Linux+Win9x+Grub HOWTO

Joel Rosen

Revision History

Revision v1.0	2002−01−10	Revised by: jr
Initial release.		

This HOWTO will tell you how to use the GRUB bootloader to turn your computer into a Windows + Linux dual boot machine, without affecting your current Linux installation.

Table of Contents

1. Introduction

1.1. Why use this method?

I'm a big fan of things that aren't Microsoft, so when I got my new computer I had it come with Linux preinstalled. It's a great desktop machine, but I've been sorely missing my Windows games, and the GIMP just isn't cutting it. So I finally gave in and decided to make the machine dual boot. When I looked for instructions on how to accomplish this, all I found was directions on how to install Linux if you already have Windows installed and how to boot with Loadlin. Unfortunately, because Loadlin runs from under DOS, the Loadlin method of making your system dual boot requires that you have Windows installed on the first partition of your first hard drive, due to the fact that Windows believes it is the center of the universe and all your other operating systems should revolve around it. This is a problem if you already have Linux on that partition and you don't want to wipe out your system to install Windows.

This is where GNU GRUB, the GRand Unified Bootloader comes in. It differs from bootloaders such as LILO in that it can lie to Windows and make Windows believe that it's installed on the first partition even if it's not. So you can keep your current Linux system where it is and install Windows on the side.

1.2. Copyright and License

2. Requirements

2.1. What are the requirements?

You will need:

- A computer with a working installation of Linux.
- Enough free disk space for your new Windows partition.
- A partition editor such as GNU Parted.
- GRUBrpm or source.
- A Windows/DOS boot disk with CDROM support and a CD with the Windows 9x install on it, or a bootable Windows 9x install CD.
- *(Recommended)*A Linux boot disk.

3. Procedure

3.1. Creating the Windows partition

If you already have a clear partition or disk for your Windows installation, you can skip this section. Otherwise, you'll need to run a partition editor to defragment your disk and create a new partition for Windows. I used GNU Parted for my installation; sfdisk should work as well. *DO NOT* use fdisk, it won't preserve your data. And note, this process is not guaranteed, make sure to backup any important files before you begin.

Here is how my partition table looked pre−install:

```
Device      Start       End         System

/dev/hda1   1           3           Linux
/dev/hda2   4           1222        Extended
/dev/hda5   4           36          Linux swap
/dev/hda6   37          1222        Linux
```

/dev/hda1 is my /boot partition (you may or may not have a separate boot partition for Linux), /dev/hda2 is an extended partition which contains two logical drives, /dev/hda5 and /dev/hda6, which contain my Linux swap partition and Linux installation partition.

As root, run your partition editor. Resize your Linux partition to however small you decide to make it. (Make sure this size isn't smaller than the amount of data on your disk.) The program will defragment for a good while and then change the partition table. Resize the extended partition which contains your Linux partition accordingly. You should now have enough unused space on your disk in which to create a new primary Windows partition. Go ahead and create the partition with a Win95 FAT32 (LBA) filesystem on it. Now your partition table should look something like this:

```
Device      Start        End      System

/dev/hda1   1            3        Linux
/dev/hda2   4            905      Extended
/dev/hda3   906          1222     Win95 FAT32 (LBA)
/dev/hda5   4            36       Linux swap
/dev/hda6   37           905      Linux
```

I strongly recommend that you run fdisk at this point and write down all of your partition information on a piece of paper. If you make a mistake in your installation, Windows will be all too happy to rewrite your partition table to its liking, and you'll need to know what your partitions look like in case of emergency.

3.2. Installing GRUB

3.2.1. Installation

You should have either an RPM or source package of GRUB somewhere on your disk. Go through the appropriate installation procedure for your system. GRUB should now be installed but not loaded. As root run:

```
# /sbin/grub-install /dev/hda
```

This will load GRUB into the master boot record of your disk. Now you need to edit /boot/grub/grub.conf to configure the GRUB boot menu:

3.2.2. Editing grub.conf

```
default=0
timeout=10
splashimage=(hd0,0)/grub/splash.xpm.gz
title Red Hat Linux (2.4.7-10)
        root (hd0,0)
        kernel /vmlinuz-2.4.7-10 ro root=/dev/hda6
        initrd /initrd-2.4.7-10.img

title Windows 98
        map (hd0,0) (hd0,2)
        map (hd0,2) (hd0,0)
        rootnoverify (hd0,2)
        chainloader +1

title DOS Boot Disk
        map (hd0,0) (hd0,2)
        map (hd0,2) (hd0,0)
        chainloader (fd0)+1
```

Let's go through and see what this means.

"default=0" and "timeout=10" will cause Linux to boot in 10 seconds if you don't touch anything. "splashimage" is self−explanatory and not very important.

"title" denotes each boot setting and the text that follows is what will appear in your menu at startup. "root" specifies which partition contains your Linux kernel image (this may or may not be your actual root (/) partition). So, "root (hd0,0)" tells GRUB that the kernel is on the first partition of /dev/hda. You can see that GRUB has a funny numbering system, 0−3 for primary partitions and 4+ for logical partitions. The next line tells GRUB just where to find your kernel and where your actual root partition with your Linux system is. "initrd" tells where your init ramdisk image is located. Note that these few lines will probably be somewhat different if you don't have a separate /boot partition. Look at the GRUB info page if you need more help.

The "map" lines under the Windows 98 section are essential for getting your installation to work. These are the magical lines that trick Windows into believing that it's installed on the first partition of the first disk. If you don't map the Windows partition to (hd0,0), Windows will destroy your partition table and you won't be able to boot anything.

"rootnoverify" tells GRUB to boot from the Windows partition, but not to attempt to mount it, and "chainloader +1" tells GRUB to chain to Windows' bootloader which will start Windows.

The "DOS boot disk" section is needed if you are going to boot from floppy to begin the Windows install, but it will also come in handy after your install if you ever want to use a DOS boot disk for whatever reason. If your Windows install CD is bootable, you'll need to have a "Windows boot disk" section which chainloads to whatever your CDROM device is called.

Now save grub.conf and exit your text editor. Unlike LILO, GRUB does not require you to run any executable after you've modified the boot configuration. Now, with your emergency Linux boot disk handy, try rebooting your computer. When the system comes up you should see the GRUB menu with your three boot options. Select Linux. Hopefully, Linux will boot as usual and you'll have installed GRUB properly. If it doesn't, you thankfully have a Linux boot disk so that you can boot into Linux and fix your grub.conf. Once you've got GRUB successfully booting Linux, it's time to install Windows.

3.3. Installing Windows

Remove all diskettes and CDs from your computer and reboot. *DO NOT boot your computer with the installation media already in the drive.* If you do that, your BIOS will just boot the Windows install directly and GRUB won't get a chance to work its magic. When the GRUB menu comes up, insert your floppy or CD and select the DOS boot disk or Windows install CD boot, depending on your installation method. Windows install should see the partition you've made for it on the C: drive. Go ahead and install Windows.

That's it. Your computer is now all set as a dual boot machine.