How to Install Samba in Ubuntu

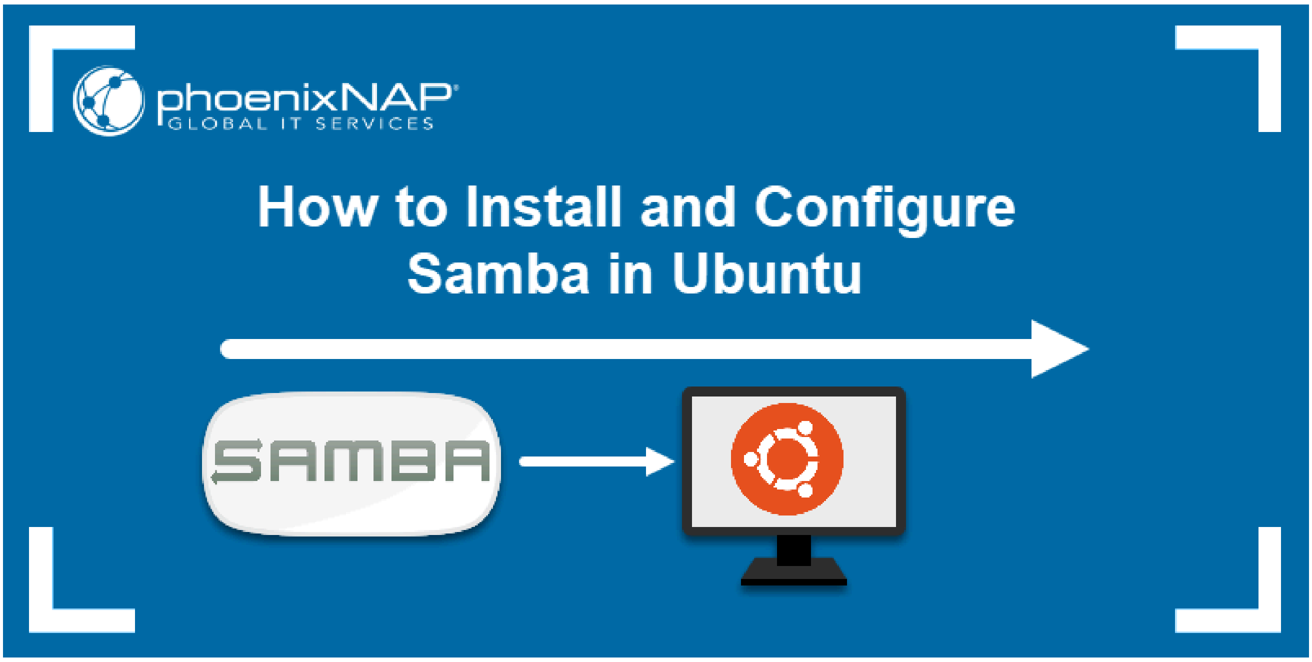
November 24, 2022

UBUNTU

Introduction

Samba is an open-source utility that enables file sharing between machines running on a single network. It enables Linux machines to share files with machines running different operating systems, such as Windows.

This tutorial teaches you how to install and configure Samba in Ubuntu 20.04 or 22.04.


phoenixNAP GLOBAL IT SERVICES
How to Install and Configure Samba in Ubuntu
SAMBA

Prerequisites

- Ubuntu system (this guide uses Ubuntu 22.04. The same steps work on Ubuntu 20.04).
- Sudo privileges.
- A text editor (this tutorial uses Vim).

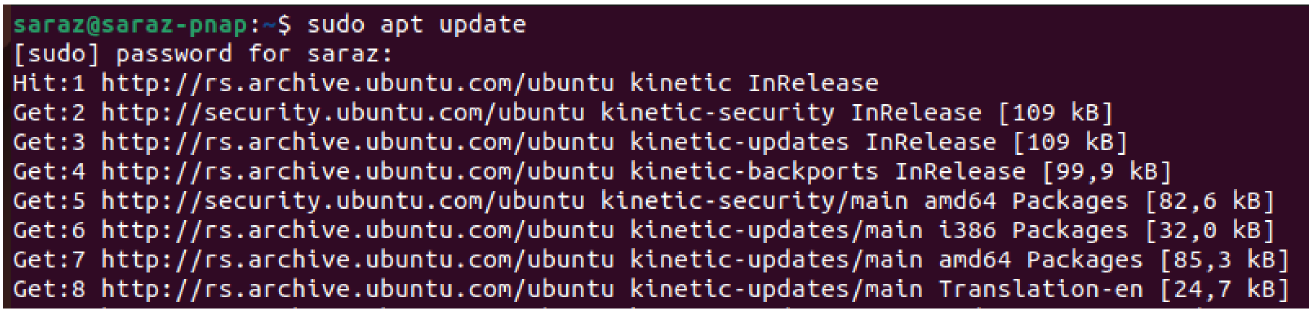
How to Install and Configure Samba on Ubuntu

Most Linux package managers have Samba in their default repository. To configure Samba on Ubuntu, follow the steps below:

Step 1: Install Samba

1. Start by updating the package registry:

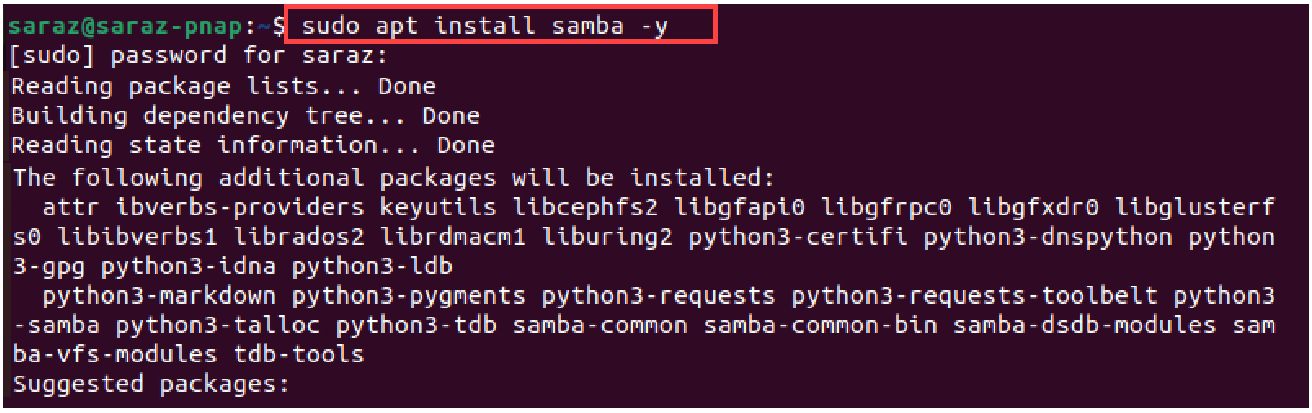
```
sudo apt update
```



```
saraz@saraz-pnap:~$ sudo apt update
[sudo] password for saraz:
Hit:1 http://rs.archive.ubuntu.com/ubuntu kinetic InRelease
Get:2 http://security.ubuntu.com/ubuntu kinetic-security InRelease [109 kB]
Get:3 http://rs.archive.ubuntu.com/ubuntu kinetic-updates InRelease [109 kB]
Get:4 http://rs.archive.ubuntu.com/ubuntu kinetic-backports InRelease [99,9 kB]
Get:5 http://security.ubuntu.com/ubuntu kinetic-security/main amd64 Packages [82,6 kB]
Get:6 http://rs.archive.ubuntu.com/ubuntu kinetic-updates/main i386 Packages [32,0 kB]
Get:7 http://rs.archive.ubuntu.com/ubuntu kinetic-updates/main amd64 Packages [85,3 kB]
Get:8 http://rs.archive.ubuntu.com/ubuntu kinetic-updates/main Translation-en [24,7 kB]
```

2. Next, use apt to install Samba. Include the **-y** argument to auto-approve any queries during the installation:

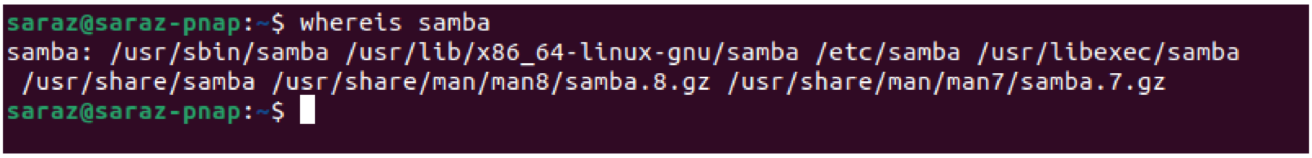
```
sudo apt install samba -y
```



```
saraz@saraz-pnap:~$ sudo apt install samba -y
[sudo] password for saraz:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  attr ibverbs-providers keyutils libcephfs2 libgfapi0 libgfrpc0 libgfxdr0 libglusterf
s0 libibverbs1 librados2 librdmacm1 liburing2 python3-certifi python3-dnspython python
3-gpg python3-idna python3-ldb
  python3-markdown python3-pygments python3-requests python3-requests-toolbelt python3
-samba python3-talloc python3-tdb samba-common samba-common-bin samba-dsdb-modules sam
ba-vfs-modules tdb-tools
Suggested packages:
```

3. Verify the installation with:

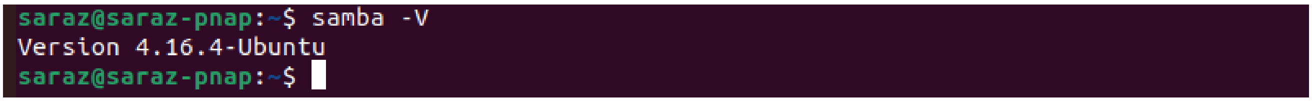
```
whereis samba
```



```
saraz@saraz-pnap:~$ whereis samba
samba: /usr/sbin/samba /usr/lib/x86_64-linux-gnu/samba /etc/samba /usr/libexec/samba
 /usr/share/samba /usr/share/man/man8/samba.8.gz /usr/share/man/man7/samba.7.gz
saraz@saraz-pnap:~$ 
```

The output prints the directory containing Samba. Another method of verifying the installation is by checking the Samba version:

```
samba -V
```

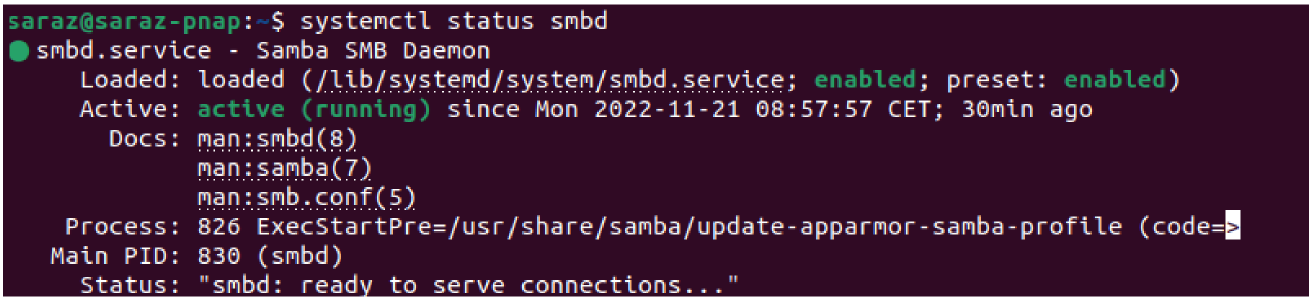


```
saraz@saraz-pnap:~$ samba -V
Version 4.16.4-Ubuntu
saraz@saraz-pnap:~$ 
```

The output shows that the system installed Samba version 4.16.4.

4. Lastly, confirm that Samba is running with:

```
systemctl status smbd
```



```
saraz@saraz-pnap:~$ systemctl status smbd
● smbd.service - Samba SMB Daemon
     Loaded: loaded (/lib/systemd/system/smbd.service; enabled; preset: enabled)
     Active: active (running) since Mon 2022-11-21 08:57:57 CET; 30min ago
       Docs: man:smbd(8)
             man:samba(7)
             man:smb.conf(5)
    Process: 826 ExecStartPre=/usr/share/samba/update-apparmor-samba-profile (code=>
   Main PID: 830 (smbd)
     Status: "smbd: ready to serve connections..."
```

The output shows that the **smbd** service is enabled and running.

Step 2: Create a Shared Directory

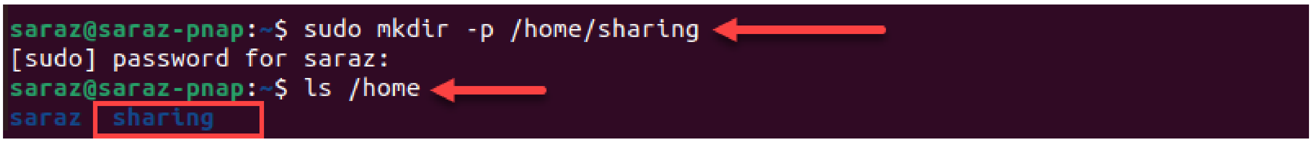
1. To share files with Samba, create a directory containing files for sharing. Use mkdir **-p** to create the directory under **/home**:

For example, make a directory called ***sharing*** with:

```
sudo mkdir -p /home/sharing
```

2. Use ls to verify the outcome.

```
ls
```



```
saraz@saraz-pnap:~$ sudo mkdir -p /home/sharing
[sudo] password for saraz:
saraz@saraz-pnap:~$ ls /home
saraz  sharing
```

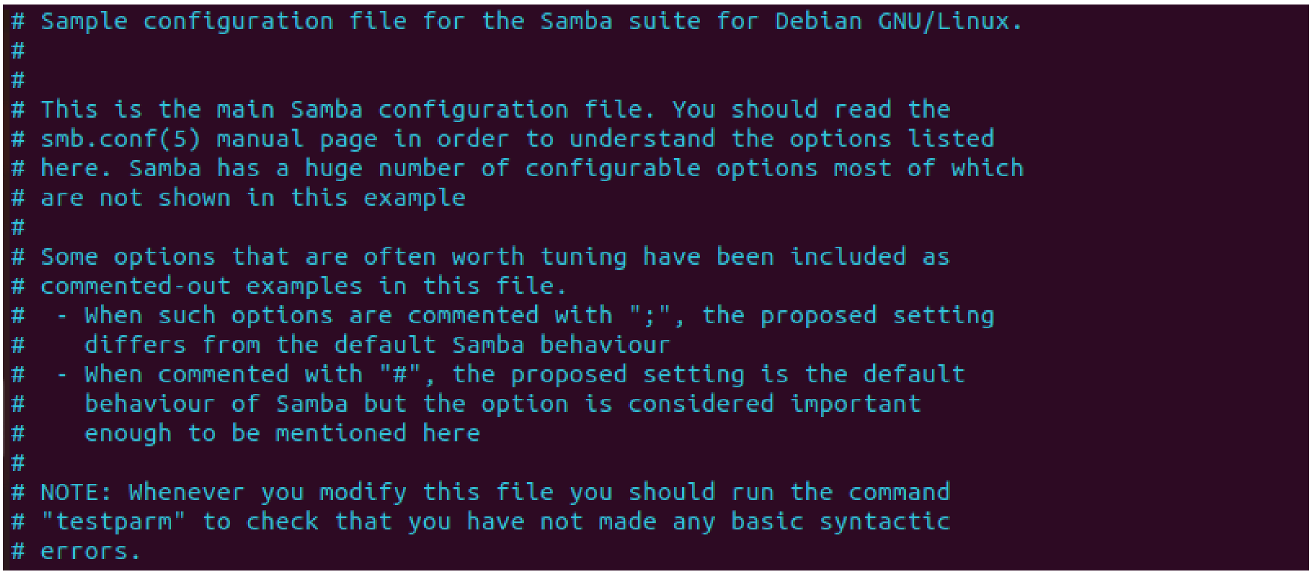
Step 3: Configure Samba's Global Options

Configure Samba by editing the ***smb.conf*** file located in */etc/samba/smb.conf*.

Access the file with Vim:

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

Next, scroll down to the ***Global Settings*** section. Use this section to configure the Samba server's behavior, name, role, and network interfaces.



```
# Sample configuration file for the Samba suite for Debian GNU/Linux.
#
#
# 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 most of which
# are not shown in this example
#
# Some options that are often worth tuning have been included as
# commented-out examples in this file.
#  - When such options are commented with ";", the proposed setting
#    differs from the default Samba behaviour
#  - When commented with "#", the proposed setting is the default
#    behaviour of Samba but the option is considered important
#    enough to be mentioned here
#
# NOTE: Whenever you modify this file you should run the command
# "testparm" to check that you have not made any basic syntactic
# errors.
```

Note: Certain settings in the ***smb.conf*** file are marked as comments. To enable and tweak those settings, uncomment them.

The key parameters to consider are in the following subsections:

Browsing/Identification

The ***Browsing*** subsection contains the **`workgroup`** and **`server string`** parameters:

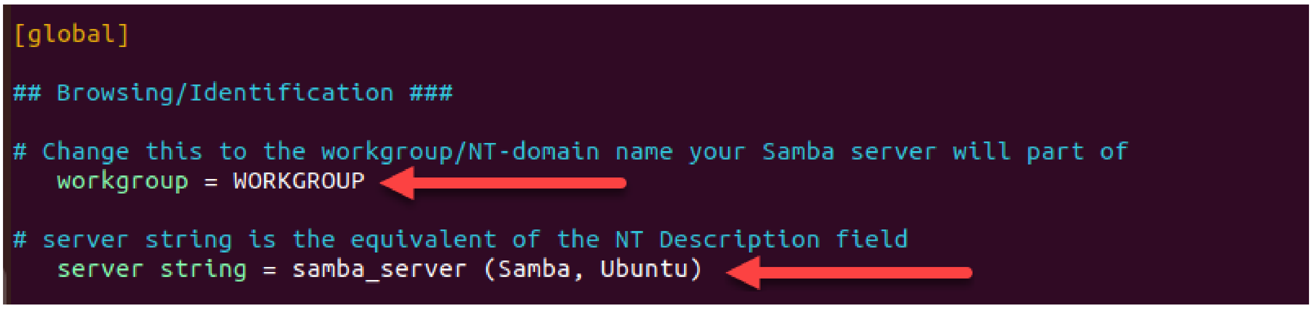
- The **`workgroup`** parameter enables file sharing between a group of computers over a local area network. Ensure the **`workgroup`** settings correspond to the ones on Windows.
- The **`server string`** setting identifies the Samba server. In our example, the server is named *samba_server*.

Note: To set the workgroup settings on Windows 10, open the **Control Panel** and access the **System and Security** settings. The **`workgroup`** parameter is under the *System* section.

To configure the identification settings, uncomment the **`workgroup`** and **`server string`** parameters and add these values:

```
workgroup = WORKGROUP
```

```
server string = samba_server
```

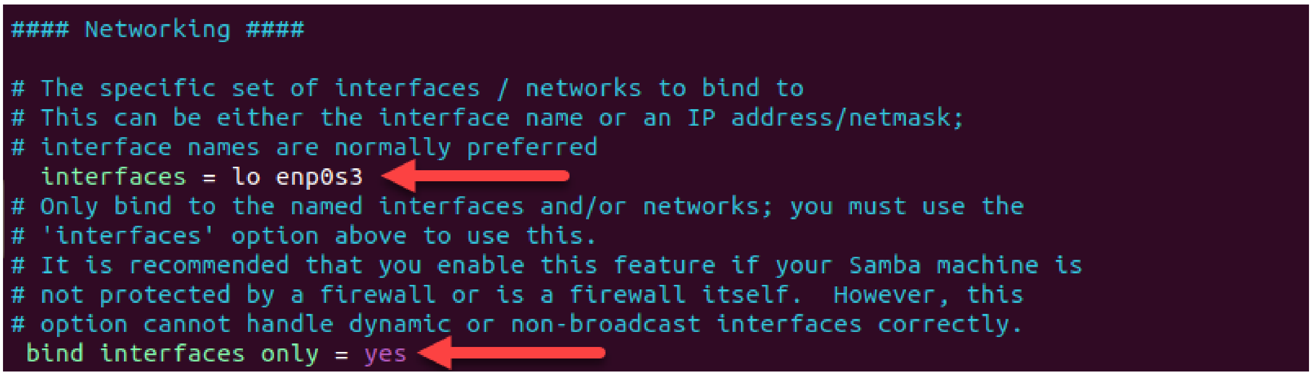


```
[global]

## Browsing/Identification ###

# Change this to the workgroup/NT-domain name your Samba server will part of
   workgroup = WORKGROUP

# server string is the equivalent of the NT Description field
   server string = samba_server (Samba, Ubuntu)
```

Networking

Use the ***Networking*** subsection to configure network interfaces that Samba binds to. ***Networking*** contains two parameters:

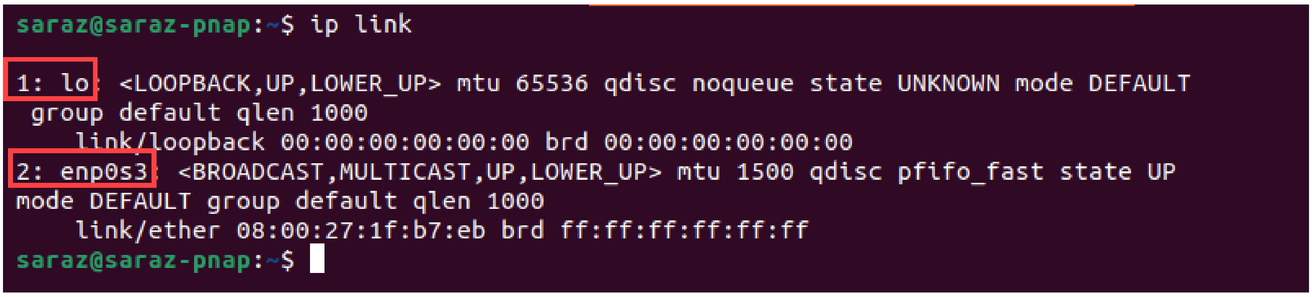
- The first parameter, **`interfaces`**, sets the network interface for Samba to bind to.
- The second parameter, **`bind interfaces only`**, ensures that Samba only binds to the interfaces listed in the file. The parameter should always be set to **`yes`**.



```
#### Networking ####

# The specific set of interfaces / networks to bind to
# This can be either the interface name or an IP address/netmask;
# interface names are normally preferred
  interfaces = lo enp0s3
# Only bind to the named interfaces and/or networks; you must use the
# 'interfaces' option above to use this.
# It is recommended that you enable this feature if your Samba machine is
# not protected by a firewall or is a firewall itself.  However, this
# option cannot handle dynamic or non-broadcast interfaces correctly.
 bind interfaces only = yes
```

To set the **`interfaces`** parameter, first check the available interfaces with the ip command:

```
ip link
```



```
saraz@saraz-pnap:~$ ip link

1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT
 group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
mode DEFAULT group default qlen 1000
    link/ether 08:00:27:1f:b7:eb brd ff:ff:ff:ff:ff:ff
saraz@saraz-pnap:~$
```

The example output indicates Samba binds to two interfaces: ***lo***, the loopback interface, and ***enp0s3***, the network interface.

For example, in this case, the settings are:

```
interfaces = lo enp0s3
```

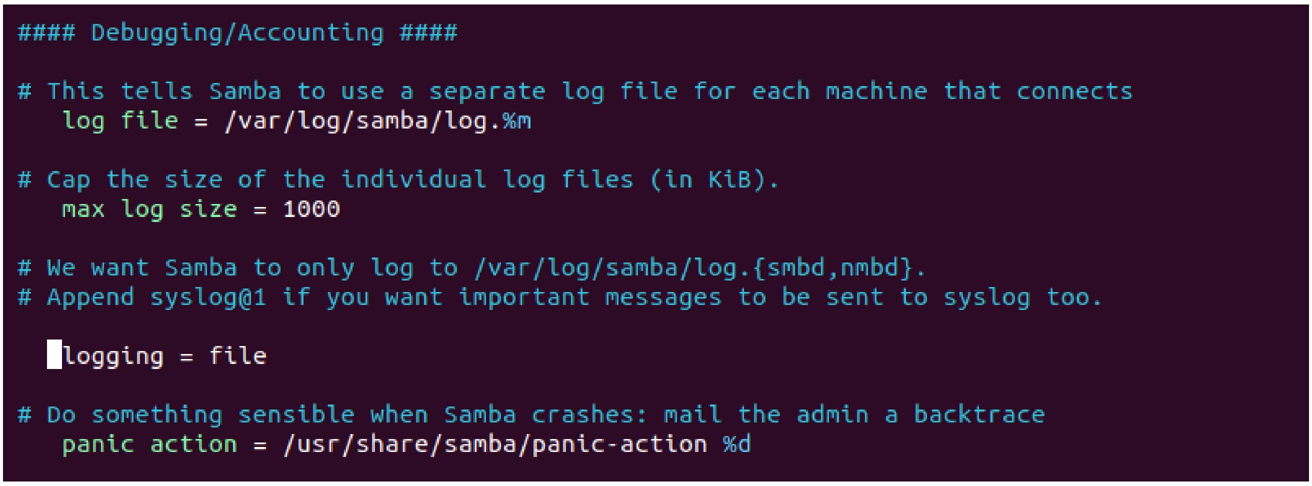
```
bind interfaces only = yes
```

Note: The network interfaces Samba binds to may differ from one machine to another.

Debugging

The ***Debugging*** subsection has four parameters. Set them as follows:

```
log file = /var/log/samba/log.%m
max log size = 1000
logging = file
panic action = /usr/share/samba/panic-action %d
```



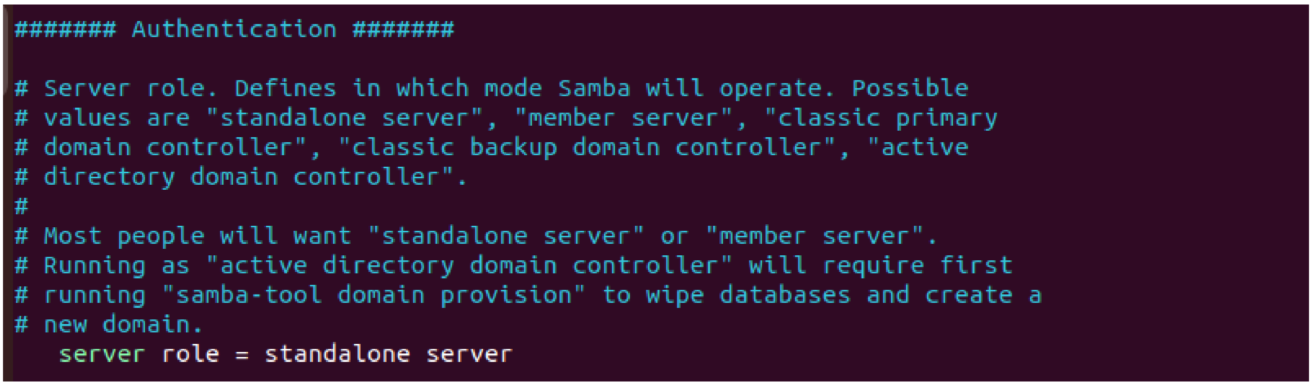
```
#### Debugging/Accounting ####

# This tells Samba to use a separate log file for each machine that connects
   log file = /var/log/samba/log.%m

# Cap the size of the individual log files (in KiB).
   max log size = 1000

# We want Samba to only log to /var/log/samba/log.{smbd,nmbd}.
# Append syslog@1 if you want important messages to be sent to syslog too.

   logging = file

# Do something sensible when Samba crashes: mail the admin a backtrace
   panic action = /usr/share/samba/panic-action %d
```

Authentication, Domain, and Misc

The most significant ***Authentication*** parameter is `server role`. This parameter determines the server type for Samba.

1. Set Samba as a standalone server:

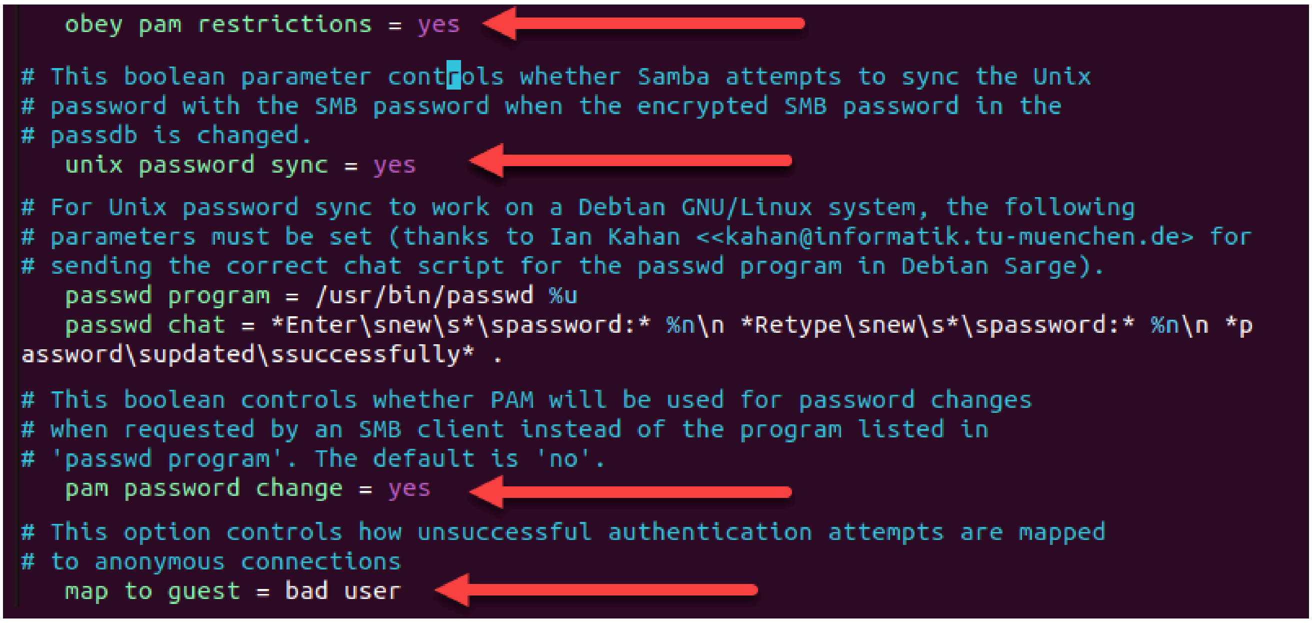
```
server role = standalone server
```



```
####### Authentication #######

# Server role. Defines in which mode Samba will operate. Possible
# values are "standalone server", "member server", "classic primary
# domain controller", "classic backup domain controller", "active
# directory domain controller".
#
# Most people will want "standalone server" or "member server".
# Running as "active directory domain controller" will require first
# running "samba-tool domain provision" to wipe databases and create a
# new domain.
   server role = standalone server
```

The following is an extensive list of other authentication settings:

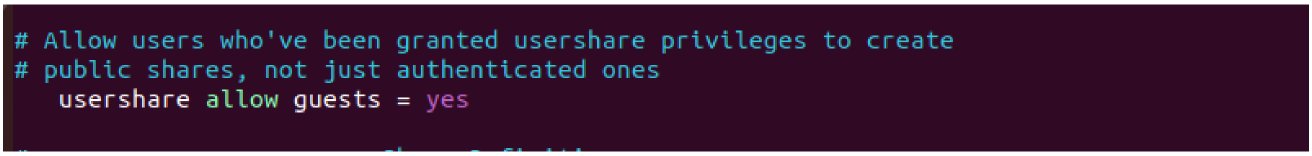
```
obey pam restrictions = yes

unix password sync = yes

passwd program = /usr/bin/passwd

passwd chat = *Enter\snew\s*\spassword:* %n\n *Retype\snew\s*\spassword:* %n\n *pass
word\supdated\ssuccessfully* .

pam password change = yes

map to guest = bad user
```



```
   obey pam restrictions = yes

# This boolean parameter controls whether Samba attempts to sync the Unix
# password with the SMB password when the encrypted SMB password in the
# passdb is changed.
   unix password sync = yes

# For Unix password sync to work on a Debian GNU/Linux system, the following
# parameters must be set (thanks to Ian Kahan <<kahan@informatik.tu-muenchen.de> for
# sending the correct chat script for the passwd program in Debian Sarge).
   passwd program = /usr/bin/passwd %u
   passwd chat = *Enter\snew\s*\spassword:* %n\n *Retype\snew\s*\spassword:* %n\n *p
assword\supdated\ssuccessfully* .

# This boolean controls whether PAM will be used for password changes
# when requested by an SMB client instead of the program listed in
# 'passwd program'. The default is 'no'.
   pam password change = yes

# This option controls how unsuccessful authentication attempts are mapped
# to anonymous connections
   map to guest = bad user
```

2. Do not change any settings in the **Domain** subsection, but scroll down to **Misc** and set the following:

```
usershare allow guests = yes
```

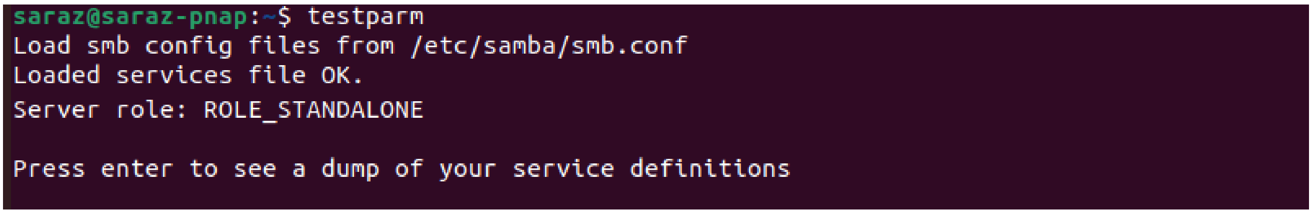


```
# Allow users who've been granted usershare privileges to create
# public shares, not just authenticated ones
   usershare allow guests = yes
```

Keep all other ***Global Settings*** unchanged.

3. Save and exit the file and run the Samba utility **`testparm`** to check for syntax errors:

```
testparm
```



```
saraz@saraz-pnap:~$ testparm
Load smb config files from /etc/samba/smb.conf
Loaded services file OK.
Server role: ROLE_STANDALONE

Press enter to see a dump of your service definitions
```

The output shows the ***Loaded services file OK*** message, which signifies no syntax errors. With ***Global Settings*** configured, the Samba server is ready to use.

Still, not configuring the users and the directory limits the Samba functionality.

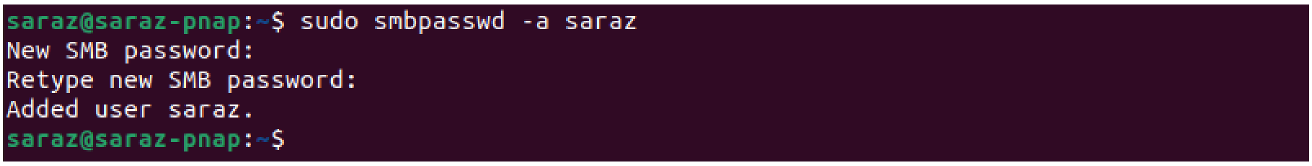
Step 4: Set Up a User Account

1. To create a user account, set a username and password with:

```
sudo smbpasswd -a username
```

Note that the username should belong to a system user. For instance, in this example, the system account on the Ubuntu system is **saraz**. Hence, the username is the same:

```
sudo smbpasswd -a saraz
```



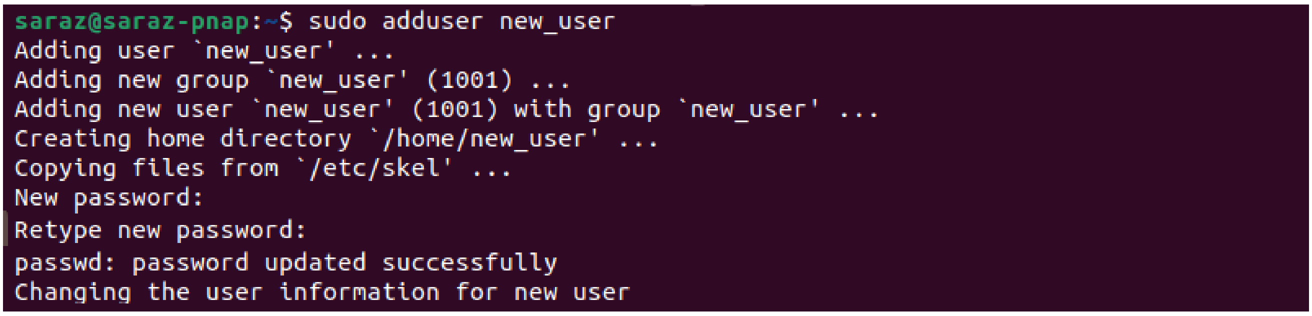
```
saraz@saraz-pnap:~$ sudo smbpasswd -a saraz
New SMB password:
Retype new SMB password:
Added user saraz.
saraz@saraz-pnap:~$
```

2. To add a new user to Samba and the system, use adduser:

```
sudo adduser username
```

For instance, add *new_user* to the system with:

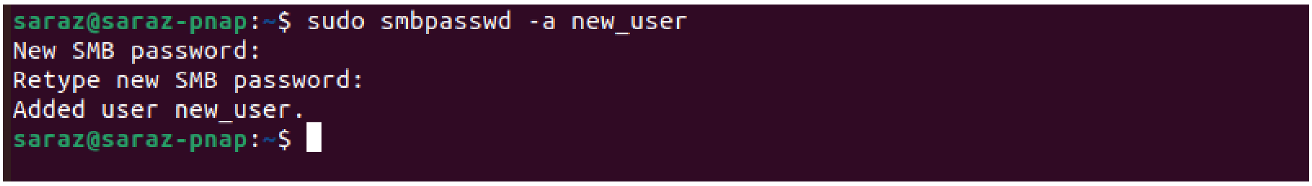
```
sudo adduser new_user
```



```
saraz@saraz-pnap:~$ sudo adduser new_user
Adding user `new_user' ...
Adding new group `new_user' (1001) ...
Adding new user `new_user' (1001) with group `new_user' ...
Creating home directory `/home/new_user' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for new user
```

3. After entering and confirming the system password for *new_user*, create a new Samba user with:

```
sudo smbpasswd -a new_user
```



```
saraz@saraz-pnap:~$ sudo smbpasswd -a new_user
New SMB password:
Retype new SMB password:
Added user new_user.
saraz@saraz-pnap:~$ 
```

Next, both users need to have read, write and execute access to the *sharing* directory. However, *saraz* has these permissions by default. On the other hand, *new_user* does not.

4. To grant read, write, and execute permissions to the *sharing* directory, run **`setfacl`**:

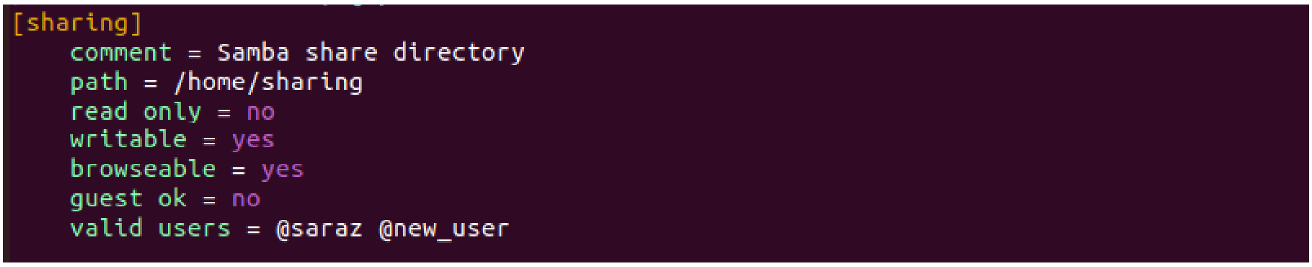
```
sudo setfacl -R -m "u:new_user:rwx" /home/sharing
```

The command doesn't produce any output.

Step 5: Configure Samba Share Directory Settings

1. Access the configuration file once again to add the previously made *sharing* directory. Go to the end of the file and add:

```
[sharing]
comment = Samba share directory
path = /home/sharing
read only = no
writable = yes
browseable = yes
guest ok = no
valid users = @saraz @new_user
```



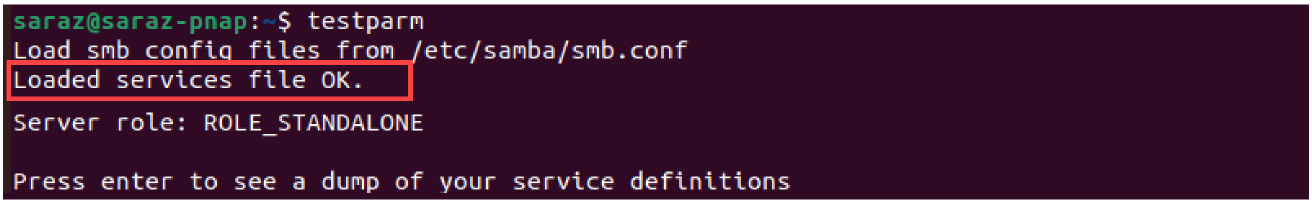
```
[sharing]
    comment = Samba share directory
    path = /home/sharing
    read only = no
    writable = yes
    browseable = yes
    guest ok = no
    valid users = @saraz @new_user
```

Each line grants specific permissions to access the directory. For instance:

- **[sharing}.** Represents the directory name. This is the directory location Windows users see.
- **comment.** Serves as a shared directory description.
- **path.** This parameter specifies the shared directory location. The example uses a directory in */home*, but users can also place the shared files under */samba*.
- **read only.** This parameter allows users to modify the directory and add or change files when set to **no**.
- **writeable.** Grants read and write access when set to **yes**.
- **browseable.** This parameter allows other machines in the network to find the Samba server and Samba share when set to **yes**. Otherwise, users must know the exact Samba server name and type in the path to access the shared directory.
- **guest ok.** When set to **no**, this parameter disables guest access. Users need to enter a username and password to access the shared directory.
- **valid users.** Only the users mentioned have access to the Samba share.

2. Save the changes and exit the file.

3. Rerun **testparm**:



```
saraz@saraz-pnap:~$ testparm
Load smb config files from /etc/samba/smb.conf
Loaded services file OK.
Server role: ROLE_STANDALONE

Press enter to see a dump of your service definitions
```

The output confirms that the Samba is adequately configured. For a more verbose output, hit **enter**:

Step 6: Update the Firewall Rules

To ensure the Linux firewall allows Samba traffic, run:

```
sudo ufw allow samba
```

Step 7: Connect to the Shared Directory

1. Before connecting to the Samba server, restart the services with:

```
sudo systemctl restart smbd
```

The command prints no output.

2. To connect to the shared directory via GUI, access the default file manager and choose the **Other Locations** option:

3. Type the following into the **Enter server address...** box and select **Connect**:

```
smb://ip-address/sharing
```

4. The system asks for a **Username** and **Password**. Provide the requested info and click **Connect** again:

5. This adds the *sharing* directory to the *Windows shares* location:

Conclusion

After reading this tutorial, you now know how to install and configure Samba on Ubuntu.

Next, learn essential Linux commands with this handy Linux commands cheat sheet.

Was this article helpful? Yes No

Sara Zivanov

Sara Zivanov is a technical writer at phoenixNAP who is passionate about making high-tech concepts accessible to everyone. Her experience as a content writer and her background in Engineering and Project Management allows her to streamline complex processes and make them user-friendly through her content.

Next you should read

SysAdmin

How to Install Ubuntu 22.04 LTS Desktop (Jammy Jellyfish)

April 20, 2022

Ubuntu 22.04 LTS is the newest Ubuntu

version with great new features. Follow this guide to install Ubuntu 22.04 in no time...

READ MORE

Backup and Recovery, Databases, SysAdmin

12 Data Integration Tools Reviewed

March 17, 2022

Data integration tools help transport, modify and integrate data into various systems. Learn about the different available tools today and leverage the power of your data...

READ MORE

Networking, Web Servers

Server Operating Systems: Server OS Types & How to Choose

March 10, 2022

A server operating system is software designed to power a server computer. Learn more about server operating systems and read...

READ MORE

Bare Metal Servers, SysAdmin, Web Servers

How to Install a Desktop (GUI) on an Ubuntu Server

August 4, 2022

Want to add a desktop environment after you install Ubuntu server? By default, Ubuntu Server does not include a Graphical User Interface (GUI)...

READ MORE

Sitemap