Hentzenwerke Whitepaper Series

The Ten Minute Guide to Setting Up a Linux File Server with Samba

By Whil Hentzen

One of the common requirements in a Linux-based network is to include Linux workstations. There are a multitude of ways to do so; if this network also includes Windows workstations that are connecting to the Linux file server, you're likely already using Samba. This document describes how to have a Linux workstation access data on a Linux file server via Samba. Here's a quick tutorial to the essential steps involved in setting up a Linux File Server and having Linux workstations connect to it, using Fedora Core as an example.

Hentzenwerke Publishing, Inc. • books@hentzenwerke.com • www.hentzenwerke.com

1. Preface

1.1 Copyright

1.2 Revisions

1.2.1 History

Version	Date	Synopsis	Author
1.0.0	2004/8/23	Original	WH

1.2.2 New version

The newest version of this document will be found at www.hentzenwerke.com.

1.2.3 Feedback and corrections

If you have questions, comments, or corrections about this document, please feel free to email me at 'books@hentzenwerke.com'. I also welcome suggestions for passages you find unclear.

1.3 Acknowledgments

This guide is an abbreviated cookbook from the Linux Transfer for Windows Network Admins by Hentzenwerke author Michael Jang with assistance by Steve Suehring.

1.4 Disclaimer

No warranty! This material is provided as is, with no warranty of fitness for any particular purpose. Use the concepts, examples and other content at your own risk. There may be errors and inaccuracies that in some configurations may be damaging to your system. The author(s) disavows all liability for the contents of this document.

Before making any changes to your system, ensure that you have backups and other resources to restore the system to its state before making those changes.

All copyrights are held by their respective owners, unless specifically noted otherwise. Use of a term in this document should not be regarded as affecting the validity of any trademark or service mark. Naming of particular products or brands should not be seen as endorsements.

1.5 Prerequisites

This document was written using Fedora Core 1.0 as a workstation and RedHat 9.0 as a file server, and assumes a beginner's familiarity with use of Linux via the GUI and the Command Window.

2. Materials needed before you begin

Note that this is a cookbook that simply lists the steps required to connect a Linux workstation to shares on a Linux server, using Samba. The intended audience is someone who just wants a concise list of the exact steps to follow, perhaps because they've done this before, but need a quick refresher, or maybe because they're familiar with the ideas behind file sharing, but simply need a single cohesive 'how-to' to put it all together. This isn't a full-blown tutorial that explains the concepts, reasoning, and options behind each step. There are plenty of books (including one of ours!) and materials on the Web that provide the background should you need it.

3. Sample scenario

The scenario I'll use is a software development company with a president (al) and a vice president (barb), an administrative assistant (carl), several developers (dave, donald, donna, dylan), and several testers (tami, thorne, tom). For purposes of this document, we're going to make the following assumptions and requirements:

3.1 File server only
The file server will not be a primary domain controller (PDC), and so users will not authenticate against the file server. In other words, they'll log on to their own workstations.

3.2 All data on server
All data is stored on the file server. Common data is stored in directories that are shared by users. Users will not store their own data on their own machines, but will have separate home directories on the server for that purpose.

3.3 Server data directory structure
The file server's data directory structure is as follows. The following directories are shared by users, according to permissions and rights:

```
/home/admin
/home/development
/home/testing
/home/music
/home/users
```

For example, /home/admin is accessible by the administration folks (al, barb and carl), while /home/development is accessible by all of the development team members and /home/testing is accessible by the testing folks. /home/music is accessible by all. The users directory is further divided into subdirectories for each user, like so:

```
/home/users/al
/home/users/barb
/home/users/carl
/home/users/dave
```

and so on, one for each user.

3.4 Network configuration
Finally, the network configuration we're going to use during this discussion consists of a Linux file server and two Linux workstations. All the machines are on an internal subnet, like so:

Function	Machine Name	IP Address
Server	Jeeves	192.168.1.100
Workstation1	WS1	192.168.1.1
Workstation2	WS2	192.168.1.2

4. Concepts
Before we get started, let's revisit a couple of key concepts.

4.1 Why Samba?
Although I've already explained that this paper is covering the connection between a Linux workstation and a Linux server using Samba, some folks may be wondering about the choice. "After all, Samba is used for connecting Windows and Linux machines. It's not necessary if you're just connecting Linux machines." This is true; there are ways, such as NFS, to accomplish this without Samba. However, this paper is based on the scenario that your network has Windows (and possibly Mac) machines as well, and thus the server already has Samba installed. In this situation, connecting Linux workstations to the server is relatively simple.

4.2 A refresher for how Windows groups and users work
With Windows file servers, you have directories, groups, and users. A user can belong to one or more groups, and a group can include one or more users. You allow one or more groups to access a directory, and give that group specific permissions, such as read but not write.

In this scenario, there would be groups for management, administration, development, testing, and for each individual user. Users al and barb belong to the management group, those two plus carl belong to the administration group, donna, dave, donald and dylan belong to the development group, tom, tami and thorne belong to the testing group, and all ten users belong to the 'everyone' group. Each user belongs to a group of their own (al belongs to the 'al and friends' group, barb belongs to the 'barb

and friends' group, and so on.) The reason for putting each individual user in their own groups is so that if a second user temporarily needed access to the same entities, you'd just have to add them to the group, not create a second user and then try to figure out which entities to give them access to.

The final step, once the users and groups are set up, is to give the management group access to every directory, give the administration group access to the administration directory, give the development group access to the development directory, and give every group access to the music directory.

Now, when someone joins the company, a new user is created and they're added to the appropriate group. They automatically get access to the same directories as everyone else in the group.

4.3 How file serving works on Linux

In Linux, the same general concepts apply. You create users and groups, assign users to groups, and then assign permissions to directories according to groups. The specific mechanisms work somewhat differently, of course.

On the server, you'll use software called Samba that provides the functionality of serving files to users (much like Apache serves Web pages to users.) Samba allows a non-Windows server to communicate with Windows computers using the same networking protocol that is used by Windows computers. It can also be used by Linux workstations for the same purpose.

You'll create users (with the Linux adduser command), and groups that contain those users (with the Linux usermod command) – all on the server. Then you'll create Samba users that correspond to the Linux users (with the Samba smbuser command). The third step is create Samba shares for specific directories on the server, via entries in the smb.conf file. For example, the /home/admin directory would have an entry in smb.conf that allows Samba to share it's contents to other machines. Finally, you'll assign users or groups to those Samba shares.

Then, on the client, you'll create a mount point (roughly, the Linux equivalent of a mapped drive in Windows) by creating a new directory. Then you'll assign the server's Samba share to that mount point either by the 'mount' command (good for just the current session) or via an entry in the client's /etc/fstab configuration file (which automatically makes the connection each time the machine is started up).

5. Set up the file server

The first step is to set up the file server so it can begin dishing out files to clients that want them. There are two general parts to this process. The first part is setting up users and data directories on the server. The second part is installing and configuring Samba.

5.1 Useful files and commands

During the process of setting up the server, we'll be accessing several configuration files and using a number of commands on a regular basis. Here's a quick reference to the important configuration files and programs that you'll be using in this section.

5.1.1 Useful Linux files

The /etc/passwd file contains Linux users together with their user and group ID numbers. It looks something like this:

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
.
.
.
al:x:500:500::/home/users/al:/bin/bash
barb:x:501:501::/home/users/barb:/bin/bash
carl:x:502:502::/home/users/carl:/bin/bash
dave:x:503:503::/home/users/dave:/bin/bash
donna:x:504:504::/home/users/donna:/bin/bash
```

Another useful file is /etc/group, which contains a list of the Linux groups and the Linux users in each group. It looks something like this:

```
al:x:500:
barb:x:501:
carl:x:502:
dave:x:503:
donna:x:504:
```

```
grpadmin:x:550:al,barb,carl
grpall:x:551:al,barb,carl,dave,donna
```

5.1.2 Useful Linux commands

There are a few Linux commands you'll be using regularly. Here's a compact list of what they are and typical examples of each. These aren't exhaustive explanations or tutorials; rather, they're terse reminders of the name and syntax of each command. These are all performed as root.

- useradd - adds a Linux user to the system.

```
[root@jeeves ~] useradd -d /home/users/al al
```

to create user 'al' and create his home directory of /home/users/al.

- userdel - deletes a Linux user from the system.

```
[root@jeeves ~] userdel al
```

deletes al and all of the files in his home directory.

- usermod - adds a Linux user to an existing Linux group.

```
[root@jeeves ~] usermod -G differentgroup al
```

makes al a member only of 'differentgroup'

- passwd - changes the password of a Linux user.

```
[root@jeeves ~] passwd al
```

prompts you (and requires confirmation) for a new password for al

- groupadd - adds a Linux group to the system.

```
[root@jeeves ~] groupadd -f grpdev
```

creates a new group named 'grpdev' but warns if there's already a group with that name.

- chown - changes the owner of a file or directory.

```
[root@jeeves ~] chown al -R testfile.txt somedir
```

changes the ownership of testfile.txt to al, and the ownership of the somedir directory (and everything underneath it)

- chgrp - changes the group ownership of a file or directory.

```
[root@jeeves ~] chgrp grpdev -R /home/dev
```

changes the group ownership of the /home/dev directory (and everything underneath it) to grpdev

- chmod - changes the permissions of a file or directory.

```
[root@jeeves ~] chmod 750 somefile.txt
```

changes the permissions of the somefile.txt file so that the owner can read/write/execute, the group members can read/execute, and all others can't do anything.

5.1.3 Useful Samba files

Samba uses a number of configuration files. The ones we'll be using are:

- /etc/samba/smbpasswd - contains the names of Samba users. It looks something like this:

```
al:500:5FJEI28M9SASL39S49:7223F9SDWIMNZSF93LFSS35D51D6:[UX         ]:LCT-54133534:
barb:501:6K3SF893KSHJDF0982:AFSD875NSF983S932JR09F3LS9DF:[U          ]:LCT-51134534:
carl:502:3945MF0943J09DF08L:0239JFS0S9DS9D823K5J018SFA78:[U          ]:LCT-5120FEEW:
```

- /etc/samba/smb.conf - contains the configuration information for Samba. You'll be using this file a lot, particularly to set up Samba shares. The default version is over 300 lines long, so I'm not going to reproduce it here.
- /var/log/samba -> contains multiple log files that can be useful for debugging.

There's another one called /etc/samba/smbusers - while it sounds useful, we won't be using this file.

5.1.4 Useful Samba commands

There are a number of Samba commands that you'll use now and again. These are executed as root.

- /etc/rc.d/init.d/smb stop - stops the Samba server program
- /etc/rc.d/init.d/smb start - restarts the Samba server program
- /sbin/service smb restart - combines the stop and restart commands into one command
- /etc/rc.d/init.d/smb reload - causes the Samba server to reload the settings from the Samba configuration file (smb.conf) without restarting the Samba server itself.

As a side note, you can also stop and start the Samba service via the GUI if you've got the GUI installed on the server. Open the Service Configuration window via the Redhat | server settings | service menu. Then select the 'smb' service by clicking on the square symbol to turn on the check mark. The service will be highlighted. You can then click on the Start button to start the service if it wasn't started, or on the Restart button to stop and start the service.

5.2 Install Samba

Samba isn't automatically installed during the installation of a Linux distribution. If you're using a mainstream distribution like Fedora Core, RedHat, SuSE or Mandrake, and you have the GUI installed, you can look in the list of available services. If Samba is installed, you'll see an entry like "Samba" or 'smb' displayed. On Fedora Core and RedHat, the list of available services can be found via the Redhat | Server Settings | Service menu as noted a few paragraphs ago. In Mandrake, the list of available services is found via the Star | System | Configuration | Configure Your Computer menu option. Click on the System icon, and then the Services icon to display the list.

If you're using an RPM-based distribution, you can issue the command

```
[root@jeeves ~] rpm -q samba-server
```

from the command window as root. If the samba-server package is installed, no feedback will be presented and another command prompt will display. If samba-server is not installed, you'll get feedback like so

```
[root@jeeves ~] rpm -q samba-server
package samba-server is not installed
```

If not installed, it's generally easiest to use the GUI to install Samba, but you can also do so through a command like

```
[root@jeeves ~] rpm -i samba-server
```

You'll want to read the man page for rpm (or the documentation that came with your distribution) for additional options.

5.3 Make sure the Samba server is running

Once installed, you'll want to make sure the Samba server is running. You can check the GUI service configuration as described in section 4.2, or you can open up a command window and use the 'ps' command as shown in the following listing (most of which has been cut out):

```
[root@jeeves ~] ps -aux
<snip>
root      2959  0.0  0.1  5788 2028 ?        S    18:29   0:00 smbd -D
root      2963  0.0  0.1  4608 1768 ?        S    18:29   0:00 nmbd -D
root      2966  0.0  0.0  2640  708 pts/0    R    18:30   0:00 ps -aux
```

If you see the services named 'smbd' and 'nmbd' listed, Samba is running.

5.3.1 Start the Samba service if not running

If Samba isn't running, switch to root and issue the start command:

```
[root@jeeves ~] /etc/rc.d/init.d/smb start
Starting SMB services:                                     [  OK  ]
Starting NMB services:                                     [  OK  ]
[root@indy ~]
```

By the way, if you don't do this as root, you'll just get another command prompt, without any confirmation that anything happened (or didn't happen.)

You can also do this with the Service Configuration widget in the GUI, as described in section 5.1.4.

5.4 Configure your machine to start up automatically

You'll probably want Samba to be started every time your machine is started. Installation of Samba should have installed an 'smb' entry in /etc/rc.d/init.d, as shown in the following listing (look at the bottom of the fifth column):

```
[root@jeeves /etc/rc.d/init.d] ls
anacron     functions  kdcrotate  network  random      snmpd      ypbind
apmd        gpm        keytable   nfs      rawdevices  snmptrapd
atd         halt       killall    nfslock  rhnsd       sshd
autofs      httpd      kudzu      nscd     saslauthd   syslog
crond       iptables   lisa       ntpd     sendmail    winbind
cups        irda       named      pcmcia   single      xfs
firstboot   isdn       netfs      portmap  smb         xinetd
```

If this is the case, you can configure which runlevels will automatically start the smb service via this entry. First, you'll see what the existing runlevels are, and then change them to your liking if they're not already set up the way you want.

5.4.1 See what the existing startup runlevels are

Switch to root, because you're going to be changing files in the /etc directory hierarchy. Then issue the “chkconfig” command, as shown here:

```
[root@jeeves ~] /sbin/chkconfig --list smb
smb             0:off   1:off   2:off   3:off   4:off   5:off   6:off
```

This output shows that Samba is not configured to start up in any of the runlevels.

5.4.2 Turn certain runlevels on

In order to change the 'off' setting to on for runlevels 3 and 5, issue the following command.

```
[root@jeeves ~] /sbin/chkconfig --level 35 smb on
```

And then confirm that they're set up properly:

```
[root@jeeves ~] /sbin/chkconfig --list smb
smb             0:off   1:off   2:off   3:on    4:off   5:on    6:off
```

5.4.3 See the symlinks that have been created in rc0.d, rc3.d and rc5.d:

The /etc/rc0.d directory now has a symlink named K35smb which executes during normal shutdown, like so:

```
[root@jeeves /etc/rc0.d] ls
K03rhnsd      K15httpd     K44rawdevices  K74ntpd     K92iptables
K05anacron    K20nfs       K45named       K75netfs    K95firstboot
K05atd        K24irda      K50snmpd       K80random   K95kudzu
K05keytable   K25sshd      K50snmptrapd   K86nfslock  K96pcmcia
```

```
K05saslauthd  K30sendmail  K50xinetd      K87portmap  S00killall
K10cups       K35smb       K60crond       K88syslog   S01halt
K10xfs        K35winbind   K72autofs      K90network
K15gpm        K36lisa      K74apmd        K91isdn
```

Runlevel 3 has a symlink that points to smb named S91smb:

```
[root@jeeves /etc/rc3.d] ls
K05saslauthd  K50snmptrapd  S14nfslock   S56rawdevices  S91smb
K15httpd      K95firstboot  S17keytable  S56xinetd      S95anacron
K20nfs        S05kudzu      S20random    S58ntpd        S95atd
K24irda       S08iptables   S24pcmcia    S80sendmail    S97rhnsd
K35winbind    S09isdn       S25netfs     S85gpm         S99local
K36lisa       S10network    S26apmd      S90crond
K45named      S12syslog     S28autofs    S90cups
K50snmpd      S13portmap    S55sshd      S90xfs
```

as does runlevel 5:

```
[root@jeeves /etc/rc5.d] ls
K05saslauthd  K50snmptrapd  S14nfslock   S56rawdevices  S91smb
K15httpd      K95firstboot  S17keytable  S56xinetd      S95anacron
K20nfs        S05kudzu      S20random    S58ntpd        S95atd
K24irda       S08iptables   S24pcmcia    S80sendmail    S97rhnsd
K35winbind    S09isdn       S25netfs     S85gpm         S99local
K36lisa       S10network    S26apmd      S90crond
K45named      S12syslog     S28autofs    S90cups
K50snmpd      S13portmap    S55sshd      S90xfs
```

5.5 Tweak security in smb.conf

Depending on your network configuration, you may want to restrict access to the file server. One easy way to do this is through a setting in the /etc/samba/smb.conf. In the [global] section, restrict access to everyone with an "ALL" setting in the 'hosts deny' line, and then explicitly permit access from specific IPs. For example, to allow access only from computers on the local network, set the choices like so:

```
[global]
  hosts deny = ALL
  hosts allow = 192.168.1. 127.
```

This prevents anyone except localhost or users on 192.168 boxes from accessing the server.

5.6 Set up a Linux user on the server

We're going to add a couple of users to the server with the useradd command. Before we do, however, we need to create a directory under '/home' where all of the user's home directories will go. Otherwise, the useradd command will create user directories right under '/home', such as '/home/al' and '/home/barb'. Create a directory under '/home' called 'users', like so:

```
[root@jeeves ~] mkdir /home/users
```

Then, use the useradd command to add 'al' to the Linux server, indicating that his home directory will be '/home/users/al':

```
[root@jeeves ~] useradd -d /home/users/al al
```

Similarly, add 'barb':

```
[root@jeeves ~] useradd -d /home/users/barb barb
```

5.7 Create a Samba user for each Linux user

Now we're going to create users that Samba knows about. Issue the smbpasswd command, like so:

```
[root@jeeves ~] smbpasswd -a al
```

You'll be prompted for a new SMB password. Enter the same password that you used to set up al in section 5.6 (his actual Linux account password), and then enter it again when prompted. Once you're done, a new entry will have been added to the /etc/samba/smbpasswd file, as shown in section 4.1.3. Here's an example:

```
[root@jeeves ~] smbpasswd -a al
New SMB password:
Retype new SMB password:
unable to open passdb database.
Added user al.
[root@jeeves ~]
```

Again, the New SMB password is the same as the Linux user password. Note that the stupid "unable to open passdb database" line is standard.

5.8 Verify that 'homes' directory entry has been made in smb.conf

One purpose of the Samba configuration file, /etc/samba/smb.conf, is to identify and configure Samba shares - directories on the server that can be seen from workstations. The end of smb.conf has a number of examples that show how to create directory and printer shares; a typical example looks like this:

```
;[fredsdir]
;   comment = Fred's Service
;   path = /usr/somewhere/private
;   valid users = fred
;   public = no
;   writable = yes
;   printable = no
```

As you can imagine, it would be inconvenient and rather time-consuming to have to create entries like this for every user's home directory on the server. The Samba designers anticipated this and provided for a single [homes] entry that automatically creates shares for every Samba user (those users created in section 5.7). The entry should look like this:

```
[homes]
  comment = Home Directories
  browseable = no
  writeable = yes
  valid users = %S
  create mode = 0664
  directory mode = 0775
```

Again, this entry automatically 'shares' each user's home directories when they log in. That means you don't have to manually create an entry in smb.conf to share /home/users/al, and then a second entry in smb.conf for /home/users/barb, etc. The [homes] entry in smb.conf means that when al connects to the server (via a mechanism we haven't talked about yet), his /home/users/al directory will be available to him via the name 'al', while barb's '/home/users/barb' share will be available to her on her workstation as 'barb'.

If this [homes] entry isn't in your smb.conf file, put it in there now.

5.9 Create a share on the server that will be shared by all

The next step is to create and share a directory on the server that will contain data that an entire group will have access to. For example, the /home/admin directory will be accessible by all of the members of the grpadmin group (al, barb and carl.)

5.9.1 Create the directory

Enter the command, as root, to create the /home/admin directory:

```
[root@jeeves ~] mkdir /home/admin
```

Then change the group ownership of the directory so that the 'grpadmin' group is the group owner, like so:

```
[root@jeeves ~] chgrp grpadmin /home/admin
```

And change the permissions so that both the owner and group owner can read, write and execute, like so:

```
[root@jeeves ~] chmod 775 /home/admin
```

When you do a directory listing of /home, you should see something like this:

```
[root@jeeves home]$ ls -al
drwxr-xr-x    9 root     root         4096 Aug 12 14:07 .
drwxr-xr-x   19 root     root         4096 Aug  8 17:32 ..
drwxrwxr-x   11 root     grpadmin     4096 Aug 12 11:53 admin
```

For ease of testing, I always put a sample file in a new directory, like so:

```
[root@jeeves ~] touch /home/admin/testfileinADMIN.txt
```

5.9.2 Share the directory

Create an entry in the /etc/samba/smb.conf file like so:

```
[admin]
   comment = shared administration directory
   path = /home/admin
   valid users = @grpadmin
   public = no
   writable = yes
   printable = no
```

Remember that al, barb and carl are all members of the 'grpadmin' group. The '@' character in front of the name of the group in the 'valid users' line indicates that 'grpadmin' is a group and not the name of an individual Samba user. The word in brackets ('[admin]') in the first line is the name of the share. When connecting to it from a client, you just need that word, not the name of the entire path. You don't have to give it the same name as the directory itself, but doing so can help keep multiple shares straight.

5.10 Reload Samba

Finally, once you're done changing the configuration file, be sure to have Samba reread the configuration file with the reload command, like so:

```
[root@jeeves ~] /etc/rc.d/init.d/smb reload
Reloading smb.conf file:                                   [  OK  ]
[root@jeeves ~]
```

6. Set up Linux workstation to connect to server

Now that you're done with the server, it's time to configure a workstation. First, we'll create a mount point on the workstation, and then we'll issue a command that mounts the server's shared directory on the mount point on the workstation. As mentioned earlier, the mount point on a Linux workstation functions similarly to a mapped drive on a Windows workstation. Since Linux doesn't have drive letters, directories are used instead. In other words, on the workstation, the mount point is simply a directory through which the files on the server's share will be accessible.

Let's start with a mount point for al's home directory (located on the server) on al's workstation, and then create a second mount point for the admin share (located on the server).

6.1 Naming conventions for mount points

The first time you create a mount point on a client, it's tempting to just give the directory 'any old name', thinking the choice will be obvious to the user. However, if you end up creating mount points for more than one or two shares. For example, if you see directories named “home”, “data”, and “personal”, is it clear which one is the share on the server? And if you're creating mount points for shares that are on more than one server, it can quickly get confusing which share is which. You may want to impose some rhyme and reason to your naming scheme.

I start the name of the directory that will act as a mount point with an abbreviation for the server. Then I'll use common terms that identify which share the mount point belongs to. For example, in this scenario, I could get away with a generic 'svr' abbreviation for the server, and then descriptions of the data past that, like so:

```
svrhome
svradmin
svrmusic
```

If there were multiple servers in this network, a better convention might be

```
alfredhome
alfredmarketing
jeevesadmin
jeevesmusic
```

where 'alfred' and 'jeeves' were the servers (get it?) and the rest of the names described the data. Your preferences and style may vary, but if you don't have a convention, you might think about developing one.

6.2 Create a mount point for the home share

On al's workstation, issue the mkdir command. You can do this as al.

```
[al@ws1 ~] mkdir /home/al/svrhome
```

6.3 Issue mount command for the home share

As root, issue the mount command, like so:

```
[root@ws1 ~] mount -t smbfs -o username=al,password=secret //192.168.1.12/al /home/al/svrhome
```

This command mounts a user's home directory located on the server (192.168.1.12/al) to a mount point on the client (/home/al/svrhome), using the Samba user's username (al) and password (secret). A home directory share takes the username of the login and automatically shares the home directory for that user on the server.

6.4 Create mount point for the admin share

Issue the mkdir command again to create the admin share's mount point.

```
[al@ws1 ~] mkdir /home/al/svradmin
```

6.5 Issue mount command for the admin share

And then switch to root and issue the mount command.

```
[root@ws1 ~] mount -t smbfs -o username=al,password=secret //192.168.1.12/admin /home/al/svradmin
```

6.6 Test the shares

Now that the server shares are mounted on the workstation's mount points, it's time to test them. On the client, navigate to one of the mount points, either through the command window or through a graphical file manager, such as Konquerer or Nautilus. Trying modifying the file you placed there in section 4.9.1, creating new files and deleting them.

6.7 Mount the shares automatically

To mount a Samba share automatically, place mount commands in the file /etc/fstab on the Linux workstation. The /etc/fstab file controls the mounted filesystems for the Linux computer. If you've never modified fstab before, you might consider making a backup copy of this file before making changes to it, like so:

```
[root@jeeves ~] cp -i /etc/fstab /etc/fstab.backup
```

Now add an entry (on a separate line) to the file, like so:

```
//192.168.1.12/al /home/al/svrhome smbfs username=al,password=secret
```

For the admin share

```
//192.168.1.12/admin /home/al/svradmin smbfs username=al,password=secret
```

(You would use al's actual password instead of the word 'secret', of course.) Now, open a graphical file manager like Konqueror or Nautilus as al. You'll see a number of directories under /home/al, as shown in **Figure 1**.

Figure 1. Mounted shares appear in a graphical file manager just like all other directories on the machine.

As you can see, it's easy to identify which directories represent mount points to shares on the server.

7. Troubleshooting

There are a number of great troubleshooting guides available on the Web if something goes wrong. Two that I've found very useful are:

http://www.oreilly.com/catalog/samba/chapter/book/ch09_02.html
http://ranger.dnsalias.com/mandrake/muo/connect/csamba4.html

8. Where to go for more information

Samba is large and sophisticated, and there are many resources available. In addition to the help files (type 'smbpasswd –help', 'man smbpasswd', 'info smbpasswd', for example), the Samba website is at

http://www.samba.org

The O'Reilly & Associates book, Using Samba (Eckstein, Collier-Brown and Kelly) is also an excellent resource.

9. About the author

Whil Hentzen started out life in the early '80's as a custom software developer using dBASE II (he still has the original 8 1/2 x 11 grey binder of documentation, much to the chagrin of his wife), and switched to FoxPro in 1990. Besides billing 15,000 hours in the 90's, he presented more than 70 papers at conferences throughout North America and Europe, edited FoxTalk, Pinnacle Publishing's high end technical journal for 7 years, hosted the Great Lakes Great Database Workshop since 1994. He's written 7 books and published 30 more on a variety of software development topics. He was a Microsoft Most Valuable Professional from 1995 through 2003 for his contributions to the FoxPro development community, and received the first Microsoft Lifetime Achievement Award for Visual FoxPro in 2001.

Whil began using Linux on the desktop when OpenOffice.org became a standard in the mainstream distributions, as it spelled potential for custom application development in the future, and has been a Linux user, developer, and evangelist ever since. His first book on Linux, Linux Transfer for Windows Power Users, was published in early 2004.

He is available for new and legacy Visual FoxPro application development as well as Web and desktop development on Linux.

10. A word from our sponsor

This free whitepaper is published and distributed by Hentzenwerke Publishing, Inc. We have the largest lists of "Moving to Linux", OpenOffice.org, and Visual FoxPro books on the planet.

We also have oodles of free whitepapers on our website and more are being added regularly. Our Preferred Customer mailing list gets bi-monthly announcements of new whitepapers (and gets discounts on our books, first crack at special deals, and other stuff as we think of it.)

Click on "Your Account" at www.hentzenwerke.com to get on our Preferred Customer list.

If you found this whitepaper helpful, check out these Hentzenwerke Publishing books as well:

Linux Transfer for Windows® Network Admins:
A roadmap for building a Linux file and print server
Michael Jang

Linux Transfer for Windows® Power Users:
Getting started with Linux for the desktop
Whil Hentzen