

# Build a Linux test network

Skill Level: Introductory

[Carla Schroder \(carla\\_sg@bratgrrl.com\)](mailto:carla_sg@bratgrrl.com)

Linux administrator and author

Independent

19 May 2003

This tutorial shows how to combine Samba and GRUB to build a compact, highly adaptable, cross-platform test network, capable of booting and networking a large number of operating systems on a small number of machines. Though Samba and GRUB can manage many different operating systems, this tutorial focuses on Linux® and Windows®.

## Section 1. Before you start

### About this tutorial

This tutorial shows how to combine Samba and GRUB to build a compact, highly adaptable, cross-platform test network, capable of booting and networking a large number of operating systems on a small number of machines. Though Samba and GRUB can manage many different operating systems, this tutorial focuses on Linux and Windows.

This is neither a networking tutorial, nor a Linux system administration tutorial. Basic knowledge of running Linux and Windows, including user authentication, installing operating systems, partitioning, and managing user accounts will get you going a lot faster. I'll use values that are valid for my system, such as fd0 and hda. I trust that you will use what is correct for your system.

### Prerequisites

You'll need reasonably modern hardware: PCs four years old and newer ought to do the job. Running multiboot systems requires large hard drives, and support for Large-Block Addressing (LBA). GRUB can read any part of a hard disk supported by true LBA. Unfortunately, a small number of motherboards that claim to support LBA do not, and the only way to find out which ones they are is to try to boot a system from beyond the 1024 cylinder limit.

You also need a generic Linux boot/rescue disk, such as tomsrtbt, or H. Peter Anvin's SuperRescue CD, and a Windows 98 rescue disk, which is the all-time most useful Windows disk. GRUB does not yet have the ability to boot a CD; we still need floppy disks.

---

## Section 2. Solutions for cross-platform developers

### Using Samba and GRUB

Cross-platform developers face interesting challenges for building testing environments. In my ideal happy world, computer labs are huge, stocked with every little gadget a person might ever need -- even a Segway to scoot around on. But, as we are all painfully aware, resources are more limited in the real world.

One good option is to use excellent programs like VMWare or User-Mode Linux. These create virtual environments for running several operating systems side-by-side on a single PC. (See the [Resources](#) for links to more info on VMWare and User-Mode Linux.)

But for users who prefer completely native environments, the open source/free software world gives us two great tools for packing a lot of operating systems into a small space: Samba and GRUB.

**Samba** enables file and printer sharing between different operating systems. It can be a primary domain controller or a stand-alone server. It is primarily used to network Linux, UNIX, and Windows. Macintosh, OS X, OS/2, and other platforms are supported with varying degrees of tweakage.

**GRUB**, the Grand Unified Bootloader, is a most remarkable program. GRUB is capable of booting nearly any OS, and nearly any number of them. This tutorial will focus on multibooting and networking Linux and Windows.

### Testing in native environments

I use a three-PC test network. Thanks to the combined magic of Samba and GRUB, there is enormous flexibility; you can:

- Test applications and application servers (in native environments)
- Test all manner of networking configurations
- Use removable drive trays to swap in and out as needed
- Boot bare kernels, passing in parameters and modules from the GRUB command line

You have probably already spotted the one flaw in this beautiful scheme: the inconvenience of rebooting (as only one OS per machine can run at a time). However, this is the only real drawback I have yet encountered, and I think that it is a small price to pay for having a host of native environments in such a compact space.

---

## Section 3. Working with GRUB

### Basic scheme

First set up the individual computers in the test network. Install all the operating systems, and get everything booting up correctly. Then set up networking.

Install Linux first: think of it as the "host booting" system. Don't use the whole hard disk! Leave room for all the systems you wish to install. Use native partitioning utilities; Windows has its own `fdisk`, as does Linux. Linux's `fdisk` is more powerful and featureful, and can be used to create Windows partitions. But it's tricky; see `man fdisk` for details. If a shared data partition is needed, format it in `fat32` or `fat16`, depending on what versions of Windows are running.

### GRUB

Rejoice, ye who have suffered under the tyrannies of the Master Boot Loader, or the fickleness of LILO, for the new kid on the block, GRUB, is robust, flexible, and helpful. GRUB can find any kernels anywhere. GRUB contains its own little command shell, for passing in or editing commands at boot time. It can read from a configuration file. It supports many filesystems, currently BSD FFS, DOS FAT16 and FAT32, Minix fs, Linux ext2fs, ReiserFS, and VSTa fs; and blocklists for files that do

not appear in filesystems, such as chainloaders.

GRUB reads filesystems and kernel executables, rather than inflexibly restricting the user to disk geometry. Install and remove operating systems as needed. Boot bare kernels, passing in modules and parameters from the command line. GRUB will even download OS images over the network. OK, enough talking, let's cook up some GRUB.

## Installing GRUB

Get comfortable and resign yourself to a lot of rebooting.

GRUB only runs on \*nix systems; I'm using Libranet Debian. Most newer Linux distributions come with GRUB. (If you already have it installed, run `locate grub` to find where all its files are, and make a hard copy of the list. Then skip below to Making a boot floppy.)

Be sure to have a rescue disk at the ready in case GRUB goes wrong. If you are running a system with a version of GRUB older than 0.9, you must upgrade. Antiques are good for collecting, not computing. To find the version, type:

```
$ /sbin/grub --version
grub (GNU GRUB 0.93)
```

This may lead to a system upgrade, as GRUB requires `binutils-2.9.1.0.23` or newer. Run `ld- v` to find out. If `binutils` is older than that, you have a way-too-old Linux. Best to grab a whole new Linux; it's easier and better. This is a test lab, so I'm assuming you're not messing with an important production system! If you upgrade GRUB, remove the old version first, keeping your `/boot/grub/menu.lst` file.

Get the latest GRUB from <http://www.gnu.org/software/grub/>. As of this writing, the current edition is `grub-0.93.tar.gz`. Debian users, of course, employ `apt-get`, and do not need to build from source. Everyone except Debian users must unpack the tarball:

```
$ zcat grub-0.93.tar.gz | tar xvf -
```

Check out `README`, `NEWS`, and `INSTALL`. Most likely, the generic install will suffice. Change to the directory where GRUB is unpacked, and run:

```
# ./configure
# make install
```

When that's finished, run `updatedb`.

Next you need to make a boot disk.

## Making a boot floppy

I quote from the official GRUB manual:

"To create a GRUB boot floppy, you need to take the files 'stage1' and 'stage2' from the image directory, and write them to the first and the second block of the floppy disk, respectively."

Run `locate /grub/i386-pc`, and change to that directory, then copy files to the boot floppy:

```
# cd /usr/lib/grub/i386-pc
# dd if=stage1 of=/dev/fd0 bs=512 count=1
# dd if=stage2 of=/dev/fd0 bs=512 seek=1
```

Here's a funny little gotcha: This disk is not mountable; it's a boot image. If you are in the habit of mounting disks to verify them, you'll get all kinds of warnings and error messages, and think your disk is bad. It's not.

## Installing GRUB into the MBR

Now reboot the system with your shiny new boot floppy. You will see something like:

```
GRUB version 0.93 (640K lower / 3072K upper memory)

[ Minimal BASH-like line editing is supported.  For the first word, TAB
  lists possible command completions.  Anywhere else TAB lists the possible
  completions of a device/filename. ]

grub>
```

Be not afraid, for GRUB is helpful. Type `help` for a menu of the most common commands; type `help --all` for all of them.

Continuing with our installation process, use the `find` command to locate the root device. `find` will then display the devices that contain the file:

```
grub>find /boot/grub/stage1
(hd0,0)
```

Now set the root device:

```
grub>root (hd0,0)
```

Enter the setup command:

```
grub>setup (hd0)
```

Now GRUB is installed into the MBR. Remove the floppy and reboot, and the pretty blue GRUB menu will greet you. Everything in GRUB is configurable, including the colors.

Go ahead and start up Linux; then run `fdisk -l` to display the partition table. Make a hard copy.

Important note: GRUB uses a different disk and partition numbering scheme than `fdisk`. GRUB numbers sequentially, starting from zero. So this:

```
/dev/hda1
/dev/hda2
/dev/hda4
```

translates to:

```
hd0,0
hd0,1
hd0,2
```

A second hard drive is `hd1`, and so forth.

---

## Section 4. Adding Windows

## Deceiving Windows

We must weave a web of deceit around Windows 95/98/ME, for it does not care to share the bootloader, and freaks out at the sight of too many partitions. Windows NT/2000/XP are not possessed of such tender sensibilities; still, we'll deceive them too, as they have their own little quirks. Besides, it's easy and fun.

Fire up the computer; when GRUB appears, hit `c` to get to the command line. Insert the Windows 98 rescue floppy. All the existing partitions must be hidden:

```
grub>hide (hd0,0)
grub>hide (hd0,1)
```

Because GRUB cannot boot DOS/Windows directly, we must invoke the chainloader:

```
grub>chainloader (fd0) +1
grub>boot
```

As the rescue disk is booting up, be sure to enable CDRom support. Use `fdisk` to create a primary DOS partition to install Windows 98 into. When that's done, insert your Windows 98 CD, and run Windows Setup from the prompt:

```
A:\>D:
D:\>setup
```

Windows 9x will overwrite the MBR, which is handy for getting through the many installation reboots. But where did Linux go? Fear not, when Windows is finished installing, simply install GRUB to the MBR just like we did before. Now the GRUB menu will come back, but without a Windows entry. Let's do like real hot GRUB commandos, and boot to Windows from the GRUB command line:

```
grub>hide (hd0,0)
grub>hide (hd0,1)
grub>root (hd0,2)
grub>makeactive
grub>chainloader +1
grub>boot
```

Feel the power!

## Editing menu.lst

It is not necessary to do this every time. The GRUB boot menu is configured in `/boot/grub/menu.lst`. Add this entry to `/boot/grub/menu.lst` to put Windows on the boot menu:

```
title           Windows 98
hide (hd0,0)
hide (hd0,1)
root            (hd0,0)
makeactive
chainloader +1.
```

GRUB is very flexible, and even when using `menu.lst` boot commands, it can be edited on the fly.

## Windows NT/2000/XP

Let's call it Win2k for short, because I like it that way. Follow these steps:

- Start up the computer with the Win2k installation media, hopefully a bootable CD, rather than all the installation floppies
- Win2k setup will not be alarmed at the presence of other operating systems or partitions; use Setup to partition and format the free space
- Go through all the endless reboots necessary until it is happy and complete
- Restore GRUB to the MBR yet again
- Add Win2k to `menu.lst`

## The final menu.lst

Your final `menu.lst` should look something like this (of course the drive and partition numbers will reflect your system):

```
title           Libranet GNU/Linux, kernel 2.4.19
root            (hd0,0)
kernel          /boot/vmlinuz-2.4.19 root=/dev/hda3 ro
savedefault

title           Libranet GNU/Linux, kernel 2.4.19 (single user mode)
root            (hd0,0)
```

```
kernel                /boot/vmlinuz-2.4.19 root=/dev/hda3 ro
savedefault

title                 Windows 98
hide (hd0,0)
hide (hd0,1)
unhide (hd0,2)
hide (hd0,3)
root                  (hd0,2)
makeactive
chainloader +1

title                 Windows 2000
hide (hd0,0)
hide (hd0,1)
hide (hd0,2)
unhide (hd0,3)
root                  (hd0,3)
makeactive
chainloader +1.
```

---

## Section 5. Adding more Linuxes (or Linuxi or Linuces)

### Installation

The steps are similar to installing Win2k:

1. Boot the computer with the installation media
2. Partition and format from the setup program
3. When it asks about bootloaders, **choose one** of these options:
  - Do not install one
  - Install it to the first /boot partition, not the MBR

After installation, rebooting takes you back to the GRUB menu.

### Finding boot parameters from GRUB

If you install one of those inconvenient Linuxes that require rebooting during installation, this is how to get back to it (as the boot menu will not magically update itself). Boot to the main GRUB menu, and hit `c` to get to the command line. To boot a second Linux that does not have its own bootloader, we must pass in the root and

kernel parameters. Type commands partway, then hit the Tab key for auto-completion:

```
#use tab-completion
grub>root (hd0,
Possible partitions are:
  Partition num: 0, Filesystem type is reiserfs, partition type 0x83
  Partition num: 1, Filesystem type unknown, partition type 0x92
  Partition num: 3, Filesystem type unknown, partition type 0x17
  Partition num: 4, Filesystem type is ext2fs, partition type 0x83

grub>root (hd0,4)
Filesystem type is ext2fs, partition type 0x83

#use tab-completion
grub>kernel /boot/vmlinuz
Possible files are: vmlinuz vmlinuz-2.4.19

grub>kernel /boot/vmlinuz-2.4.19 root=/dev/hda5
grub>boot
```

And there you are. Mind the partition numbers! Always remember (or at least, never forget) that -- for instance -- `hd0,4 = hda5`.

## Chainloading Linux

If you installed a bootloader to the first `/boot` partition on Linux #2, set the root device as described above, then:

```
grub>chainloader (hd0,4)+1
grub>boot
```

Edit `menu.lst` accordingly; remember it does not need the `boot` command.

No installation disks? A common method of installing \*nix systems, such as the BSDs, and many Linuxes, is to start up with Tom's Root Boot (`tomsrtbt`), or the SuperRescue CD (see the [Resources](#) for links to these); then partition, set up networking, and download the rest of the system files. It may seem overwhelming, but it is very simple once you get the hang of it -- and the techniques demonstrated in this tutorial will adapt nicely for this.

## Section 6. Working with Samba

### Connecting Windows clients

Samba is more than abundantly documented, so we won't rehash the basics. See the excellent tutorial "[Using Samba as a primary domain controller](#)". Two of the primary functions of my own little test network are fine-tuning Samba's abilities as a primary domain controller, and testing different uses as a stand-alone server.

Samba documentation is quite thorough on the server side, but not so complete on how to connect Windows clients. Each version of Windows implements networking, managing shared resources, and connecting to domains and workgroups differently. Windows 9x is the easiest to connect to anything; it's wide open. Windows NT4 and 2000 go well with Samba, as it is based on the NT4 domain model. Windows XP introduces some new, er, challenges.

### Common problems and fixes

#### **Password woes:**

Probably the most common problem for Samba admins is making passwords work correctly with Windows clients. These versions of Windows do not support encrypted passwords:

- Windows 95 pre-OSR2
- Windows NT 3x
- Windows NT 4 pre- SP3

Samba cannot accept both cleartext and encrypted passwords; you must choose one or the other. If you really want to mix these old things on a LAN with newer ones that use encrypted passwords, encryption must be disabled. Look in `/usr/share/doc/samba-doc/Registry` for files containing registry hacks to enable cleartext passwords for the various Windows. They look something like this:

```
[HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\VxD\VNETSUP]
"EnablePlainTextPassword"=dword:00000001
```

#### **Too many protocols:**

One common source of trouble is having too many network protocols installed on

Windows clients. Too many protocols means time wasted trying to make connections on the wrong ones. Install only what will be used.

**Slow performance:**

Set your NICs explicitly to their highest performance specs, such as "100 TX Full Duplex," rather than using Auto-Detect.

**Messed-up smb.conf:**

Rather than using SWAT, Webmin, or editing the sample smb.conf, start fresh with a clean new smb.conf. This will prevent many headaches, and will be easier to troubleshoot. One of the most useful Samba utilities is `testparm`, Samba's built-in smb.conf checker:

```
# testparm /etc/smb.conf
```

`testparm` will find syntax errors. This does not guarantee that the commands are good, only the syntax!

**Network configuration errors:**

Too big a topic to go into here; the good news is diagnosis is straightforward, because we are dealing with native environments, and have not introduced the complexity that comes with virtual environments.

**Windows NT 4/2000:**

When logging into a Samba PDC for the first time, be sure to log in as root. This will set up the machine account, and then other users on the machine can access the domain.

**Windows XP Home:**

XP Home has the newfangled Simple Sharing. Simple Sharing offers zero granularity: shared files are accessible to everyone. It's about at the level of sharing in Windows 9x, except that the Program Files directory, the Windows directory, and Administrative shares are not accessible. XP Home cannot join a domain, neither Windows nor Samba domains, so if domain access is needed, XP Home will not work.

## Windows XP Pro

XP Pro comes with the full complement of Microsoft networking protocols, minus one: Netbeui. If you really really want Netbeui on either version of XP, it is included in the CD-ROM of the full retail versions only. See q301041.

To join a Samba domain, XP Pro needs a bit of fine-tuning. (Same for a Windows domain, so don't feel picked on.) Just like everyone else who administers XP, I get

different results on different systems. Here are things to try:

-Sharing is not enabled by default, as it is in NT and 2000. Not even administrative shares. Turn on sharing. Right-click the drive, then choose "Sharing And Security". On the "Sharing" tab, there are scary warnings. Ignore them and proceed. On the "Local sharing and security- Network sharing and security" tab, configure your shares.

To log into a Samba domain:

- Turn off Simple File Sharing. Open Windows Explorer, Tools >Folder Options >View. SFS totally gets in the way of proper domain account and user management; it's a simpler model for simpler needs. It cannot be running if you want to join a domain.
- XP Pro by default is configured to the Windows 2000 domain model, which means it expects the signing and sealing of netlogon packets across the network. Since Samba emulates a Windows NT 4.0 PDC, which does not use this, it's registry hack time. Look for /usr/share/doc/samba-doc/Registry/WinXP\_SignOrSeal.reg, which looks something like this:

```
; This registry key (gathered from the Samba-tng lists) is needed
; for a Windows XP client to join and logon to a Samba domain
;

HKKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\netlogon\parameters
"RequireSignOrSeal"=dword:00000000
```

Some users report that ControlSet001 & ControlSet003 also need this key.

Some users report that the Local Policy setting in the Local Computer Policy > Computer Configuration > Windows Settings > Security Settings > Local Policies > Security Options > Network Security:Lan Manager Authentication level should be set to "Send LM & NTLM responses".

## Coming soon: Samba 3.0

Another fun use for our compact test network is to beta-test Samba 3, which adds all kinds of new features and improvements on existing features. It will support Unicode, trust relationships, and add substantial integration with Active Directory. Samba 2.2 has already proven its stability, scalability, and high performance. 3.0 promises to be even better.

# Resources

## Learn

- In "[Boot loader showdown: Getting to know LILO and GRUB](#)" (developerWorks, August 2005), review the pros and cons of these two popular boot loaders.
- The tutorial "[Getting to know GRUB](#)" (developerWorks, January 2001) shows how to install and use GRUB.
- Carla prefers GRUB to LILO. You can find [the GRUB Manual](#) online; here also are some [LILO resources](#) (via the project description at freshmeat.net).
- Other ways to run multiple operating systems include [VMWare](#) and [User-Mode Linux](#). See Carla's tutorial "[Introduction to User-Mode Linux](#)" (developerWorks, January 2003).
- You'll find everything you need on [Samba](#) at samba.org. Take a look at [SMB/CIFS Clients](#).
- See the excellent tutorial "[Using Samba as a primary domain controller](#)" (developerWorks, April 2002).
- Check out "[Understanding the Network Neighborhood: How Linux Works With Microsoft Networking Protocols](#)" (*Linux Magazine*, May 2001).
- Good rescue disks include Tom Oehser's Root Boot Disk (or [tomsrtd](#)) and H. Peter Anvin's [SuperRescue CD](#).
- See also [information and help with Microsoft DOS](#).
- [Windows Resource Kits](#) are indispensable for Windows admins.
- If there were no difference between open source software and free software, the two types of software wouldn't have two different names. Read the Free Software Foundation (or FSF) [Free Software Definition](#) or [History of Unix, Linux, and Open Source / Free Software](#) by David A. Wheeler (he has lots more worth reading on his site as well) for just a sampling of some of the more important differences.
- For basic system administration info, try the developerWorks [LPI certification tutorials](#). These will help you navigate and understand basic admin concepts and naming conventions for everything from drives and file systems to much, much more.
- Find more [tutorials for Linux developers](#) in the [developerWorks Linux one](#).
- Stay current with [developerWorks technical events and Webcasts](#).

## Get products and technologies

- [Order the SEK for Linux](#), a two-DVD set containing the latest IBM trial software

for Linux from DB2®, Lotus®, Rational®, Tivoli®, and WebSphere®.

- Download [IBM trial software](#) directly from developerWorks.

### Discuss

- Read [developerWorks blogs](#), and get involved in the developerWorks community.

## About the author

### Carla Schroder

Carla Schroder is a freelance PC tamer, administering Linux and Windows systems for small businesses, and writes how-tos for real people. Loves computers and high tech, thinks Linux/Open Source/Free Software is the best playground in the world. Carla discovered computers and high-tech in 1994; her first PC was an Apple II. She progressed through DOS/Windows, from 3.1 to XP. Discovered Linux in 1998. Carla is living proof that self-taught middle-aged ladies can be fine computer gurus. You can contact Carla [carla\\_sg@bratgrrl.com](mailto:carla_sg@bratgrrl.com).