

1 Samba

by Alain Knaff

Samba (named after Microsoft's **S**erver **M**essage **B**lock protocol) is an Open Source/Free Software suite that provides seamless file and print services to Windows clients. It can act as a primary domain controller (authentication server) to all major variants of Windows.

The course will show how to:

- install the necessary software
- configure samba for some basic file service tasks
- configure samba for to operate as a PDC

1.1 What is Samba

As the front page at samba.org says, "Samba is an Open Source/Free Software suite that provides seamless file and print services to SMB/CIFS clients." Samba is freely available, unlike other SMB/CIFS implementations, and allows for interoperability between Linux/Unix servers and Windows-based clients.

Samba is a software package that gives network administrators flexibility and freedom in terms of setup, configuration, and choice of systems and equipment. Because of all that it offers, Samba has grown in popularity, and continues to do so, every year since its release in 1992.

1.2 Installation & Configuration

The installation of Samba on Debian GNU/Linux is normally quite easy! The packages are pre-built, and you just have to run `apt-get` for the installation of the needed packages. Thus, the first step is:

```
apt-get install samba smbclient
```

This has to be done as root! During the installation, a few questions will be asked to build an initial database.

Question	Answer
Workgroup/Domain Name?	[country]
How do you want to run Samba?	daemons
Create samba password database?	Yes

After the installation, you should be able to do a first test (samba is automatically started from the system). This example assumes that your machine is called `berne`:

```
smbclient -L berne
```

This will ask for a password, simply type return, and you will see a list of all shares that are defined, as well as some other servers¹

1.2.1 Structure of the configuration file

The configuration file is made up of various *sections*. These are named [`section name`]. Most sections represent file or print shares. Parameters which apply to samba as a whole are set in a `global` section.

There are a number of reserved shares or sections, such as for example

- `printers` defines parameters for printers which are not explicitly listed
- `netlogon` is used for holding the Windows startup scripts when operating as a PDC
- ...

1.3 Samba as a simple file server

1.3.1 Global parameters

The following global parameters are relevant to simple file server operation:

Name	Description
<code>workgroup</code>	Windows workgroup or Domain
<code>NetBIOS name</code>	NetBIOS name by which a Samba server is known. By default it is the same as the first part of the Unix host name.
<code>wins support</code>	yes if this server should be a wins server, no otherwise.
<code>wins server</code>	the IP address of the Wins server (only set this if <code>wins support</code> is no).
<code>map to guest</code>	set this to <code>Bad User</code> to allow all users (even not logged in) to browse.

Set `wins support` to `no` if you intend to put your server into an existing Windows network, which already has a Wins server.

¹This list of nodes is only shown on a samba server which is a master browser for a workgroup, and only contains nodes of that workgroup. A master browser is not the same thing as a wins server. The master browser, responsible for supplying a list of all servers for the network neighborhood, is dynamically elected. The WINS server, responsible for translating names into IP addresses, is configured statically. For more information about master browsers, see <http://ubiqx.org/cifs/Browsing.html>

1.3.2 Per share parameters (all shares)

The following parameters are useful on all shares:

Name	Description
<code>comment</code>	Informative text to be displayed near share in Windows Browser
<code>browseable</code>	If set to <code>yes</code> (default), share shows up in network neighborhood, else is hidden
<code>public</code>	All users may access this share
<code>read only</code>	Users may only read from share
<code>available</code>	If set to <code>no</code> , share is switched off

1.4 Parameters for file shares

The following parameters are useful on file shares:

Name	Description
<code>path</code>	Path of Unix directory which is exported in this share

1.4.1 Parameters for printer shares

The following parameters are useful on printer shares:

Name	Description
<code>printing</code>	Identifies the printing system. One of <code>cups</code> , <code>plp</code> or <code>lprng</code> ...
<code>printable</code>	must be set to <code>yes</code>
<code>printer</code>	Unix (Cups, ...) printer corresponding to this share
<code>path</code>	Path of temporary directory where print jobs are to be spooled to (by default <code>/tmp</code>)
<code>cups options</code>	If your printing system is <code>cups</code> , this specifies the options passed on to <code>cups</code> . Usually, " <code>raw,media=a4</code> ". These mean that "raw" mode should be used (no postscript processing, because in the Windows world, the "printer driver" lives on the client), and that the paper size is A4 .

1.4.2 Parameters for the global printers share

If you don't want to define each printer individually, you can set up a global `printers` share which exports all printers known locally to Windows.

For this, set `load printers = yes` in the `global` section, and define a `printers` section.

1.4.3 User management

So far, we have not yet defined any users in Samba. The file server is already usable, but only for anonymous access (guest account).

If you want to set up named access, the following parameters need to be defined:

Name	Description
<code>guest account</code>	Name of Unix user who will serve anonymous (guest) requests (usually <code>nobody</code>)
<code>username map</code>	(Optional) Name of a file which maps long Windows user names to short Unix login names

The `username map` has the following format (left is the Unix login, right the long Windows name):

```
root = admin administrator
tridge = "Andrew Tridgell"
```

Once these changes are done, you add samba users using the following command:

```
smbpasswd -a user
```

These users have to be existing Unix users; the `smbpasswd` command only enables them for samba.

1.4.4 Testing

The following tools are available for testing:

- `testparm`: this parses the `/etc/samba/smb.conf`, mentions any errors that it finds, and waits for a keystroke. After the keystroke, it prints out the whole configuration, as understood by samba
- `smbclient`: `smbclient` is a samba client that allows you to access your file server, just as a Windows workstation would. Of course, it can also be used to access a real Windows server.

```
smbclient -L server -U user
```

```
smbclient //server/share -U user
```

The first command logs in as *user* and lists all shares on *server*.

The second command logs in as *user* connects to *share* on *server*. Once connected, the you get an ftp-like command line interface to get and put files on the server.

- log files are put into `/var/log/samba/machine.log`, where *machine* is the NetBIOS name of the client having connected. Set `log level` to at least 3 in the global section of `smb.conf` to see a log of all files that are opened.

- `nmblookup`: `nmblookup` resolves windows computer names (NetBIOS names) into IP addresses.

```
nmblookup NetBIOSname
```

```
nmblookup -U winsServer -R NetBIOSname
```

The first finds *NetBIOSname* using broadcast. The second finds *NetBIOSname* by specifically the *winsServer*.

1.4.5 Example

This example shows the configuration file of a simple file server:

```
[global]
    workgroup = samba
    printing = cups
    cups options = "raw,media=a4"
    load printers = yes
    encrypt passwords = yes
    log level = 3

[public]
    comment = A Test Share
    browseable = yes
    public = yes
    read only = yes
    path = /samba/public

[authenticated]
    comment = An authenticated share
    browseable = yes
    read only = no
    path = /samba/auth

[printers]
    comment = Printers share
    printable = yes
```

Exercises:

- Create the share directories on Unix (`/samba/auth`, `/samba/public`).
- Log in using `smbclient`, using various users, and put files into the shares, where possible

1.5 Primary domain controller

A primary domain controller acts as an authentication server for all windows workstations in its domain. Users authenticate to the PDC when they log in to their workstation. Once authenticated, they have access to all resources in

the domain, be it on their local workstation, on the PDC, or on other windows servers participating in the domain.

To set up a primary domain controller, you need to

- add some attributes to the global sections
- define a `netlogon` share
- define a `homes` share, which will contain Windows' users home directory
- set up a place where roaming profiles are stored

1.5.1 Global settings

Name	Description
<code>workgroup</code>	name of the domain
<code>domain logons</code>	<code>yes</code>
<code>wins support</code>	<code>yes</code> in most cases, unless you've defined several domains which share a same LAN
<code>add machine script</code>	Script to handle the joining of workstations to the domain: <code>add machine script=/usr/sbin/useradd -d / -g 100 -s /bin/false -M %u</code>
<code>logon drive</code>	Drive letter for home directory (default: Z:)
<code>logon home</code>	profile location for Windows 95/98 ²
<code>logon path</code>	profile location for Windows NT/2000/XP ³

Notes:

- Unlike with earlier Samba versions, the drive letter (`logon drive`), profile directory (`logon path`), and others should **not** be enclosed in quotes
- On Debian, by default the `homes` share is declared as non-writable. This must be changed to `writable`, or else profiles won't work!
- The purpose of the `add machine script` is to create the Unix accounts that back the machine accounts which are created when a workstation joins the domain. Care must be taken that those accounts cannot be abused for interactive logins; that's why we set the login shell to `/bin/false`.
- `logon home` and `logon path` are interpreted by the Windows workstation (after substitution of samba variables), and should refer to an existing share. Example: `\\%L%\%U\profile`. After substitutions of samba variables by the server, this will be `\\server\user\profile`, which is then interpreted by the workstation to mean the `profile` directory in the user's home share.

1.5.2 Homes share

The homes share represents the user's home directories. Each user will "see" a share named after himself, containing his own home directory.

```
[homes]
    comment = Home Directories
    browseable = no

# By default, the home directories are exported read-only. Change next
# parameter to 'yes' if you want to be able to write to them.
    writable = yes

# File creation mask is set to 0700 for security reasons. If you want to
# create files with group=rw permissions, set next parameter to 0775.
    create mask = 0700

# Directory creation mask is set to 0700 for security reasons. If you
# want to create dirs. with group=rw permissions, set next parameter
# to 0775.
    directory mask = 0700
```

1.5.3 Roaming Profiles

On login the user's is copied from the location identified by `logon path` to the local workstation.

On logout, it is copied back to the server.

On first login, when the user does not yet have a profile on the server, his profile gets initialized from the "Default User"'s profile.

1.5.4 Netlogon share

The main purpose of the netlogon share is as a location for the startup script (identified by the `logon script` parameter), which is executed on the client workstation on login

1.5.5 Adding a workstation to the domain

When joining a workstation to the domain, you need to supply a user name and a password on the server, who has the appropriate privileges.

One such user is `root`; however in Debian, `root` is marked as `invalid user` in `smb.conf`. In order to enable him, you need to comment out the following line, if present:

```
invalid users = root
```

After having made sure that the above line is gone, `root` can now add new workstations to the domain.

In many circumstances, however, it may not be desirable to hand out the root password of the server to the people doing the maintenance on the clients. In such cases, you may set up another user in such a way that he is entitled to add machines to the domain.

After having created this user (using `useradd` and then `smbpasswd -a`), you declare him to be `admin user` on the reserved `IPC$` share.

```
[IPC$]
    admin users = winjoin bigmouse
    path = /ipc
```

The `IPC` share is a reserved file share which is used for administrative communications among windows machines. Windows workstations log in to this share for numerous tasks, including joining the domain.

The `admin users` parameter defines a space-separated list of users who will enjoy root privileges when connecting to this share.

Caution: In addition to its special meaning, the `IPC$` share may contain files just like any other shares, and the admin users have full privileges on those files. Therefore it is important to point it to a directory which contains nothing of value (create an empty directory `/ipc` just for this purpose).

After having set up the `IPC$` share in this way, the user named `winjoin`, and `bigmouse` are now entitled to add machines to the domain.

1.5.6 Setting up a “Domain administrator”

“Domain administrators” are users, defined on the PDC, which have administrative privileges on the workstations. They do not, however, have any particular privileges on the server itself.

In order to define domain administrators, you need to:

- Create or chose a Unix Group which will hold the domain administrators:

```
groupadd domadm
```

- Define this group as domain administrators:

```
net groupmap modify ntgroup="Domain Admins" unixgroup=domadm
```

- add users to this group (by editing `/etc/group`. These users now enjoy administrative privileges on the workstations.

Notes:

- The `net groupmap` database may get corrupted, especially when samba’s SID changes due to re-installation. In such case execute the following command: `net groupmap cleanup`, and try again.
- Use `net groupmap add ...` if the Windows group does not yet exist, and `net groupmap modify ...` if it does exist, or else the command will happily create two groups with different SIDs!

1.5.7 Example

```
[global]
## Browsing/Identification ###

# Change this to the workgroup/NT-domain name your Samba server will part of
workgroup = switzerland
domain logons = yes
encrypt passwords = yes
add machine script = /usr/sbin/useradd -d / -G '' -g 100 -s /bin/false %u

printing = cups
cups options = "raw,media=a4"
load printers = yes
username map = /etc/samba/user.map
...
# Windows Internet Name Serving Support Section:
# WINS Support - Tells the NMBD component of Samba to enable its WINS Server
wins support = yes
...
[homes]
comment = Home Directories
browseable = no
# By default, the home directories are exported read-only. Change next
# parameter to 'yes' if you want to be able to write to them.
writable = yes
```

1.6 Password synchronization

Due to the peculiar way how Windows workstations authenticate to servers and PDCs, the Windows password record format (as stored in `/etc/samba/smbpasswd`) is fundamentally incompatible with Unix' password records (as stored in `/etc/shadow`).

By default, these passwords are independent of each other: if the end user changes his Unix password, his Windows password is unaffected, and vice-versa. However, this is rather confusing to the end users, and this is where password synchronization steps in.

With password synchronization, the `smbpasswd` utility also changes the Unix password, and vice versa.

1.6.1 Unix password follows Samba

In order to make the Unix password follow the samba password, two steps are needed:

configure samba : Add (or uncomment) the following lines to `/etc/samba/smb.conf`:

```
unix password change = yes
pam password change = yes
```

configure pam : Add the following line to `/etc/pam.d/samba`:

```
@include common-password
```

You may test password change:

```
smbpasswd -r server -U user
```

This command performs the samba password change the same way as a windows workstation would. Change the samba password of user, and then check (by logging in via `ssh`, for instance), that the Unix password has been changed as well.

1.6.2 Samba password follows Unix

In principle, a similar method should allow to do the synchronization in the other direction, by appending the following line to `/etc/pam.d/common-password`:

```
password sufficient pam_smbpass.so nullok try_first_pass use_authtok
```

However, unlike other distributions, such as SuSE, Debian does not ship the `pam_smbpass` module with its samba.

To solve this, you may either:

- compile samba from source, and enable the feature:

```
./configure --with-pam --with-pam_smbpass
```

- simply symlink `/usr/bin/passwd` to `/usr/bin/smbpasswd`

1.7 Access control

In addition to the normal Unix file permissions, samba allows very fine-grained access control to shares.

This section describes how to control access by user, by workstation IP, and how to control the mapping of samba users to Unix users.

1.7.1 Access control by user

The following settings define which users may access a share:

Name	Description
<code>valid users</code>	Users who may connect to this share
<code>invalid users</code>	Users who may not connect to this share

Both lists may contain individual users or groups. If a user ends up in both lists at once, `invalid users` takes precedence.

The following settings define which users may access a share:

Name	Description
<code>write list</code>	Users who may write to this share
<code>read list</code>	Users who may not write to this share

Both lists may contain individual users or groups. If a user ends up in both lists at once, `write list` takes precedence, i.e. users that are in both lists at once may write.

The `admin users` setting defines a list of users who are granted administrator access to the share (i.e. they perform all operations on the share as root).

1.7.2 Access control by IP

The following settings define which IP addresses may access a share:

Name	Description
<code>hosts deny</code>	IP addresses who may not connect
<code>hosts allow</code>	IP addresses who may connect.

Both lists may contain individual machines or subnets (IP/netmask). If a machine happens to be in both lists at once, `allow` takes precedence.

1.7.3 Unix rights granted to share users

The following settings define which Unix rights the Samba users get:

Name	Description
<code>force user</code>	If this is set, all valid users connecting to the share act as this Unix user
<code>force group</code>	If this is set, all valid users connecting to the share act as belonging to this Unix group
<code>create mask</code>	“maximal” set of permissions set on newly created files. If client who creates a file asks to grant more permissions than specified in mask, the additional permission bits are silently ignored. For instance, if the mask does not include the <i>world writable</i> bit, samba will not create any world writable files, even if client asks it to.
<code>directory mask</code>	same <code>create mask</code> , but for new directories, rather than files
<code>force create mode</code>	“minimal” set of permissions set on newly created files. If client who creates a file asks to grant less permissions than specified in mask, the missing permission bits are set anyways. For instance, if the mode includes the <i>group readable</i> bit, samba will make files group readable, even if client didn't ask for group readable files.
<code>directory mode</code>	same as <code>force create mode</code> , but for new directories, rather than files
<code>force security mode</code>	same as mode, but applies to permission bit changes (<code>chmod</code>), rather than new object creation. There is a <code>force security mode</code> , a <code>security mask</code> , a <code>force directory security mode</code> and a <code>directory security mask</code> (minimal/maximal bit masks, applicable to directories or plain files)
<code>dos filemode</code>	The default behavior in Samba is to provide UNIX-like behavior where only the owner of a file/directory is able to change the permissions on it. However, this behavior is often confusing to DOS/Windows users. Enabling this parameter allows a user who has write access to the file (by whatever means) to modify the permissions on it.

1.8 Samba variables

Often it is interesting to make values of samba configuration parameters dependent on the environment, such as properties of the client or user connecting to the service. This can be done using *samba variables*. Samba variables start with a percent sign (%) followed by a letter.

Variable	Description
<code>%U</code>	User name who connected to the share. This is the user name “requested”, by the client, i.e. before it is changed by <code>username map</code> , <code>force user</code> or <code>admin users</code> .
<code>%u</code>	User name assigned by samba (after taking into account any remapping performed by <code>username map</code> , <code>force user</code> and <code>admin user</code>)
<code>%G</code>	Primary Unix group of <code>%U</code>
<code>%g</code>	Forced group (if set), or primary Unix group of <code>%u</code> if there is no <code>forced group</code> not
<code>%H</code>	Unix home directory of <code>%u</code>
<code>%m</code>	Net BIOS name of client workstation
<code>%I</code>	IP address of client workstation
<code>%a</code>	Windows variant of client (one of <code>WfWg</code> , <code>Win95</code> , <code>WinNT</code> , <code>Win2k</code> , <code>WinXP</code>). This variable is particularly useful to use different profile directories for different windows versions (there may be some issues when <code>WinNT</code> and <code>Win2k</code> use the same profile, so we better keep them separate).
<code>%L</code>	Net BIOS name of server

1.9 Miscellaneous Gimmicks

1.9.1 User monitoring

`smbstatus` : The `smbstatus` command displays the currently logged in users, as well as the shares, locks and files that they have currently open.

`root preexec` : The `root preexec` and `root postexec` share parameters specify a program that is executed whenever the share is mapped and/or unmapped. It can be used to propagate samba login/logout activity to Unix’s `last` facility:

```
root preexec = /usr/X11R6/bin/sessreg -l %m -h %M -a %u
root postexec = /usr/X11R6/bin/sessreg -l %m -h %M -d %u
```

1.9.2 Time synchronization

The following line, in the global section, enables the samba server to act as a timeserver for its workstations:

```
time server = yes
```

To make use of this feature, the client workstation needs to execute the following command (for instance, from its startup script):

```
net time \\berne /set /y
```

1.9.3 Hiding files

hide The following hides the files with the named extensions (slash-separated list). They can still be accessed by their name, but will not show up in directories (their Dos H bit is set)

```
hide files = *.exe/*.scr
```

veto The following makes the files with the named extensions (slash-separated list) completely inaccessible to samba. These files cannot be accessed, even if the user knows their name

```
veto files = *.exe/*.scr
```

1.9.4 Included configuration files

It is possible for `smb.conf` to refer to other configuration files. This may be useful for better organizing the samba configuration, and also for making some configuration aspects dependent on samba variables.

complete override If the named file exists, the current configuration file is overridden (i.e. all settings read from the current file, except location of new file, are forgotten), and new file is read instead. If the named file does not exist, the settings from the current file are retained

```
configuration file = /etc/samba/lib/smb.conf.%m
```

include / merge The new file is read, and its setting merged with those read from the current file:

```
include = /etc/samba/lib/smb.conf.%m
```