And this concludes this guide on setting up **Samba** on **RHEL**, **CentOS Stream**, **Rocky Linux**, and **AlmaLinux**. Your feedback on this guide will be highly appreciated.

Some taken from <https://www.tecmint.com/install-samba-rhel-rocky-linux-and-almalinux/>

Sources

- https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/9/html/configuring_and_using_network_file_services/assembly_using-samba-as-a-server_configuring-and-using-network-file-services
- <https://www.samba.org/samba/docs/current/man-html/smb.conf.5.html>
- <https://learn.microsoft.com/en-us/windows-server/storage/file-server/enable-insecure-guest-logons-smb2-and-smb3>
- <https://www.tecmint.com/install-samba-rhel-rocky-linux-and-almalinux/>

Linux - Samba Setup No Authentication

Security warning: older versions of this page shared `/` (the whole file system) writable to guests with `force user = root`. That gives **anyone who can reach the server full root write access** to every file on it, with no password. Samba's `smb.conf` manual notes that whatever privileges the guest account has are available to any client connecting to a guest share, and `force user = root` makes that account root. A guest-only setup like this is only for an isolated lab network and must **never** be exposed to an untrusted network or the internet. The `[httpd]` and `[html]` shares below also let guests write to the web server configuration and content. This page now shares a dedicated folder as the unprivileged guest account instead.

Step 1: Install Samba on Linux

To get started out with **Samba**, install the **Samba** core packages including the client package:

```
dnf install -y samba samba-common samba-client
```

The command installs the packages specified along with the dependencies as displayed on the output. After the installation is complete, you will get a summary of all the packages that have been installed.

Samba Installation Completes

Step 2: Create and Configure Samba Shares

Once all the **samba** packages have been installed, the next step is to configure the **samba shares**. A samba share is simply a directory that is going to be shared across client systems in the network.

Next, we are going to make some configurations in the **smb.conf** configuration file which is Samba's main configuration file. But before we do so, we will back up the file by renaming it with a

different file extension.

```
mv /etc/samba/smb.conf /etc/samba/smb.conf.bak
```

Next, we are going to create a new configuration file.

```
vim /etc/samba/smb.conf
```

This will define the samba share by adding the lines shown in the configuration file.

```
[global]
unix charset = UTF-8
hosts allow = 192.168.x.
map to guest = Bad User
log file = /var/log/samba/log.%m
log level = 1
server role = standalone server

[httpd]
path = /etc/httpd/
read only = no
guest ok = yes
guest only = yes
force user = apache
force group = apache

[html]
path = /var/www/html/
read only = no
guest ok = yes
guest only = yes
force user = apache
force group = apache

[public]
# Guest-only share: no password, guests run as the guest account (default: nobody)
path = /srv/samba/public
read only = no
guest ok = yes
guest only = yes
```

Save and exit the configuration file.

Create the dedicated public folder, make it owned by the guest account (`nobody`) so guests can write to it, and give it the Samba SELinux label:

```
mkdir -p /srv/samba/public
chown -R nobody:nobody /srv/samba/public
chmod 0755 /srv/samba/public
semanage fcontext -a -t samba_share_t "/srv/samba/public(/.*)?"
restorecon -Rv /srv/samba/public
```

There is no `force user` line on this share: with `guest only = yes` every connection already runs as the `guest account` (default `nobody`), so guests can only write where `nobody` can. Never set `force user = root` on a guest share. If `semanage` is missing, install `policycoreutils-python-utils`.

To verify the configurations made, run the command:

```
testparm
```

This verifies Samba Configuration

Next, start and enable Samba daemons as shown.

```
systemctl enable --now smb;systemctl enable --now nmb
```

Be sure to confirm that both the **smb** and **nmb** daemons are running.

```
systemctl status smb;systemctl status nmb
```

This verified Samba Status

Step 3: Accessing Samba Share from Windows

Thus far, we have installed **samba** and configured our **samba share**. We are now ready to access it remotely. To do this on a Windows client, press the Windows logo `key + R` to launch the **Run** dialog.

In the textfield provided, enter the samba server's IP address as shown:

```
\\server-ip
```

This accesses Samba Share from Windows

Some data from here

https://wiki.samba.org/index.php/Setting_up_Samba_as_a_Standalone_Server

Sources

- https://wiki.samba.org/index.php/Setting_up_Samba_as_a_Standalone_Server
- <https://www.samba.org/samba/docs/current/man-html/smb.conf.5.html>
- https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/9/html/configuring_and_using_network_file_services/assembly_using-samba-as-a-server_configuring-and-using-network-file-services