====== Simple NAS Server ======

A huge problem in most households these days is the growing collection of data without an easy way to share it and store it in one spot. Music. Movies. Pictures. Documents, backups, operating system images.

Wow! Where do you put all of this stuff?

If you've been following this thread so far - this is a little extra credit...

It's fairly simple to create a Network Attached Storage (NAS) device that any operating system can mount and write to on the network.

By using Samba, you can turn any linux server into a common shared storage medium for OSX, Linux, and Windows. Basically anything that does CIFS/SMB networking as a client

And it gets even better. With the lower cost of disk drives, it is quite possible to setup a 5-10 TB disk array as your Samba storage pool.

Now Samba, much like Windows Server, has many options, but we're going to keep this simple...

We create a couple of mountpoints, and set up some users, and that's pretty much about it - the configuration can be tuned, and there's many options, but let's not go down that for the moment.

This is one of the easiest possible configurations one can do with Samba... oddly enough, many consumer grade routers get this wrong... mainly because they over do things.

NOTE - This assumes you’re behind a firewall, although we will still be restricting the share to the local network. As a rule, all home networking should be done on a private network behind your firewall. The reason is simple; everyone makes mistakes and you may be in a hurry someday and skip a step and allow the whole world access to your home network.

==== Getting Started ====

Ignore the default configuration file and start over from scratch. The trick to configuring Samba quickly is to use only the configuration options you really need. But first, let’s set up an account for all of your Samba users, create sharepoints, and give thme proper permissions.

Create an account just for your SMB user.

As admin, issue the following commands:

```
sudo useradd fileserver901
```

and give that user a password

```
sudo passwd fileserver901
```

(This is up to you. Give the account a secure password. Number and special characters are always a good combination.)

Create a share folder if you're just starting now, having skipped the LVM section

```
sudo mkdir -p /var/share
sudo mkdir -p /var/media
```

Change permissions on the shares.

```
chown -R fileserver901:fileserver901 /var/share
chown -R fileserver901:fileserver901 /var/media
```

Note - Samba uses the underlying UNIX authorization and account mechanisms. In order to set up Samba, we have to create or reuse a directory and give our Samba user account access to it. If we did not set these permissions, then the volume would be mountable, but no one could write or read from the volume.

==== Setting up and configuring Samba ====

Next we are going to set up and configure Samba to share out the directories we just set up and force all anonymous users to become the user fileserver901. It is important to note that the underlying Linux file permission structure must belong to our user, fileserver901.

Install Samba, might already be there, but let's just make sure - in this example, we're doing it the debian/Ubuntu way, as this is how we've been so far...

```
sudo apt install samba smbclient
```

Make a backup of your current samba configuration file.

```
sudo mv /etc/samba/smb.conf /etc/samba/smb.conf.original
```

Now create a new smb.conf file

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

Note the only item to change will be your subnet value in the hosts

allow line

Note2 - the workgroup item can be whatever your workgroup is...
<code>
[global]
uncomment the line below for better performance on some platforms
on gigabit ethernet, with an untuned kernel we're seeing about
115MB/Sec on large files
# socket options = TCP_NODELAY
workgroup = WORKGROUP
netbios name = TESTBOX
security = user
hosts allow = 192.168.1.0/24
restrict anonymous = 2

[share]
comment = Home File Server
path = /var/share
force user = fileserver901
force group = fileserver901
read only = no
browsable = yes
create mask = 0755

[media]
comment = Home Media Server
path = /var/media
force user = fileserver901
force group = fileserver901
read only = no
browsable = yes
create mask = 0755
</code>

//**SECURITY** - in earlier versions of this section, we had **"guest
ok = yes"**, in this version, we've commented that out as I've found a
corner case with a CIFS/SMB Client that might allow access to the
shares - if you have an older version of this how-to, please review
your **smb.conf** and make the appropriate change.//

Test the syntax of the /etc/samba/smb.conf file, making sure you have
your configuration correct:

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

You should see something similar to:
<code>
$ sudo testparm /etc/samba/smb.conf
Load smb config files from /etc/samba/smb.conf
rlimit_max: increasing rlimit_max (1024) to minimum Windows limit

```
(16384)
Processing section "[share]"
Processing section "[media]"
Loaded services file OK.
Server role: ROLE_STANDALONE

Press enter to see a dump of your service definitions

# Global parameters
[global]
        netbios name = TESTBOX
        security = USER
        idmap config * : backend = tdb
        hosts allow = 192.168.1.0/24


[share]
        comment = Home File Server
        path = /var/share
        force user = fileserver901
        force group = fileserver901
        group = fileserver901
        read only = No
        create mask = 0755

[media]
        comment = Home Media Server
        path = /var/media
        force user = fileserver901
        force group = fileserver901
        group = fileserver901
        read only = No
        create mask = 0755
</code>
Restart the smb service..

  sudo /etc/init.d/samba restart

==== Testing Samba ====

For testing, add a samba test account

  sudo useradd smbuser
  sudo smbpasswd -a smbuser
  New SMB password:
  Retype new SMB password:
  Added user smbuser

** Test the network lookup and client side **
```

<code>
nmblookup -A 192.168.1.6
Looking up status of 192.168.1.6
        TESTBOX   <00> -   B <ACTIVE>
        TESTBOX   <03> -   B <ACTIVE>
        TESTBOX   <20> -   B <ACTIVE>
        ..__MSBROWSE__. <01> - <GROUP> B <ACTIVE>
        WORKGROUP   <00> - <GROUP> B <ACTIVE>
        WORKGROUP   <1d> -   B <ACTIVE>
        WORKGROUP   <1e> - <GROUP> B <ACTIVE>

        MAC Address = 00-00-00-00-00-00
</code>

Now login and query the server directly

<code>
smbclient -U smbuser -L \\TESTBOX
Enter smbuser's password:
Domain=[WORKGROUP] OS=[Windows 6.1] Server=[Samba 4.3.9-Ubuntu]

        Sharename   Type   Comment
        ---------   ----   -------
        share   Disk   Home File Server
        media   Disk   Home Media Server
        IPC$ IPC   IPC Service (Samba 4.3.9-Ubuntu)
Domain=[WORKGROUP] OS=[Windows 6.1] Server=[Samba 4.3.9-Ubuntu]

        Server   Comment
        ---------   -------
        TESTBOX   Samba 4.3.9-Ubuntu

        Workgroup   Master
        ---------   -------
        WORKGROUP   TESTBOX
</code>

==== Add Samba Accounts ====

This setup uses the linux/unix accounts, but to all access to the Samba server, we have to create an SMBPASSWD - those accounts do not need to have unix passwords (even though they might, if you, as admin want, if then, just do a sudo passwd userid)

```
  sudo useradd alice
  sudo smbpasswd -a alice
  sudo useradd bob
  sudo smbpasswd -a bob
```

//Note - you can also use **adduser** - this builds a full blown

linux/unix user account with /home/user directory with full login privileges - we don't need that for basic SAMBA account//

//**Security Note** - We have built a SMB server that copies over an “anonymous read/write configuration” that will allow all machines on the 192.168.1.0/24 subnet to access the share. We use the USER directive to allow users into the server - without an account, they don't have access - but all users with accounts will have access all files om the specific shares//

//From a unix filesystem view - Every user account that connects to Samba will be “forced” to become user/group=fileserver901, e.g. the SAMBA unix account.//

//As long as there is an external firewall in front of your small network, then this is a perfectly acceptable configuration as it offers two layers of security. Layer 1 is the firewall, which should not allow any incoming traffic. Layer 2 explicitly allows only your local subnet access. This is an ideal home network setup, but may not acceptable for a small office or a corporation, obviously.//

See below - all the files in that share, they are "owned" by the fileserver901 account, not the userid account.

```
  ls -l /var/share
  total 1529796
  -rwxr--r-- 1 fileserver901 fileserver901    4578672 Mar  5 16:46
Acrylic_WiFi_Home_v3.1.5877.19629-Setup.exe
  -rwxr--r-- 1 fileserver901 fileserver901   10466486 May 13 17:10
mibbrowser.zip
  -rwxr--r-- 1 fileserver901 fileserver901   26951680 Mar  1 17:31
MobaXterm_Setup_8.6.msi
  -rwxr--r-- 1 fileserver901 fileserver901   38622000 May 13 17:10
setup.exe
  -rwxr--r-- 1 fileserver901 fileserver901 1485881344 Apr 21 10:58
ubuntu-16.04-desktop-amd64.iso
```

See? All SMB users will have access to those files...

Security - when you're done testing the SAMBA server - disable the smbuser account

```
  sudo smbpasswd -d smbuser
```

In summary

Getting cross platform file sharing working with Samba can be incredibly complicated, unless you focus on just the components of Samba you need.

We want to set up cross-platform file sharing via smb, not set up a Samba Domain Controller. The next step for the home user is to ponder how big of a NAS to set up. If you plan ahead, you could buy several large disks and stripe them together for redundancy and speed, and store all of your music and videos on it.