CPU
COMPUTER POWER USER

Tips & Tricks

Click To Print

Tips & Tricks
February 2008 • Vol.8 Issue 2
Page(s) 98-99 in print issue

Warm Up To Penguins

Do The Samba

Because it can run on very low-end hardware, even fanless systems, a Linux-driven centralized file server lets you focus your budget on fat drives. But you'll need to set up your Linux box for Microsoft's SMB protocol.

SMB is what you're using when you poke around Windows' Network Neighborhood. It also lets you share printers across systems. Starting in the early '90s, the Samba toolset has let Linux systems join the party. But until recently, configuring Samba was a bit of an arcane art.

One reason that Samba setup can be difficult to master is that it can do so many things. In addition to operating as a file and print server (and client), it can also act as a PDC (Primary Domain Controller) or secondary controller in a domain-based network infrastructure. Domain-based logins are normally found in large companies; they provide (among other things) single sign-on to all the computers in the domain.

One of the first signs of hope for the everyday user was the development of SWAT (Samba Web Administration Tool). With SWAT, you can configure the smb.conf file (which initializes Samba) via a more user-friendly Web interface on the local server rather than using vi.

These days, Samba configuration tends to be built into a distribution's GUI-based system administration tools. In this article, we'll look at how to set up a simple set of network shares using openSUSE 10.3. All the distros handle it a bit differently, but the basic concepts are constant.

Simple Setup

In the example, we have a local workgroup-based network called BlackBear-Soft. We want to let users access their home (~username) directories from Windows. To do this, we used openSUSE's YAST (Yet Another Setup Tool). You can find the Samba Server setup under the Network Services tab.

The first thing to specify is the domain or workgroup name to join. If the Linux system is the first you'll set up, any name should work. Otherwise, use the same name that the Windows systems use.

On the next screen, YAST asked us if we wanted to set this server up as a PDC. For our simple home network, the answer to this is almost certainly "No." Then it presented us with a window with three tabs. The first tab (Start-Up) lets you specify whether you want to manually start the Samba server or let it load when the system boots. If you want your shares available all the time (even if the system reboots), you'll want to opt for automatic startup during boot. It also asks if it can open a port in the Linux firewall for the SMB port; you should also check this.

The next tab, Shares, defines what directories on the system will be shared and who can access them. If you've ever set up a Windows directory as shared, you'll find this a bit familiar. For our purposes, we checked the Allow Users To Share Their Directories checkbox.

Finally, the Identity tab lets you change the workgroup name and give your computer a different name on the Windows network.

You can probably leave this as is. Then you can click the Finish button, and you're in business.

Well, almost in business. Even though you've set Samba up to offer user directories, you haven't created SMB passwords for those users. You might think that your Linux user password would work, but Samba requires you to maintain a separate (although not necessarily unique) password for SMB access. To add them, use the smbpasswd program.

The easiest way from the command line is as root, or using sudo. At the command line, we typed **sudo smbpasswd –a james** and pressed ENTER. (The "-a" tells smbpasswd we're adding a user.) Next, we set a password and confirmed it, and smbpasswd informed us we successfully added a user.

Open-Source Shares

To access your shares from a Windows PC, open Explorer and type the IP address of the server in the Address bar. (In our example, it was 192.168.1.217.) A dialog box should prompt you for a username and password; this is what we just created from the openSUSE command line. When you log in, you'll see the shares available on the system. On openSUSE, if you take the defaults, you'll see folders labeled Groups, Profiles, Users, and Printers. For our purposes, the one you want is Users. Double-click it to open your home directory on the Linux system.

There's a lot more on the server side of Samba. You can share printers hooked up to the Linux box. You can set up publicly accessible shares or private shares to non-user directories. You can also set up read-only shares. But now, let's look at how you can go the other way, accessing Windows shares from Linux.

Before you can do this, you need to map the Windows server's IP address to its name, using the file /etc/samba/lmhosts. In this file, you put IP addresses and the host names they translate to. So to add a server at 192.168.1.201 named "blackbearpc," you'd add a line that reads "192.168.2.201 blackbearpc."

Now you're ready to poke around. To do this manually, use smbclient. If you've ever used command-line FTP clients, you'll be comfortable with smbclient. To begin, we can list all the shares and services available on the remote host, using the -L option. At the command line, we typed **smbclient –U james –L //blackbearpc/** and pressed ENTER. We entered our password and received the following output:

```
Domain=[BLACKBEARPC] OS=[Windows Vista (TM) Home Premium 6000]

Server=[Windows Vista (TM) Home Premium 6.0]
```

Sharename	Type	Comment
ADMIN$	Disk	Remote Admin
C$	Disk	Default share
H$	Disk	Default share
IPC$	IPC	Remote IP
Shared Disk		

If your Linux and Windows usernames are identical, you can leave out the "-U <username>" part. The last part of the command is the path to the server, but the backslashes you normally associate with Windows are just slashes in Linux.

If you see a share (conveniently labeled "Shared" in our example), open it with a similar command. In our example, we typed **smbclient –U james //blackbearpc/Shared** and pressed ENTER. After the initial text output, we typed **ls** at smb: \> and received the following output:

```
. D 0 Tue Nov 27 20:52:25

2007
```

```
.. D 0 Tue Nov 27 20:52:25

2007



58311 blocks of size 8388608. 9152 blocks available
```

There's not much there at the mo-ment, so let's put a file (let's say "stuff.txt") in it. We typed **put stuff.txt** and pressed ENTER.

Of course, moving stuff around with the command line is clumsy, so what you really want to do is mount (running the program mount, appropriately) it as a local drive. At the command line, wetyped **sudo mount –t cifs –o username=james //192.168.1.201/shared /mnt/shared** and pressed ENTER. We entered our password, and then typed **ls/mnt/shared** and pressed ENTER to receive the following output:

```
stuff1.txt stuff.txt
```

CIFS (Common Internet File System) is the new Microsoft name for SMB. The "-o username=<username>" is important because otherwise mount will use root as the username it passes to the Windows server. Strangely, mount uses /etc/hosts to resolve the hostname rather than /etc/samba/lmhosts, so you'll have to add your server there, as well, or use the IP address. You can also browser remote Windows shares from Nautilus using the format:

```
smb://192.168.1.201/
```

You can also use the /etc/fstab file to request that the share be mounted whenever the system boots, although that may delay the boot process a bit if the server is down.

We've just scratched the surface of what you can do with Samba, and you'll have to make some adjustments for your particular distribution. But Samba's the key to have your Linux and Windows PCs dancing to the same tune.

by James Turner

Infinfite Loop: Would You Like Cheese On Your Prokaryote Hoagie?

You've probably heard of the "soup" theory of the origin of life, and biology buffs are probably familiar with the "pizza" theory, too. But a biochemist at the University of California, Santa Barbara, believes that life may have begun under more high-carb conditions. Helena Hansama proposed the "sandwich" theory: Thin sheets of mica ("which look like the bread") may have created a shelter to protect developing single-cell organisms from the elements. Thankfully, Jared Fogle wasn't around to literally devour all life be-fore it could propagate.

Source: news.nationalgeographic.com/news/2007/12/071210-life-origins.html