How to Install and Configure Samba Server in Ubuntu? [4 Steps]

Samba is an open-source software suite providing **file and print sharing services** between **Windows** and **Linux-based systems**. It is gaining wide popularity for its interoperability and being a **versatile solution** for **integrating** Linux and Windows. This tutorial will provide you with a step-by-step walkthrough on how to install and configure Samba in Ubuntu.

Besides file sharing, Samba supports features such as authentication and access control, integrating with cloud storage services and streaming media files. Samba relies on **smb** or **Server Message Block** protocol, which is the **core network sharing protocol** for interprocess communication.

Table of Contents

Key Takeaways

- Learning how to **install** and **configure Samba in Ubuntu 22.04**.
- Knowing about **web-based Samba configuration** process.
- Knowing how to **remove** and **backup** Samba files.

Requirements

- User must have [sudo/root privilege](#).
- User must have a text editor.

Process Flow Chart

[Distro Used Throughout the Tutorial: Ubuntu 22.04.02 LTS]

Step-by-Step Process to Install and Configure Samba

Samba is a popular choice for communication over a network due to its **cross-platform compatibility**, **resource management**, and **fast sharing services**. To avail these benefits, the foremost task is to install Samba and configure it according to your requirements. Follow the steps below and you will be done with the installation process in no time. Now, follow the instructions below and you can install Samba in no time.

Step 1: Installing Samba on Ubuntu

First, you need to install Samba on your Linux distribution, in this regard, I am using **Ubuntu 22.04**. This method ensures updates of all existing packages and verifies Samba installation.

Steps to Follow >

❶ Open your **Ubuntu** terminal.

❷ Type the following command to update all packages in your Linux system.

```
sudo apt update
```

EXPLANATION
• sudo: Permits a user with administrative privileges to execute commands. • **apt update**: Updates all outdated packages and dependencies on your system.

```
ayesha@ubuntu:~$ sudo apt update
[sudo] password for ayesha:
Hit:1 http://bd.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:3 http://bd.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:4 http://bd.archive.ubuntu.com/ubuntu jammy-backports InRelease [108 kB]
Get:5 http://bd.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [579 kB]
Get:6 http://security.ubuntu.com/ubuntu jammy-security/main amd64 DEP-11 Metadata [41.6 kB]
Get:7 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 DEP-11 Metadata [18.5 kB]
Get:8 http://bd.archive.ubuntu.com/ubuntu jammy-updates/main amd64 DEP-11 Metadata [102 kB]
Get:9 http://bd.archive.ubuntu.com/ubuntu jammy-updates/universe i386 Packages [609 kB]
Get:10 http://bd.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [887 kB]
Get:11 http://bd.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 DEP-11 Metadata [269 kB]
Get:12 http://bd.archive.ubuntu.com/ubuntu jammy-updates/multiverse i386 Packages [3,884 B]
Get:13 http://bd.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 Packages [35.3 kB]
Get:14 http://bd.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 DEP-11 Metadata [940 B]
Get:15 http://bd.archive.ubuntu.com/ubuntu jammy-backports/main amd64 DEP-11 Metadata [7,976 B]
Get:16 http://bd.archive.ubuntu.com/ubuntu jammy-backports/universe amd64 DEP-11 Metadata [12.9 kB]
Fetched 2,903 kB in 7s (391 kB/s)
```

❸ Install Samba with the following command.

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

EXPLANATION

- **sudo:** Permits a user with administrative privileges to execute commands.
- **apt install**: Specifies that the install task needs to be performed.
- **samba**: Name of an open-source software package, Samba for file sharing.
- **-y**: Automatically answers 'yes' to any confirmation message during installation.

```
ayesha@ubuntu:~$ sudo apt install samba -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
samba is already the newest version (2:4.15.13+dfsg-0ubuntu1.1).
0 upgraded, 0 newly installed, 0 to remove and 108 not upgraded.
```

❹ An easy way to **verify the installation** of Samba is to **locate** it within Linux system. The command to do this is:

```
whereis samba
```

EXPLANATION

- **whereis**: Looks for the location of the installed package.

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

location of samba installed

The folder that contains the installed Samba is shown in the above image.

❺ Next type the following command to instruct the **UFW firewall** to open the necessary port(s) or **allow network traffic** for the Samba service.

```
sudo ufw allow samba
```

EXPLANATION

- ufw: Uncomplicated Firewall (ufw) is a command line tool for managing firewall configurations on Ubuntu.
- **allow**: Command option that allows incoming network traffic for Samba service.
- **samba**: Name of an open-source software package, Samba for file sharing.

```
ayesha@ubuntu:~$ sudo ufw allow samba
Skipping adding existing rule
Skipping adding existing rule (v6)
```

The command has now granted access to the Samba service through the firewall.

Step 2: Create a Directory for Sharing Files

Now you have to create a shared directory in the Linux system that can be easily accessible from both Linux and Windows.

Steps to follow >

❶ Type the following command to create a directory for Samba sharing. I will create a folder, **Sambashare** in Desktop for file sharing.

```
mkdir /home/ayesha/Desktop/Sambashare
```

EXPLANATION

- mkdir: Creates a new directory or folder
- **/home/ayesha/Desktop/Sambashare**: Absolute path of the location of the folder, Sambashare.

Step 3: Add User to Samba

You have to add a user who will have access to the shared folder.

Steps to follow >

❶ To **add a new Samba user** and **assign a password** to that user, run the following command.

```
sudo smbpasswd  -a  ayesha
```

EXPLANATION

- **smbpasswd**: Sets password for Samba users.
- **-a**: Allows the command to add a new Samba user.
- **ayesha**: User name.

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

The image shows I had to enter a password to add user **ayesha**.

Step 4: Configure Samba Settings for Shared Folder

This step involves **customizing settings and permissions** of the shared folder, **Sambashare** in the Samba configuration file.

Steps to follow>

❶ Now run the command below to open and **edit** the configuration file.

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

EXPLANATION

- nano: Text editor
- **/etc/samba/smb.conf**: Configuration file of Samba to customize settings such as user authentication, shares, and security.
- **smb.conf**: Configuration file. smb stands for Sever Message Block Configuration.

```
ayesha@ubuntu:~$ sudo nano /etc/samba/smb.conf
[sudo] password for ayesha:
```

Provide your password and this command will open an editor file for modifying configuration settings.

❷ Insert these lines at the end of the configuration file. Remember to write the name of your shared folder inside the braces. I wrote **Sambashare** in this regard.

```
[Sambashare]
      path = /home/ayesha/Desktop/Sambashare
      available = yes
      valid users = ayesha
      read only = no
      writeable=yes
      browseable=yes
      public = yes
```

EXPLANATION

- **sambashare**: Name of the folder whose contents you want to share.
- **path**: Absolute path of the folder you want to share.
- **available**: Specifies whether the file is available for sharing or not.
- **valid user**: User name.

- **read-only**: Specifies whether the file is available for reading or not.
- **writeable**: Specifies whether the folder can be written to or not.
- **browseable**: Specifies whether the shared file is visible while browsing the network or not.
- **public**: Specifies whether the shared file is accessible to all users or not.

❸ Press **CTRL+O** and **CTRL+X** to save the changes and exit the editor respectively.

```
[Sambashare]

        path = /home/ayesha/Desktop/Sambashare
        available = yes
        valid users = ayesha
        read only = no
        writeable=yes
        browseable=yes
        public = yes
```

❹ Restart Samba and enter the password of your currently logged-in user.

```
sudo service smbd restart
```

EXPLANATION

- **service**: Controls services in the Linux system.
- **smbd**: Samba daemon which provides file-sharing services to Windows clients.
- **restart**: Stop running the service and start again.

```
ayesha@ubuntu:~$ sudo service smbd restart
[sudo] password for ayesha:
```

❺ Now to finally use the Samba service for file sharing it is important to **check its status**. The following command will tell you if the service is running, stopped, or facing any issues.

```
systemctl status smbd
```

EXPLANATION

- **systemctl**: Manages system services, along with starting, stopping, and checking the status of services.
- **status**: Displays the current status of a service.
- **smbd**: Samba daemon which provides file-sharing services to Windows clients.

```
ayesha@ubuntu:~$ systemctl status smbd
● smbd.service - Samba SMB Daemon
     Loaded: loaded (/lib/systemd/system/smbd.service; enabled;
vendor pres>
     Active: active (running) since Tue 2023-05-16 15:09:41 +06;
 30min ago
       Docs: man:smbd(8)
             man:samba(7)
             man:smb.conf(5)
    Process: 6759 ExecStartPre=/usr/share/samba/update-apparmor-
samba-profi>
   Main PID: 6763 (smbd)
```

The image above shows that the Samba service is active and running as expected.

Complementary Information

You have now successfully installed and configured Samba on your Linux system. However, the file sharing process, managing Samba from a remote device, or tackling security concerns are still unclear. Let's explore the instructions below to have a smooth sharing experience.

A. Sharing files using Samba on Ubuntu

To finally serve the purpose of Samba i.e. to facilitate file sharing, you have the **flexibility to create files in either Linux or Windows** and **access them seamlessly from both systems**.

Step 1: Establish a Connection between Linux and Windows

To help Linux and Windows to communicate with each other, you need to install the required **file sharing packages and protocols**.

Steps to follow>

❶ Run the command below to install **client and utils package** on Samba.

```
sudo apt install cifs-utils samba client -y
```

EXPLANATION

- **cifs-utils**: Package that supports mounting CIFS/SMB file system. CIFS is the core file sharing protocol used by Samba.
- **samba client**: Package containing necessary client tools for Samba shares and other Samba-related operations.
- **-y**: Automatically answers 'yes' to any confirmation message during installation.

```
ayesha@ubuntu:~$ sudo apt install cifs-utils samba client -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

❷ To access files in Windows, you need to obtain the **IP address** of the Linux system. Run the command below to find it.

```
hostname -I
```

EXPLANATION
• hostname -I: Obtain IP address of the current Linux system.

```
ayesha@ubuntu:~$ hostname -I
192.168.153.128
```

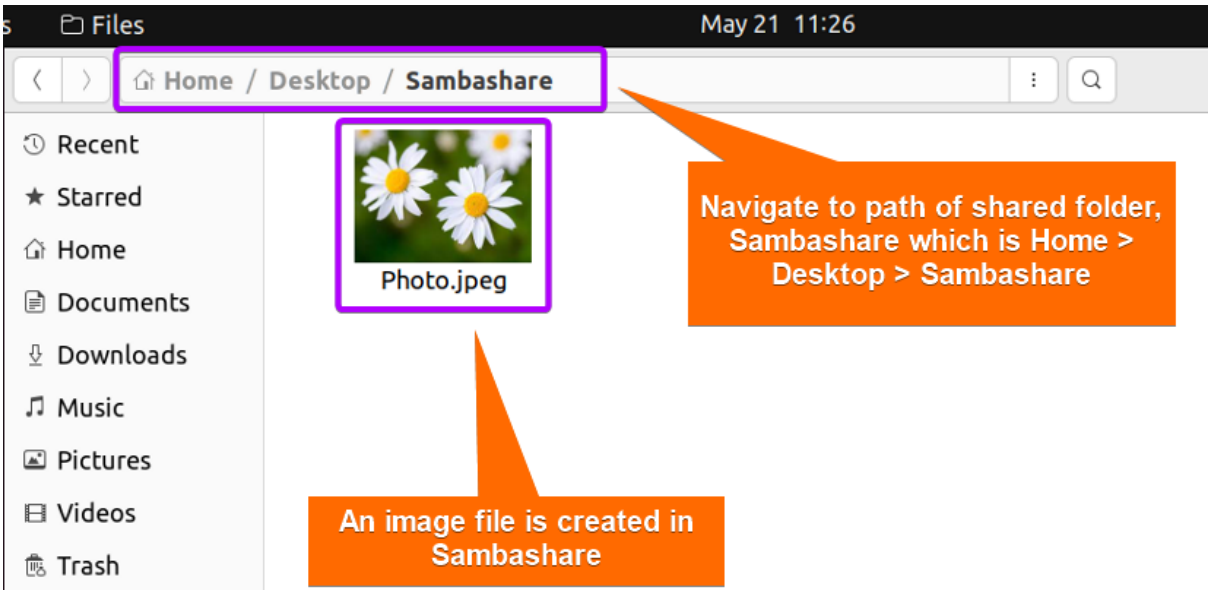
You can see from the image above, that the IP address of my Linux system is **192.168.153.128**.

Step 2: Create files in Ubuntu and Access them from Windows

In this step, I will **save an image** called **Photo.jpeg in Linux** and attempt to **access it from Windows**.

Steps to follow>

❶ In Linux, navigate to the shared folder, **Sambashare** by following the path Files>Home>Desktop>Sambashare and save the image file there.

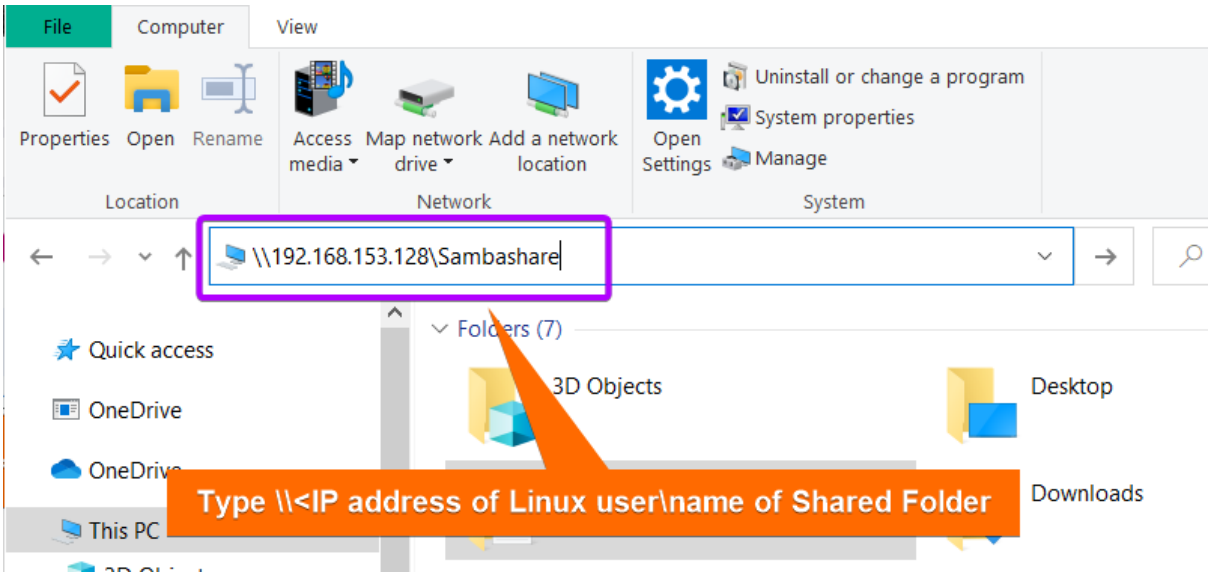


❷ Then go to Windows and type the IP address of Linux system and the name of Sambashare in the address bar.

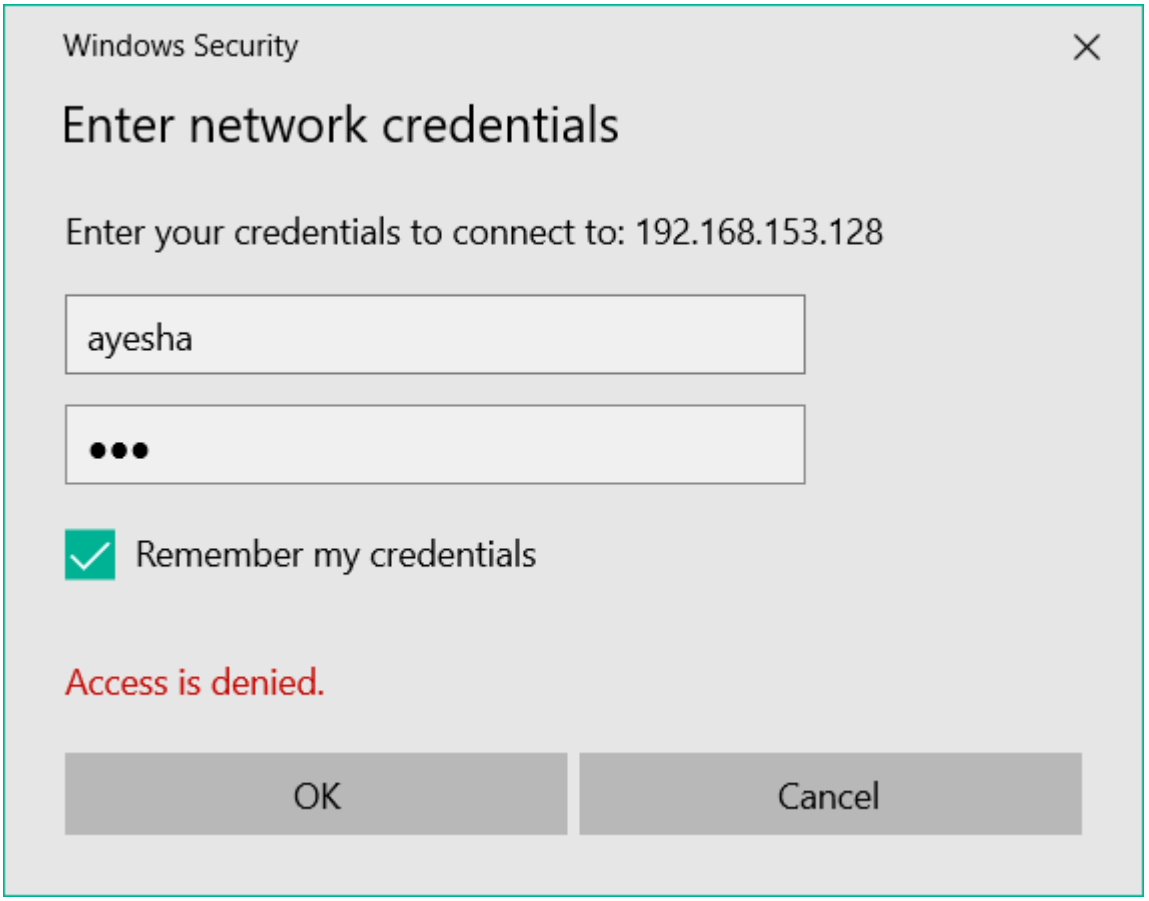
```
\\192.168.153.128\Sambashare
```

EXPLANATION
• **192.168.153.128:** IP address of my Linux system.

- **Webshare:** Name of the folder used for sharing files.

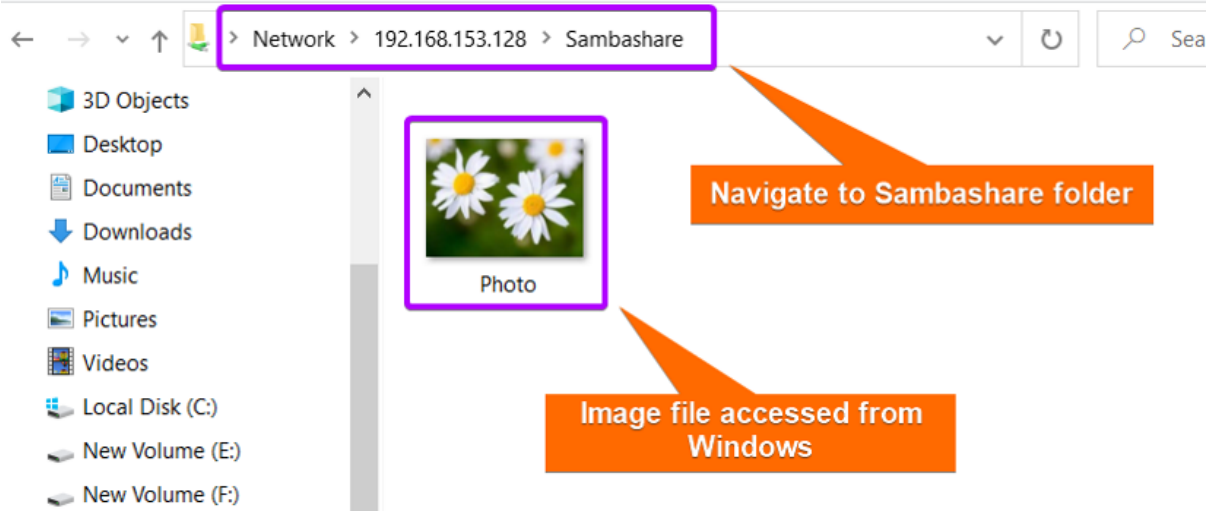


❸ A dialogue box like the image below will appear. Put in your currently logged-in Ubuntu user information.



Don't worry if "Access is denied" shows up. It will appear until you have entered your user information correctly.

❹ You have now successfully **accessed Sambashare folder from Windows** and can view its contents.



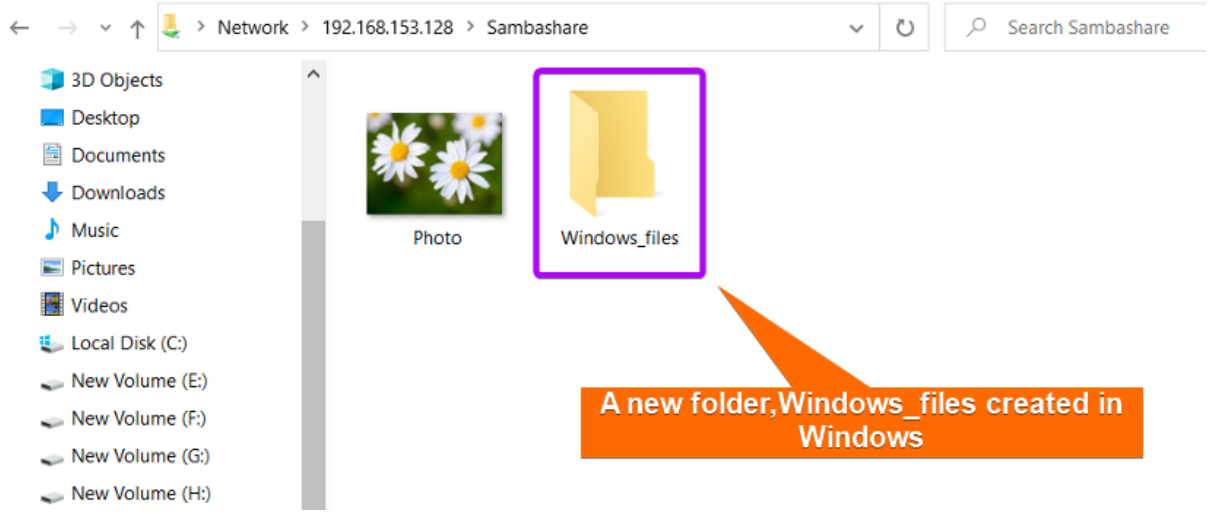
You can see that the image file is now accessible from the Windows system.

Step 3: Create Files in Windows and Access them from Ubuntu

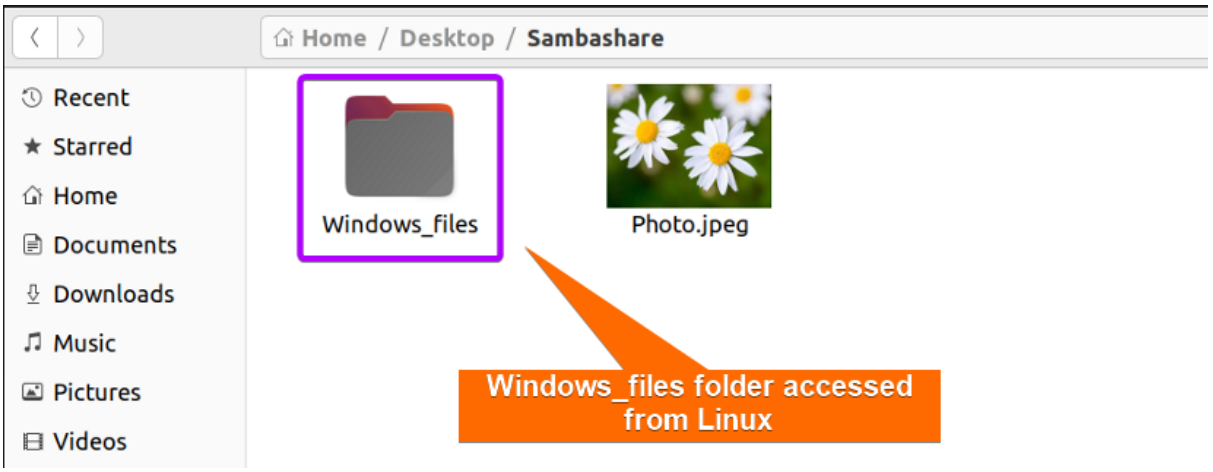
I will demonstrate to you whether I can share any file from Windows. In this case, I will **create a folder, Windows_files in Sambashare in Windows** and try to **access it from Linux**.

Steps to follow>

❶ Go to Windows and create a folder, Windows_files.



❷ Then open the Sambashare folder from Linux.



The image above shows that the Windows_files folder is now accessible from Linux.

B. Web-based Samba Configuration on Ubuntu

You can opt for web-based Samba configuration if you prefer a [graphical user interface(GUI)] or want to **manage Samba remotely from any device using a web browser**. **Webmin** is a powerful tool for managing all aspects of the Linux system. You can easily configure Samba settings with Webmin instead of a [command-line interface (CLI)].

Step 1: Installing Webmin on Ubuntu

The steps below will guide you on how to set up the required environment for installing Webmin in Ubuntu.

Steps to Follow >

❶ Open your **Ubuntu terminal**.

❷ Type the following command to set up the necessary [repositories] **for installing Webmin** on a Linux system by downloading setup-repos.sh script from the Webmin repository on GitHub.

```
sudo curl -o setup-repos.sh
https://raw.githubusercontent.com/webmin/webmin/master/setup-repos.sh
```

EXPLANATION

- **curl**: Retrieves data from or sends data to web or application server by specifying URL.
- **-o setup-repos.sh**: Option -o determines that output of the URL should be saved to a file named 'setup-repos.sh'.
- **https://raw.githubusercontent.com/webmin/webmin/master/setup-repos.sh**: URL of 'setup-repos.sh' in webmin/webmin repository of Github from where data will be fetched.

```
ayesha@ubuntu:~$ sudo curl -o setup-repos.sh https://raw.githubusercontent.com/webmin
/webmin/master/setup-repos.sh
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  4669  100  4669    0     0   8266      0 --:--:-- --:--:-- --:--:--  8263
```

The command has automatically set up the repository and GPG keys to facilitate Webmin installation.

❸ Afterwards, run the following command.

```
sudo sh setup-repos.sh
```

EXPLANATION

- **sh**: Command for 'shell' and executes shell scripts.
- **setup-repos.sh**: Name of the shell script you want to execute.

```
ayesha@ubuntu:~$ sudo sh setup-repos.sh
Setup Webmin official repository? (y/N) y
  Downloading Webmin key ..
  .. done
  Installing Webmin key ..
  .. done
  Setting up Webmin repository ..
  .. done
  Cleaning repository metadata ..
  .. done
  Downloading repository metadata ..
  .. done
Webmin package can now be installed using apt-get install webmin command.
```

The image depicts that the Linux system is now ready to install Webmin package.

❹ Finally your Linux system is ready to **install Webmin** with the following command.

```
sudo apt-get install webmin
```

EXPLANATION

- **sudo**: Permits a user with administrative privileges to execute commands.
- **apt-get install**: Specifies that the install task needs to be performed.
- **webmin:** Name of Webmin package.

```
ayesha@ubuntu:~$ sudo apt-get install webmin
[sudo] password for ayesha:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libalgorithm-c3-perl libauthen-pam-perl
  libb-hooks-endofscope-perl libb-hooks-op-check-perl
  libclass-c3-perl libclass-c3-xs-perl
```

Install Webmin

From the above image, you can see that Webmin is finally installed.

Step 2: Log into Webmin

Now you have to enter your Samba **user credentials** i.e. username and password to log into Webmin and make necessary changes.

Steps to follow >

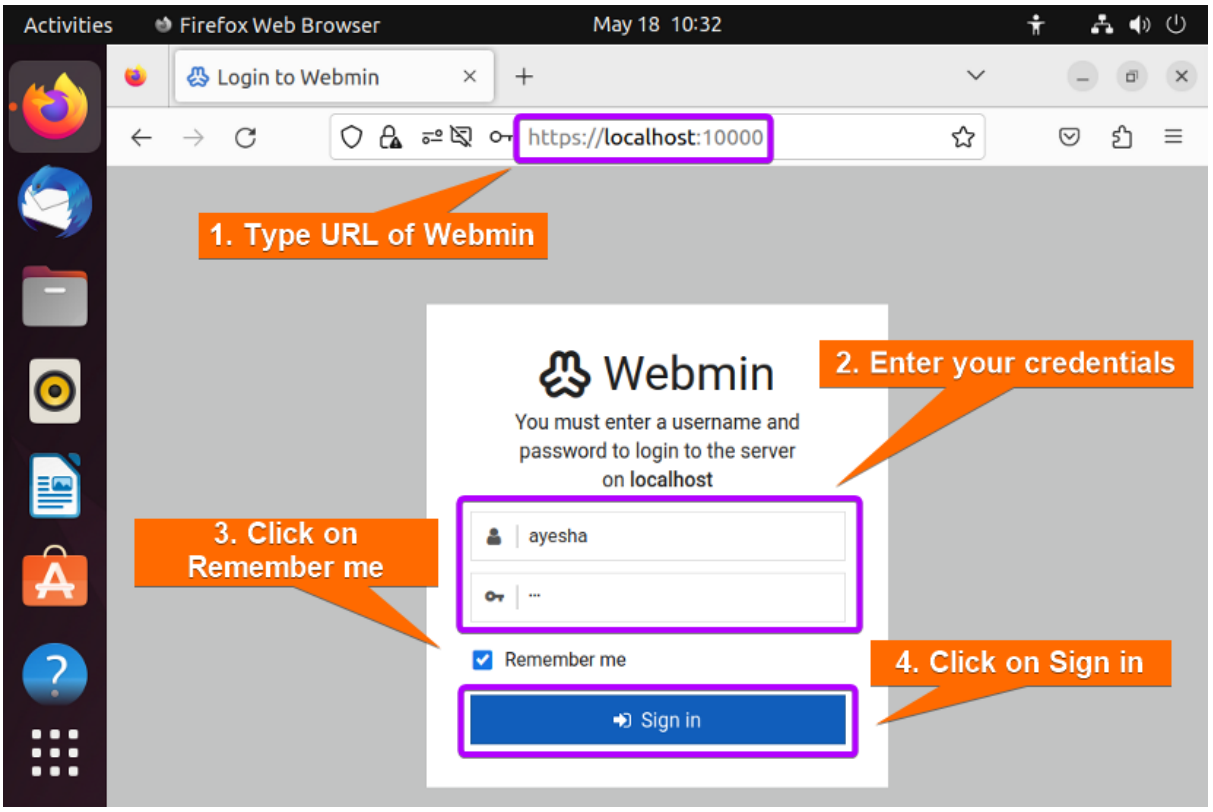
❶ Now open your internet browser and type the **URL of the Webmin** into the browser address field. Enter your username and password or your currently logged-in user.

```
https://localhost:10000
```

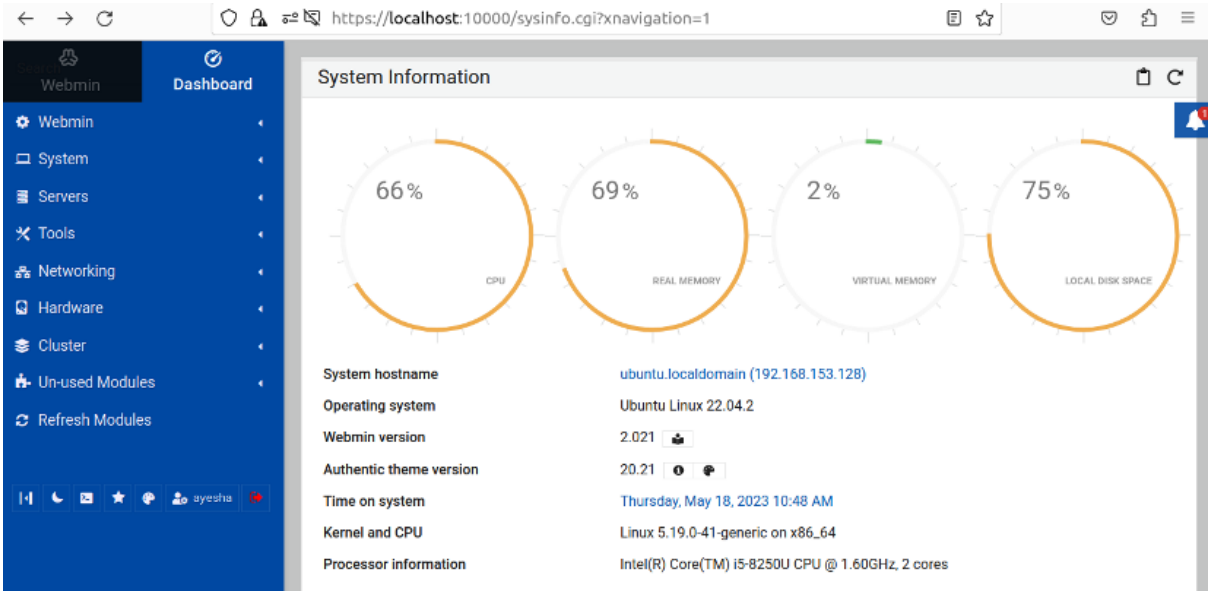
EXPLANATION
• **https:** Stands for Hypertext Transfer Protocol and encrypt data for secure communication. • **localhost:** Hostname referring to the current machine where web browser is running. • **10000:** Port number associated with Webmin.

❷ Click on **Remember me** if you want your user credentials saved.

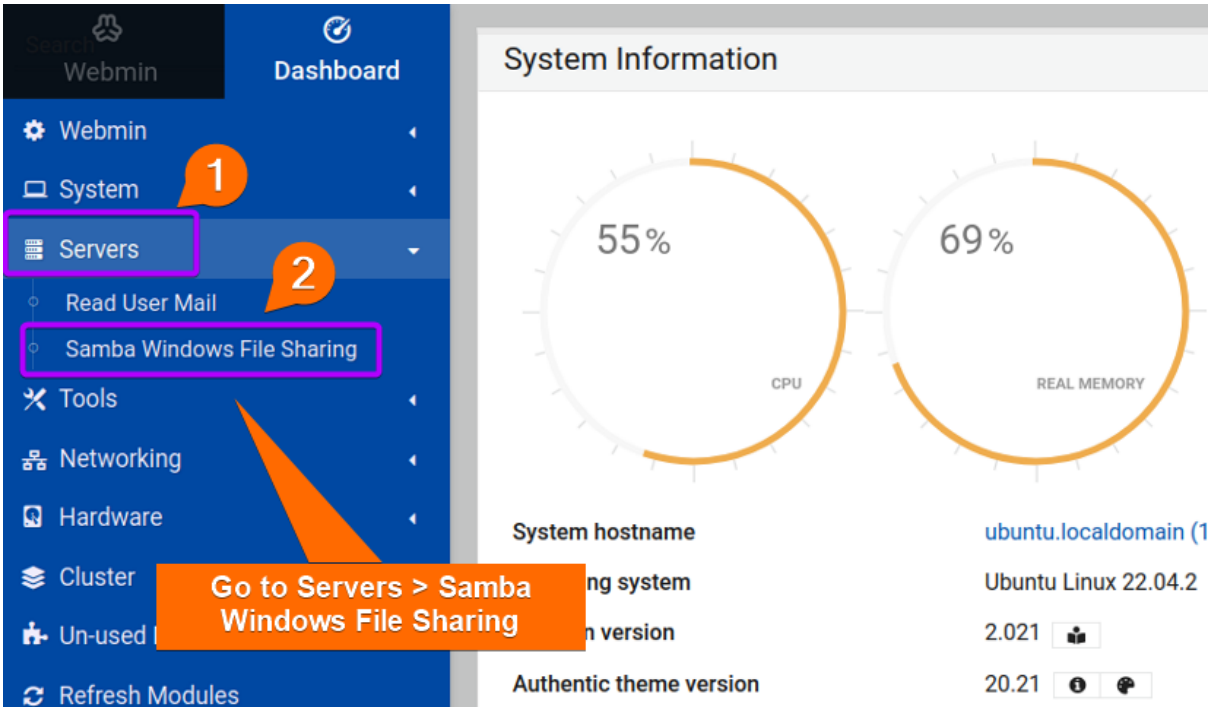
❸ Hit on **Sign in**.



❹ Webmin interface will look like the image below.



❺ Then browse through Webmin > Servers > Samba Windows File Sharing.



The preceding image displays the interface of the Webmin dropdown box.

Step 3: Create a New Shared Folder

You will see that my previously created folder, **Sambashare** is already enlisted in the sharing folders. What if I want to create a **completely new folder and configure it**? Let's see how to do that.

Steps to follow >

❶ Open your Ubuntu terminal.

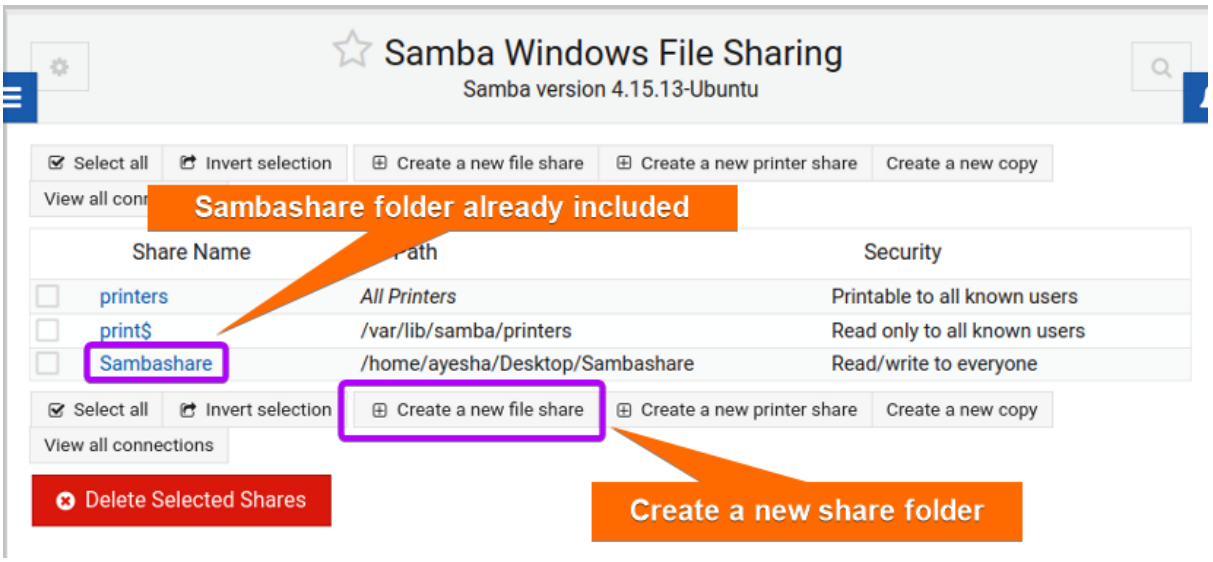
❷ I created a folder called Webshare on the Desktop of my Linux system using the command below.

```
mkdir /home/ayesha/Desktop/Webshare
```

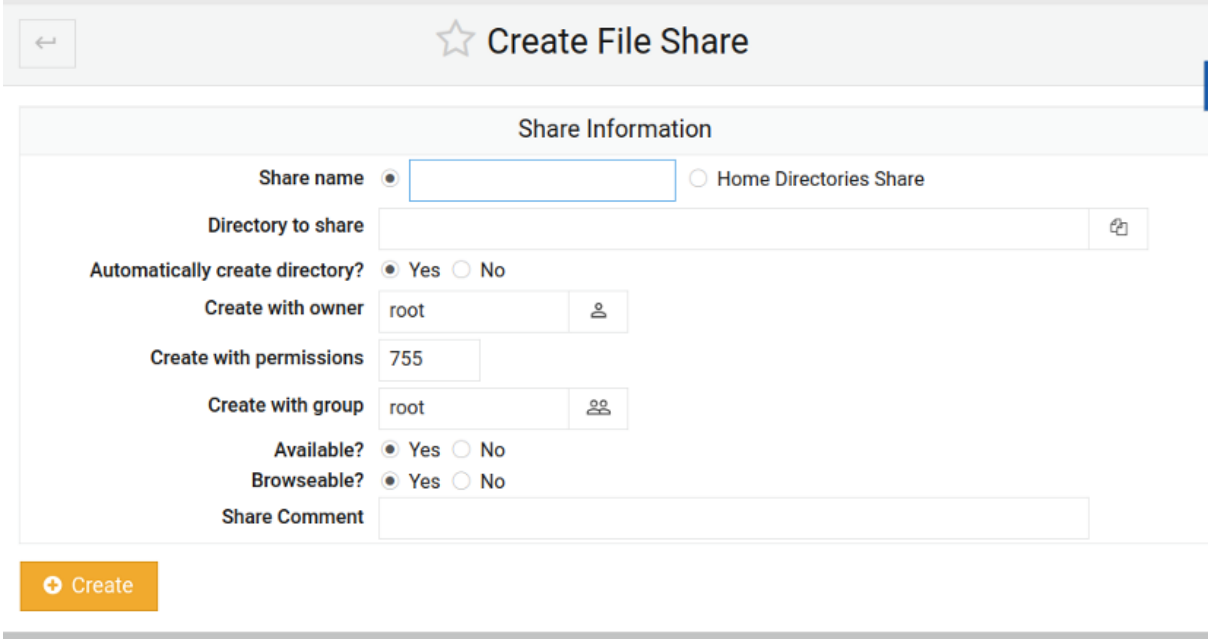
EXPLANATION

- **mkdir:** Creates a new folder or directory.
- **/home/ayesha/Desktop/Webshare**: Absolute path of the newly created Webshare folder.

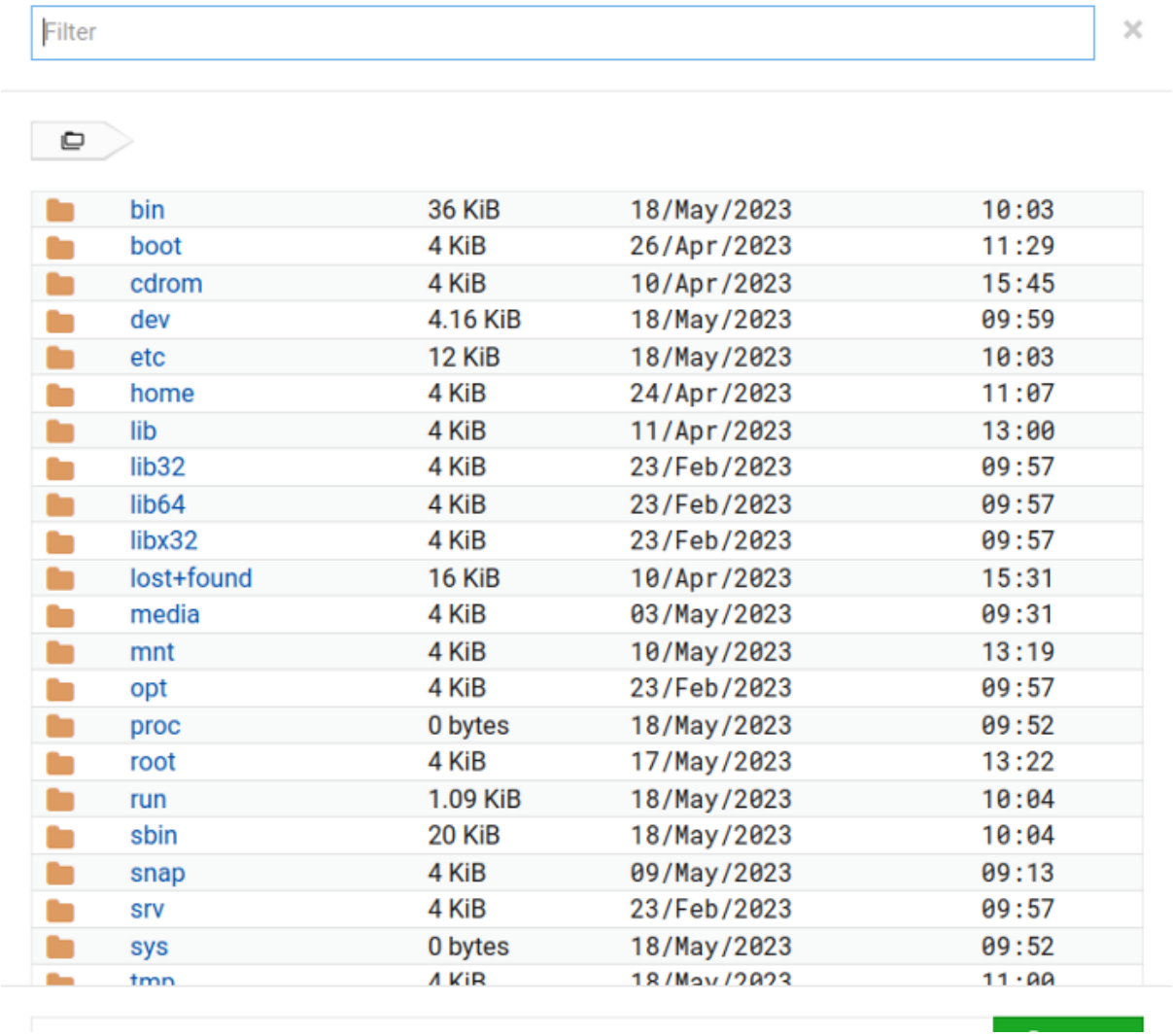
❸ Go back to your internet browser and click on **Create a new file share**.



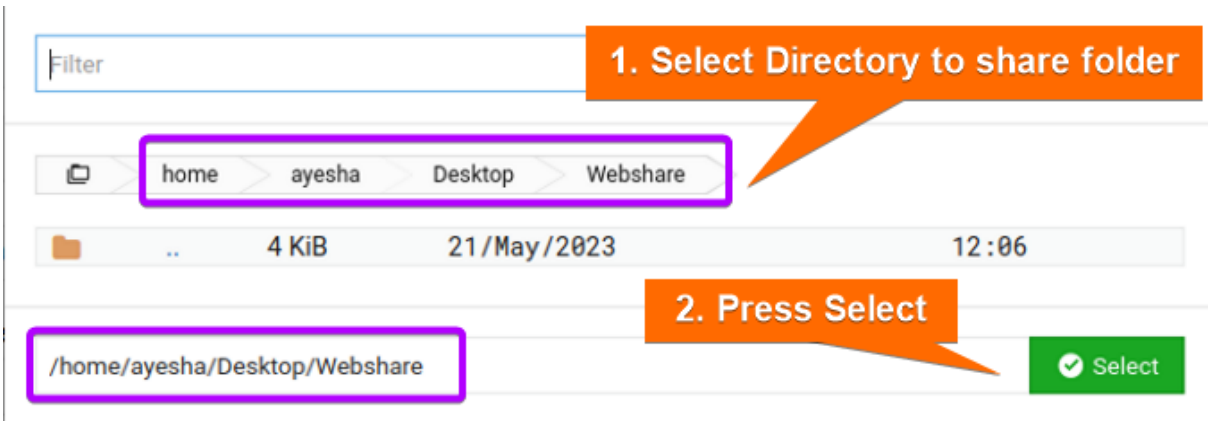
❹ The browser will direct you to the page, Create File Share. You can now enter necessary details for a new folder.



❺ Now click on **Directory to Share** icon.
❻ A list of available directories will appear on the screen.



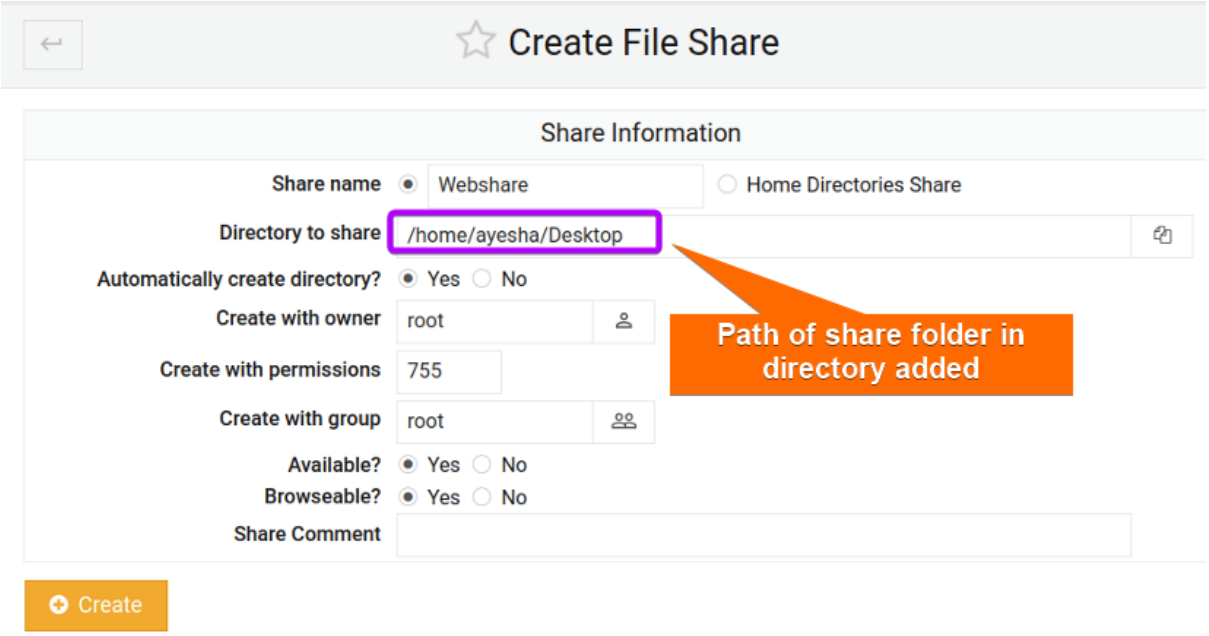
❼ Navigate to the previously created **Webshare** folder in Desktop and press **Select**.



The image displays the entire **path** of the folder, **Webshare**.

❽ You will be directed back to Create File Share page. Type the shared directory name.

❾ Then click on Create button.



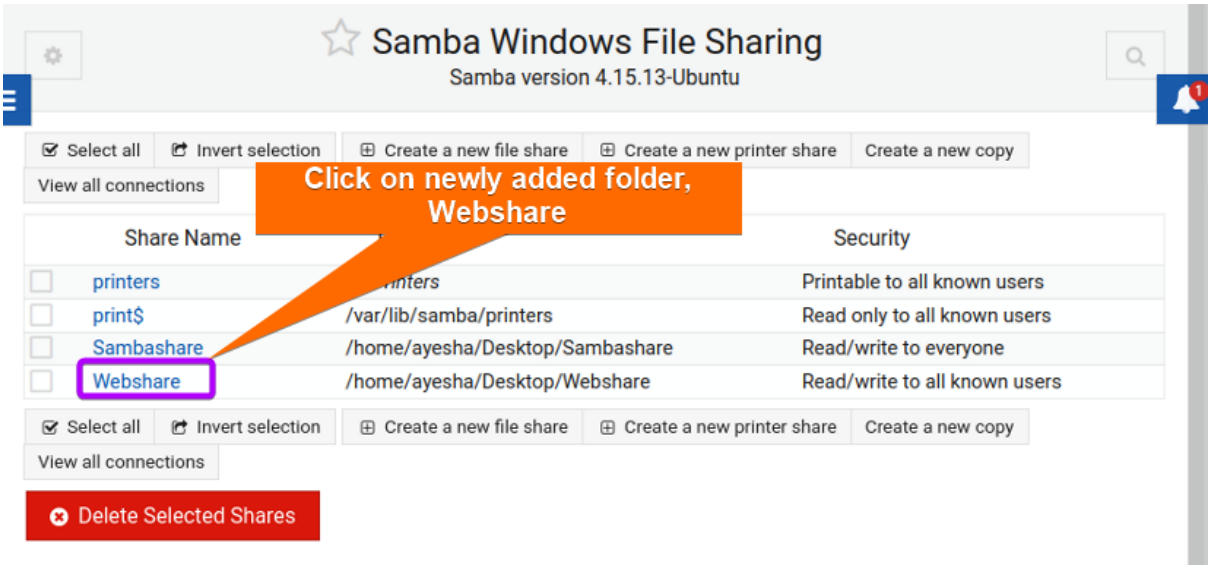
You can see in the picture above that the shared directory is enlisted here.

Step 4: Giving Security and Access Control

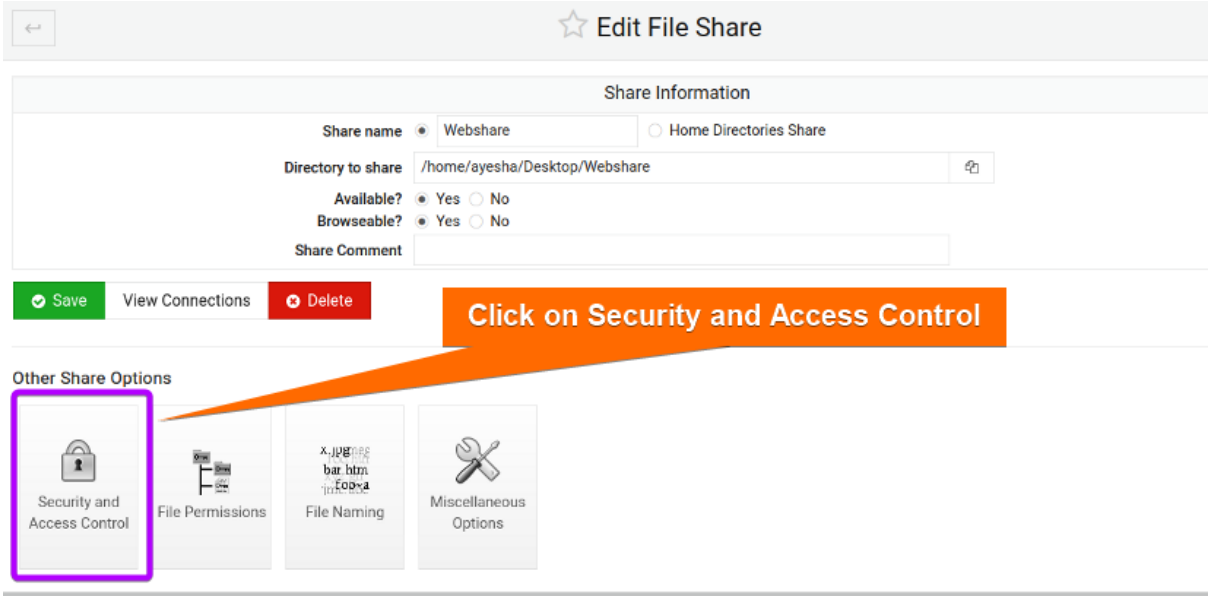
Now you have to decide if your folder will be accessible or not or who can have control over files. To do that you need to bring modifications in **Security and Access Control**.

Steps to follow >

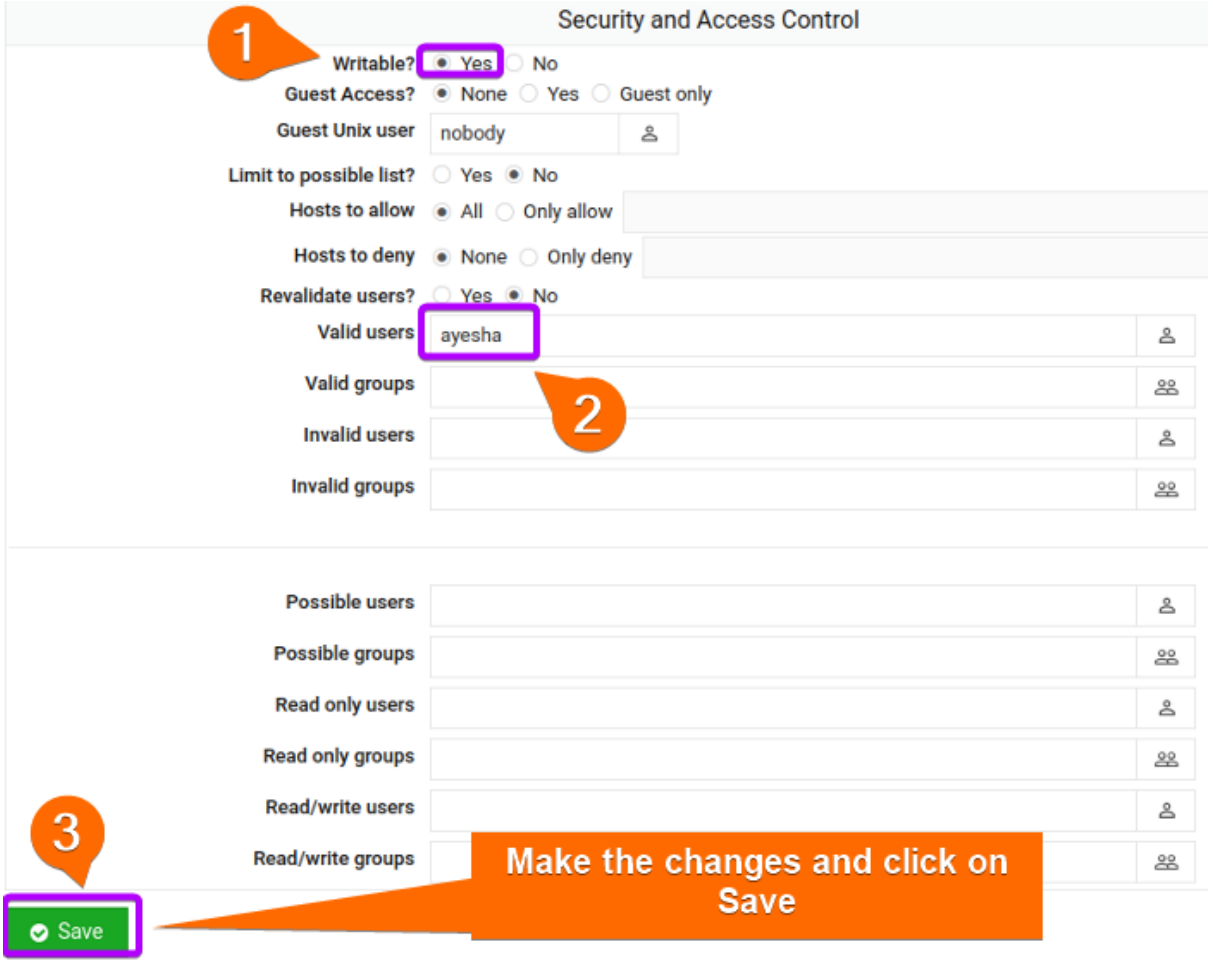
❶ Go to the main interface of Samba Windows File Sharing and click on the newly added folder, **Webshare**.



❷ Then click on Security and Access Control.



❸ Select Yes to Writeable and add your user name who will have access. Press on Save afterward.

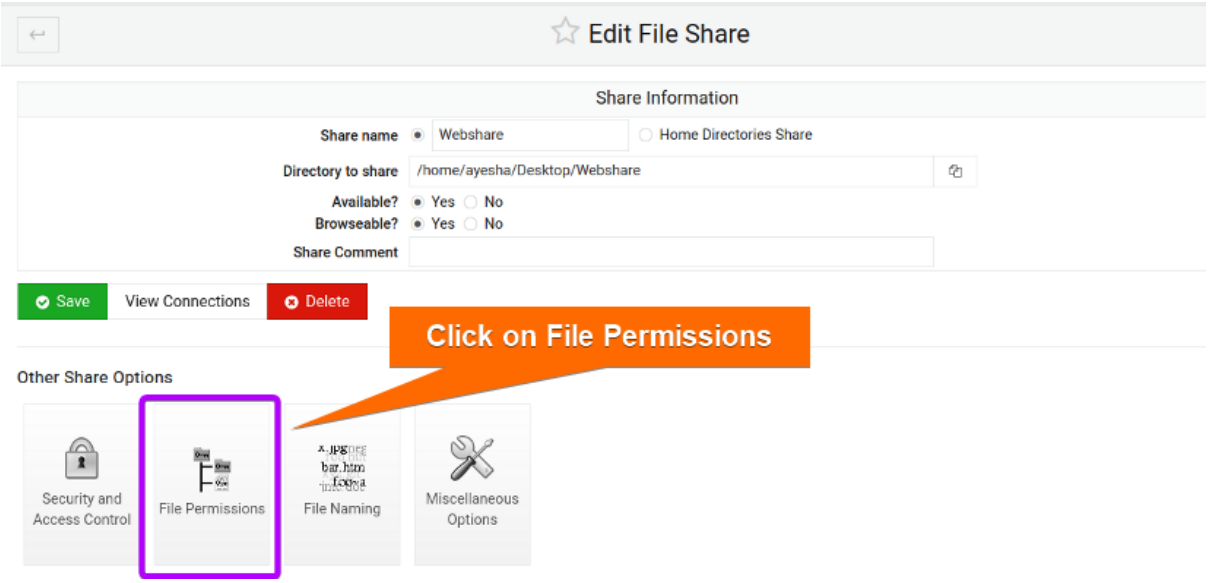


Step 5: Giving File Permissions

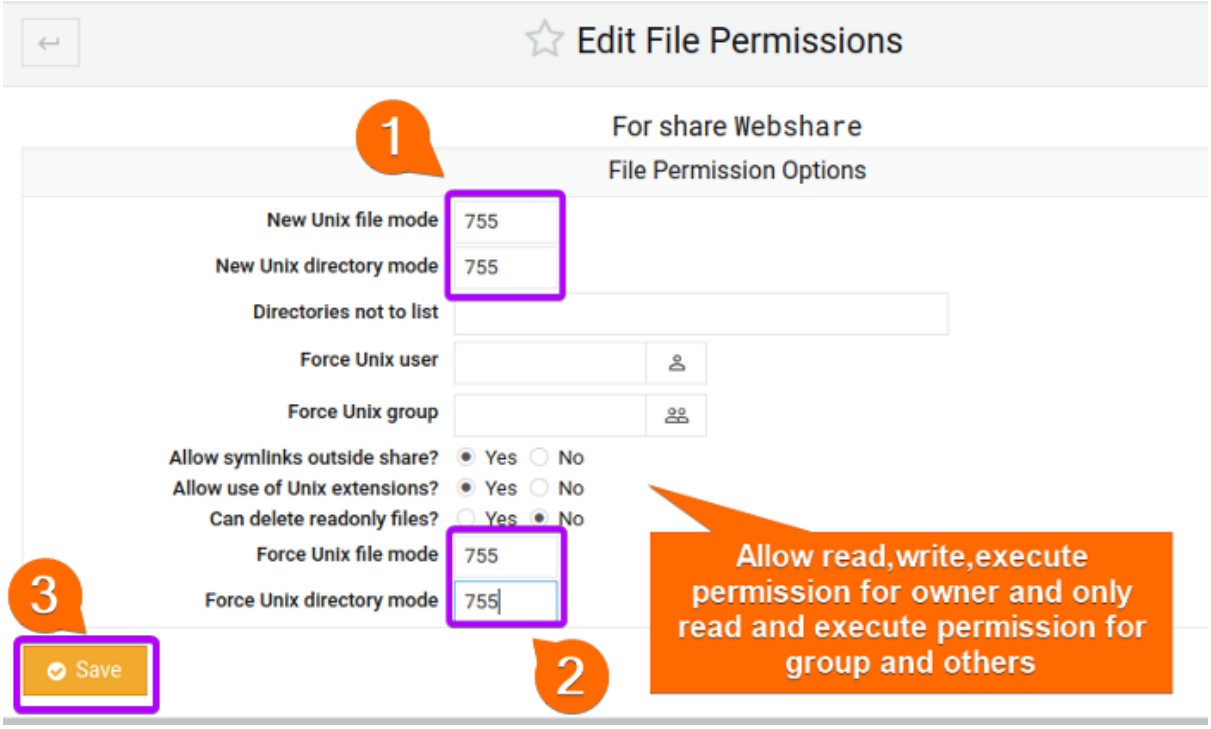
Whether the contents of a shared folder will be readable, writable, or executable depends on the **permissions granted to users**. You can perform this in the File Permissions section.

Steps to follow >

❶ Go back to Edit File Share page and click on **File Permissions**.



❷ Set the **Unix file mode to 755** as in the image below and save the changes.



The image shows that the file permission is 755. Here 7 indicates read-write-execute permission for the owner and 5 denotes read-and-execute perfusion for both groups and others.

Step 6: Accessing Files

Now to create files and access them from either Windows or Linux operating system, carry out the instructions below.

Steps to follow >

❶ Run the command below to install all **cifs and utils protocols** which are mandatory parts of the file sharing process.

```
sudo apt install cifs-utils samba client -y
```

EXPLANATION

- **cifs-utils**: Package that supports mounting CIFS/SMB file system. CIFS is the core file sharing protocol used by Samba.
- **samba client**: Package containing necessary client tools for Samba shares and other Samba-related operations.
- **-y**: Automatically answers 'yes' to any confirmation message during installation.

```
ayesha@ubuntu:~$ sudo apt install cifs-utils samba client -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

❷ Then type in the following command to find out the **IP address** of the host associated with your Linux system.

```
hostname  -I
```

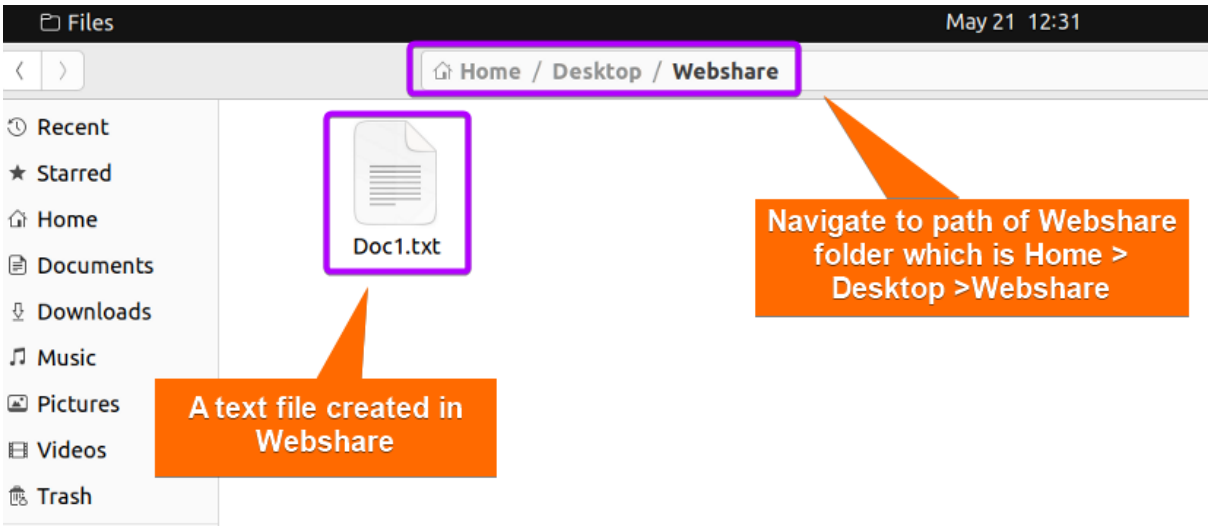
EXPLANATION

- **hostname -I**: Obtain IP address of the current Linux system.

```
ayesha@ubuntu:~$ hostname -I
192.168.153.128
```

The command shows that the IP address of my Linux system is **192.168.153.128.**

❸ Then navigate to Files >Home> Desktop>Webshare. I have already **created a text file** in **Webshare** folder.

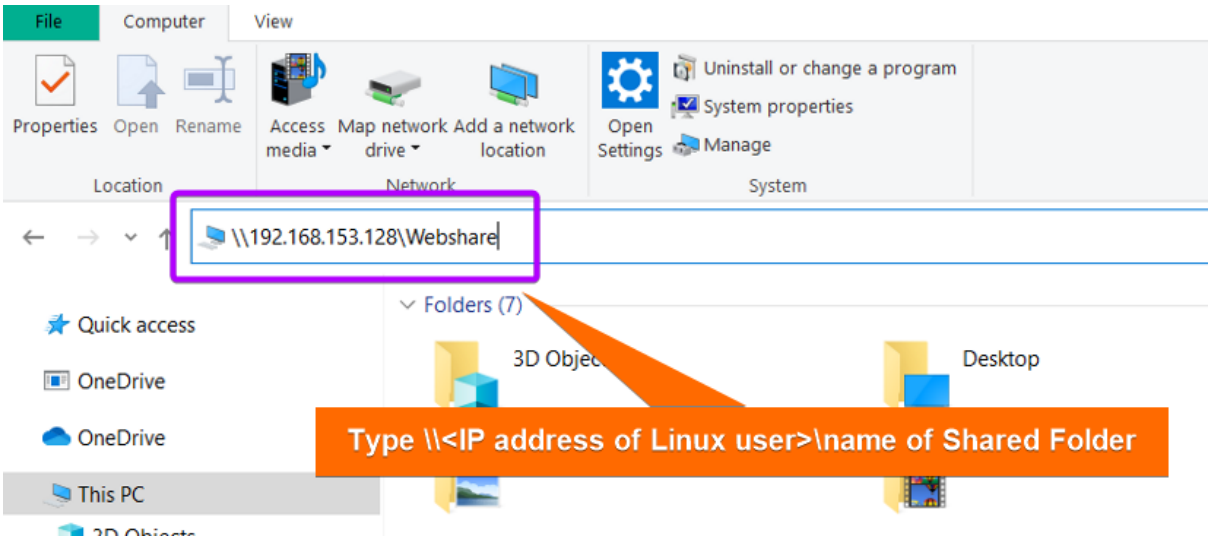


❹ Afterwards go to the Windows system. Go to This PC and type the IP address you obtained along with the folder name, Webshare in your address bar.

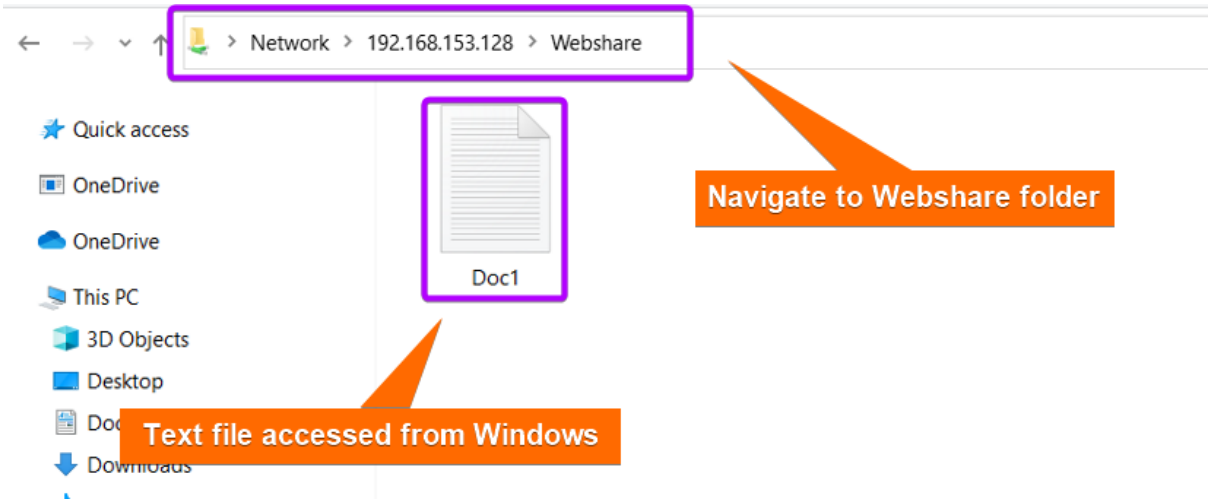
```
\\192.168.153.128\Webshare
```

EXPLANATION

- **192.168.153.128:** IP address of my Linux system.
- **Webshare:** Name of the folder used for sharing files.



❺ Now open **Webshare** folder from Windows and find the text file which you had created previously in your Windows system.



You can see that the file is now **accessible from Windows**.

C. Backup of Samba Files

It is crucial to have a backup of your Samba shared files since easy and flexible access can often pose **potential security risks**. Having a backup ensures **recovery of data** in case of **accidental deletion** or **data loss incidents** and you can continue your sharing operations without any interruption.

Steps to follow >

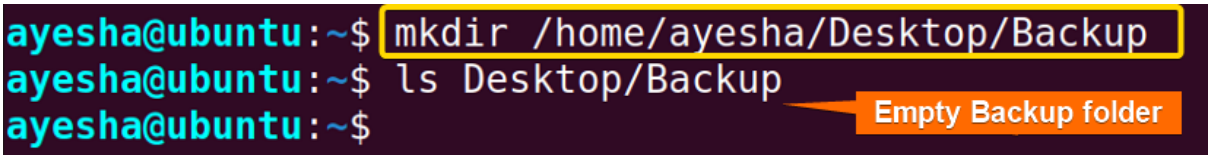
❶ First create a backup directory where you will save your shared folder. I have chosen to create a directory called **Backup** in Desktop and viewed its contents with the following commands. You can choose your directory location according to your preference.

```
mkdir /home/ayesha/Desktop/Backup
```

```
ls Desktop/Backup
```

EXPLANATION

- **mkdir**: Creates a folder or directory.
- **/home/ayesha/Desktop/Backup**: Absolute path of the Backup Directory.
- **ls Desktop/Backup**: Lists the contents of the Backup directory located in Desktop.



❷ **Regular backup** will involve **copying** the Samba shared files to the Backup folder at **specified intervals**. Edit the **cron table** with the command below and you can define your desired schedule for backing up process.

```
crontab -e
```

EXPLANATION

- crontab -e: Edits cron table or crontab. Crontab contains commands or scripts that are executed at specified time intervals.

❸ Now type the command below in the crontab editor.

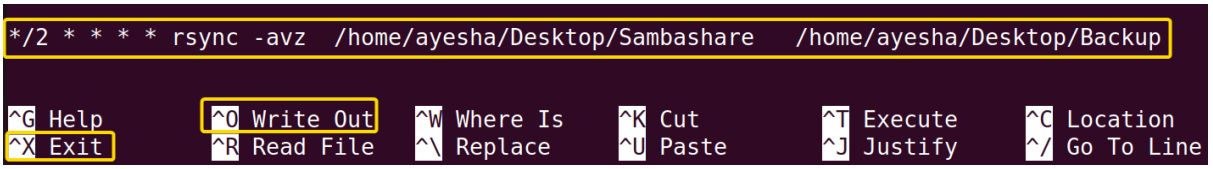
```
*/2  *  *  *  rsync  -avz  /home/ayesha/Desktop/Sambashare
/home/ayesha/Desktop/Backup
```

EXPLANATION

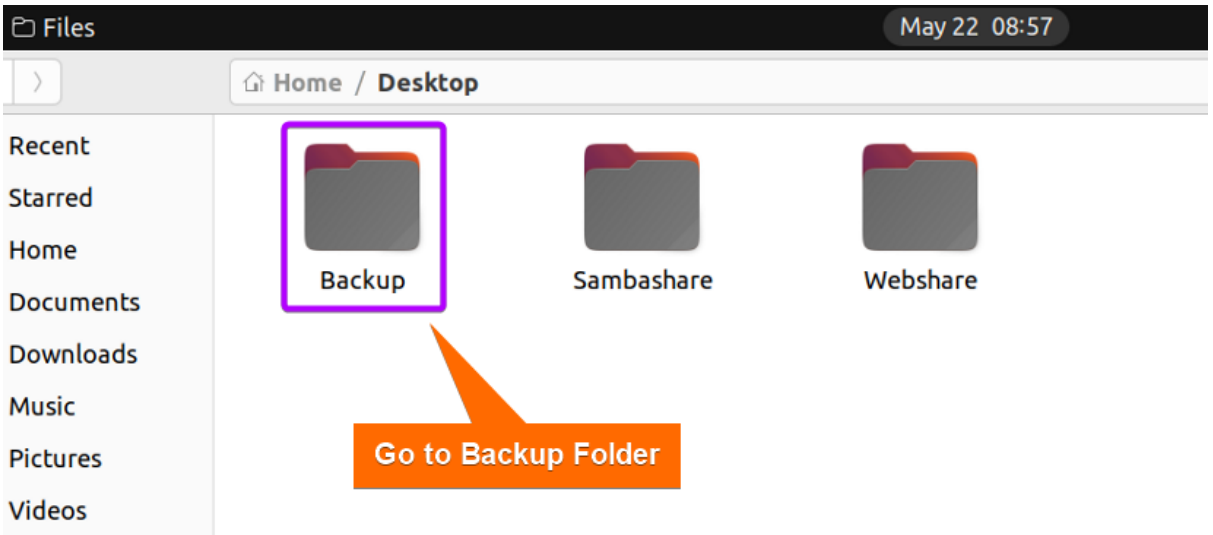
- ***/2 * * ***: Specifies that the cron job will run every two minutes. This means the copying of files will be executed every two minutes.
- rsync: Copies and synchronizes files with different locations.
- **-a**: Stands for 'archive' and preserves file permissions, ownership, timestamps, and symbolic links. Copied files will retain their original attributes.
- **-v**: Stands for 'verbose' and displays information about files being transferred.
- **-z**: Compresses data during file transfer.
- **/home/ayesha/Desktop/Sambashare**: Source directory of the files that you want to backup.

- **/home/ayesha/Desktop/Backup**: Destination directory where files will be sent for backup.

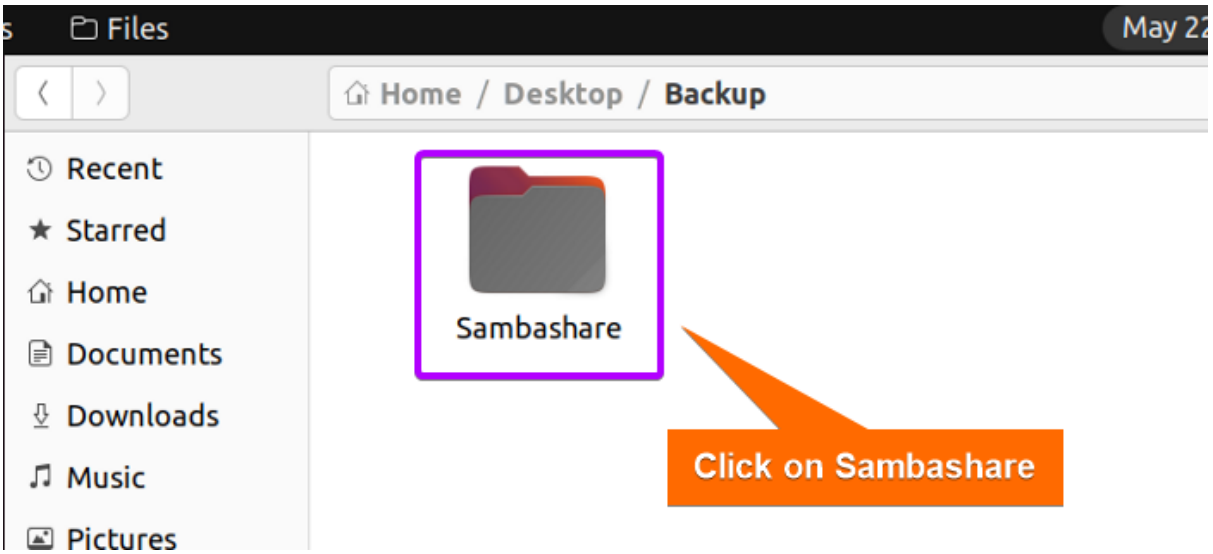
❹ Press **CTRL+O** and **CTRL+X** to save the editor file and exit respectively.



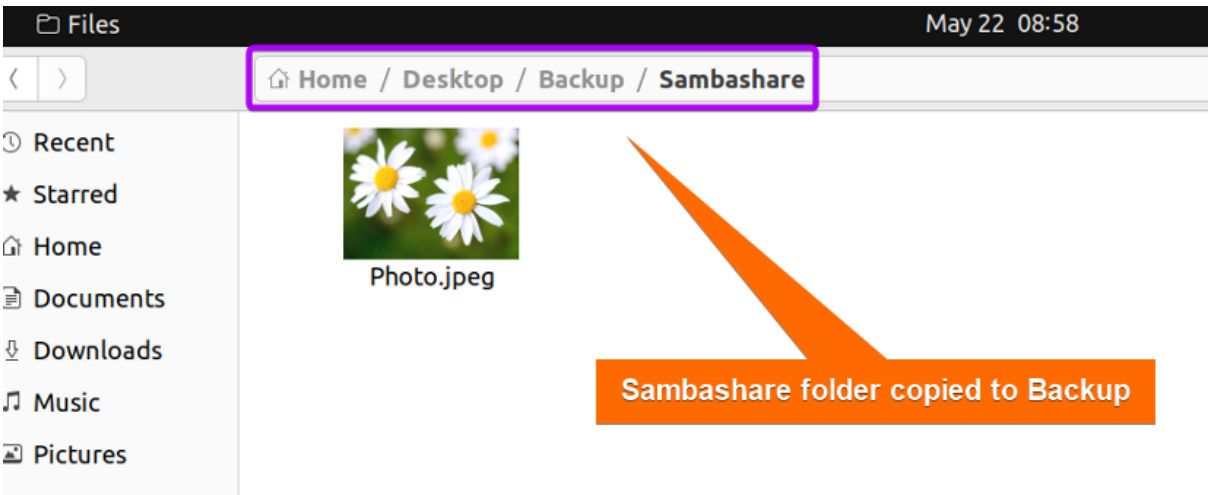
❺ Now go to Files > Home > Desktop and click on Backup folder within Desktop.



❻ You will find the entire **Sambashare folder copied to Backup folder**.



❼ Proceed to Sambashare by clicking on it.



The image shows that the contents of Sambashare are automatically copied in the Backup folder.

D. How to Remove Samba from Ubuntu

If you **no longer require** Samba or want to **simplify system configuration**, removal of Samba would be a wise decision. But before Samba deletion, make sure that it will not adversely affect any other application or service in Linux system. Remember to have a **backup of important files** before proceeding with the removal.

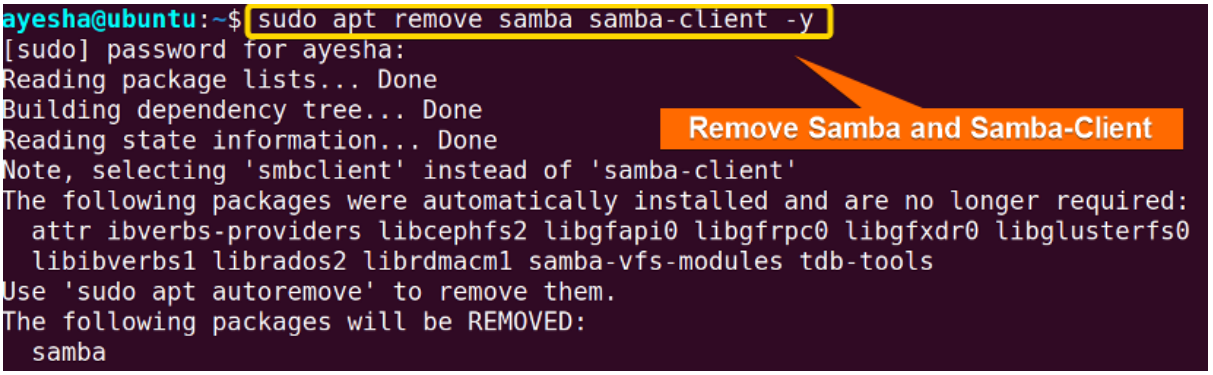
Steps to follow >

❶ At first, you need to **uninstall** the **Samba and Samba-client package** from your system with the command below.

```
sudo apt remove samba samba-client -y
```

EXPLANATION

- **remove**: Removes or uninstalls specified packages from the system.
- **samba**: Name of open-source software package, Samba for file sharing services.
- **samba-client**: Package containing tools and utilities for interacting with Samba servers.



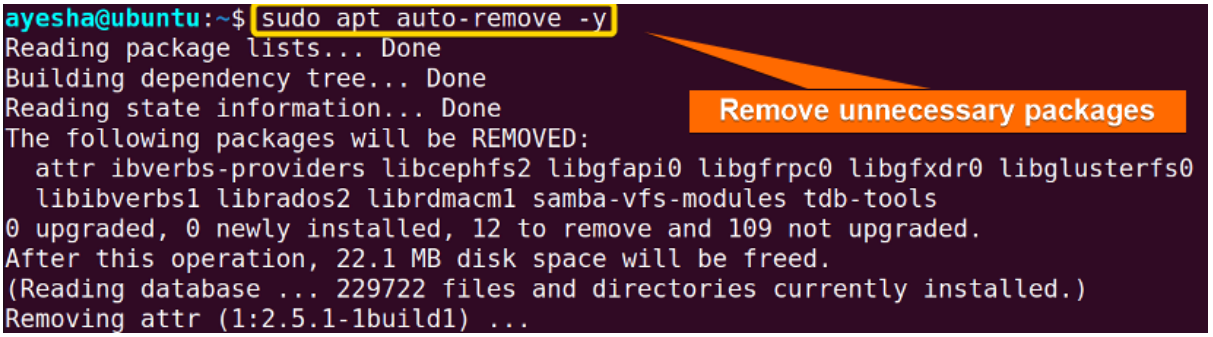
The image above shows all packages required for connecting to Samba shares and accessing files are removed.

❷ Run the command below to **remove** any **unnecessary or unused packages** that are no longer needed.

```
sudo apt auto-remove -y
```

EXPLANATION

- **auto-remove**: Automatically removes any unused packages that were initially installed but are no longer needed.



As depicted in the image above, the removal of unnecessary packages freed up disk space.

Conclusion

Samba is a powerful tool to **facilitate smooth file sharing** between Linux and Windows systems. In this article, I have demonstrated a **step-by-step guide** on how to install and configure Samba using both a command line interface and a web-based tool. Additionally, the removal of Samba and backup of Samba files enhance **data integrity** and **easy data restoration** when needed. Furthermore, this guide will provide you optimal file sharing experience. Comment below and let us know if you have found this article helpful.

Prepared By:
Ayesha Tun Nashrah

Web View: How to Install and Configure Samba Server in Ubuntu [4 Steps]