Managing Multiple Operating Systems HOWTO

Table of Contents

Managing Multiple Operating Systems HOWTO

Robert W. Schultz

v0.4, 17 Feb 2000

This HOWTO covers the procedures for using removable hard disks to install and manage multiple alternative operating systems while leaving a single fixed disk to permanently house and protect the primary operating system. It is very scalable and offers a good degree of protection to and a stable disk environment for the primary operating system.

7.Cost and gotcha's:

1.Introduction

1.1 Copyright

1.2 New Versions of this HOWTO

New versions of the Managing Multiple Operating Systems HOWTO will be available to browse and/or download at LDP mirror sites. For a list of mirror sites see:

http://metalab.unc.edu/LDP/mirrors.html.

Various formats are available. If you only want to quickly check the date of the latest version look at

http://www.linuxdoc.org/HOWTO/MultiOS−HOWTO.html

and compare it to this version: v0.4, 17 February 2000

1.3 Feedback

Please send any questions, comments, or suggestions to `mailto:rwschul@smart.net (Robert W. Schultz)`. I am very willing to help others with problems directly relating to this HOWTO and will entertain any suggestions for changes/modifications and/or improvements. However, having set up my system to my satisfaction, this HOWTO won't be significantly improved without reader input.

2.Purpose and goals:

The purpose of this HOWTO is to describe a methodology for managing multiple operating systems on a single computer system. It is intended for Linux users who have a basic familiarity with both Linux and LILO installations. Nothing here is terribly complex however, considering the amount of time it takes to install some operating systems it can be relatively time consuming.

It is different from other methods in that it doesn't require multiple operating systems on the boot disk. Instead it uses a fixed internal disk containing a single primary operating system and a selection of removable disks with one or more operating systems installed on each of them. If you absolutely have to have two, three, or four different operating systems on a single disk and are trying to get them to behave with each other, this HOWTO is not for you.

Its specific goals are:

* – A primary/operational disk that once configured and installed is rarely changed. This includes not having to repartition or otherwise modify the disk it resides on.

* – Easy selection of multiple alternative operating systems at boot time without having to go through more than one or two menu selections.

* – No need to modify the BIOS, LILO, or any other configuration to access an O/S once it is installed.

* – Easy addition/removal of operating systems depending on requirements at the moment.

* – Inexpensive and scalable to allow for an increasing number of operating systems and versions as time goes on.

I think I succeeded admirably in the first three of these requirements. You can form your own opinions on the last two.

3.Background:

Even though Linux is becoming much more user friendly and widely accepted, most of us still need access to other operating systems. I use Linux about 75% of the time but I still need access to Win 9x for those few applications that I haven't found acceptable alternatives for in Linux. My wife uses MS Office at work and wants to have access to it at home. Other people want to have access to alternative O/S just for fun, training, or to keep current in their job.

I tend to treat new O/S's as you would a game; I actually don't do anything productive with them but installing and learning how to control them is just as challenging and interesting to me as Quake or SimCity. Further, being a Computer Scientist, it keeps me current on evolving technology and has helped me solve a multitude of problems at work. At any rate, for new Linux users, computer professionals, and those just trying to migrate from one operating system to another, I believe using multiple operating systems is the norm rather than the exception.

4.System requirements:

BIOS − Any bios that allows automatic identification of disk drive geometry and allows you to select the sequence of devices to boot from should work. I successfully built systems based on both PhoenixBIOS 4.0 and AMI Plug and Play Flash BIOS.

DISKS − One fixed internal disk dedicated to Linux. (first disk) One Removable drive enclosure, with any number of drives. (second disk)

Since a lot of this HOWTO has to do with disks, from now on I will generally use the terms "first disk" and "second disk". The first disk is the one initially accessed when the machine is turned on, commonly known as the boot disk. It has LILO installed in the MBR and is dedicated to a single operating system, specifically Linux. The second disk is a removable disk that contains one or more alternative operating systems which may or may not have a boot loader in the MBR or elsewhere.

There are no other hardware/firmware requirements. Any other requirements would be dictated by the specific O/S. For instance, even though you could install it, Solaris 7 is not going to run well on an old 90MHz machine! The configuration and methodology described here should however work equally well regardless of the CPU speed or other installed peripherals.

Operating Systems − I have tested this process with Linux (Redhat and Suse), Solaris 7, BeOS, Win 98 and even MSDOS 6.22. I see no reason why it wouldn't work with Win 95, O/S 2, or FreeBSD. I am not familiar with Windows NT or 2000 so I don't know how they would react to this kind of setup.

Boot loader − I used LILO on the first disk and BeOS bootman on the second disks. I used LILO on the first disk because it was the only boot loader that allowed me to select the MBR on the second disk as an acceptable boot partition. Any relatively robust boot loader should work on the second disk.

5.How it works (a scenario):

Prior to ignition, I insert a disk, preloaded with an operating system in the removable drive. When I turn on the machine, I am presented at the LILO prompt (by pressing <tab>) with "Linux" and "Disk2" as options. Linux is the default and would automatically boot if I did nothing. When selected, Disk2 either boots directly into the single O/S stored on the second disk or presents me with a second boot menu if there is more than one O/S on that disk. If I power down, replace the second disk with another and power back up, I still get the initial Linux/Disk 2 menu and, if I select Disk2, a new menu appropriate to the newly inserted disk. Once installed, I never have to modify the LILO configuration on the first disk, I never have to change BIOS setting to boot from the second disk and I never have to go through more than two menu selections to get my selected O/S up and running.

6.The installation:

There are three distinct parts to the installation, first building the primary O/S on its own dedicated drive. Second, building a second disk with whatever alternative O/S you selected. Finally, reconfiguring the BIOS and LILO to support both disks.

6.1 Installing the primary operating system

Installation of the primary/operational O/S is fairly straight forward. Treat the system as if it were a single drive system dedicated to Linux. Refer to the documentation that came with your distribution or see http://www.linuxdoc.org/HOWTO/Installation−HOWTO.html for details on installing Linux.

Because this is ultimately a multiple disk installation, there a few steps that need to be taken to trick the install routines into thinking that it is, during the installation process, a single disk system.

First, Remove the removable Hard Drive and make sure the remaining drive is identified in the BIOS as the secondary boot device (after the floppy). The operating system install program should only see one disk, the one you are going to install to. That way, there is no question as to where it will be installed. Also, it will install everything appropriate to a single disk system.

When asked, tell the install program to use the entire disk for your operating system. I accepted the default RedHat partitioning and installed the generic LILO on the MBR.

Once the installation is complete, shutdown and reboot to confirm that your system works properly. At this point you should have a fully functional machine that boots directly into Linux.

6.2 Installing alternative operating systems

Now that you have a fully functional system, you can move on to building a second disk with your alternative operating systems.

Select an operating system or two for installation on the second disk. I decided on, for no good reason, Windows 98 and BeOS for my initial test case. I partitioned an 8GB drive into two 4GB primary partitions and installed Windows 98 in the first partition and BeOS in the second.

Do the same things with this install that you did with the first. Disable the first disk in the BIOS so that this installer will not even see it. This is very important. If you can physically remove or disconnect the first disk, do it! This will protect your primary system from any errors on your part or overly greedy operating systems that want to take over all the disks they see during the second installation. If at some time in the future you decide to create another removable disk make sure and repeat this step.

Once this is done install your chosen operating systems as if you were installing them on a single drive system.

If you are only installing one operating system on the second disk, just plug in the installation disk and let it do its thing. Windows 95 or 98 or just about any other operating system, including a second Linux should install just fine this way. Allow Windows 9x to write to the MBR. If installing Linux, select MBR as the location to install LILO.

I decided to install two operating systems on the second disk so that I could confirm the functionality of cascading boot loaders.

I first installed Windows 98 because it automatically overwrites the MBR and would have overwritten any boot loader code I eventually placed there. Next, I installed BeOS in the second partition and ran bootman, the BeOS boot loader. With it I built a boot menu for the second disk and intentionally overwrote the Windows 98 MBR.

Bootman was not essential, I could have used any MBR based boot loader but it was available and it works quite will.

Reboot frequently to make sure that everything works properly as a single disk system. I rebooted after each O/S installation to make sure it worked properly and also to make sure that the boot loader menu worked properly.

6.3 Final BIOS and LILO configuration:

Next, reconfigure the BIOS so that it again recognizes the first disk (physically reconnect it if you disconnected it earlier) as the boot disk and so that it also recognizes the second disk. How to do this is very system specific and dependent on your BIOS and whether you have a SCSI/IDE or IDE/IDE setup. I haven't tried a SCSI/SCSI setup because SCSI disks and removable frames are significantly more expensive than IDE disks and frames. I wanted performance for my primary O/S but could accept cheap on the other ones.

Make sure and set the second disk type to "Auto" or "Automatic". This will force the BIOS to dynamically determine the disk type at boot time. I have been able to successfully use an ancient 512MB disk, a 4GB, an 8GB, and even a 100MB IDE Zip disk as the second disk. All recognized automatically by the BIOS.

Reboot the system and get back to Linux. At this point, even though there are at least two operating systems installed, this LILO only knows about the original Linux and should boot to it automatically. Watch the boot process and you should see a message about automatically identifying a disk. Once booted, check dmesg to make sure Linux recognized the second disk.

Once this is done, you need to reconfigure LILO on the first disk to make it aware of the second disk. Here are two different lilo.conf files, one for a SCSI/IDE and another for an IDE/IDE system. Each has some strengths and weaknesses...

```
# lilo.conf file for an internal SCSI disk and a removable disk
# configured as a master on the primary IDE connection

disk = /dev/sda         #  These four lines are necessary
   bios = 0x80          #  to get the SCSI disk re-mapped as
disk = /dev/hda         #  the primary drive even though it
   bios = 0x81          #  is selected in the BIOS as the
                        #  boot device.  This might be a BIOS
                        #  specific problem.

# Without them you get the following errors from LILO:
#
#  LILO version 21, Copyright 1992-1998 Werner Almesberger
#
# ading boot sector from /dev/sda
# Warning: /dev/sda is not on the first disk
# And LILO either hangs at LI or repeats endless "01 "'s across the screen

boot=/dev/sda
map=/boot/map
install=/boot/boot.b
prompt
timeout=50
image=/boot/vmlinuz-2.2.12-20smp
   label=Linux
   root=/dev/sda1
   initrd=/boot/initrd-2.2.12-20smp.img
   read-only
other = /dev/hda
# other = /dev/hda is the key element.  Instead of redirecting lilo to
# a specific partition, it redirects it to the MBR on the second disk.
```

```
# that way, LILO doesn't have to know anything about the second disk and
# we can replace it with another because LILO always goes to the same place
# regardless of which specific disk is installed.  LILO was the only boot
# loader I found that would do this.
   label = Disk2
   map-drive = 0x80
      to = 0x81
   map-drive = 0x81
      to = 0x80
# The map-drive lines are necessary to make the second disk think it is
# actually the boot disk.
```

```
# lilo.conf file for a system with two IDE drives.  Both are masters,
# /dev/hda on the primary connector and /dev/hdc on the secondary.
# /dev/hdb is a CDROM slave on the primary IDE connector.
# disk = /dev/hda       # These lines are not necessary for the
#    bios = 0x80        # IDE/IDE installation because the BIOS
# disk = /dev/hdc       # already knows what order they are in
#    bios = 0x81
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
prompt
timeout=50
image=/boot/vmlinuz-2.2.5-15
        label=linux
        root=/dev/hda4
        read-only
other = /dev/hdc
# other = /dev/hdc is again the key.  This just redirects LILO to the
# MBR of the second disk.  Whatever is there gets control.
   label = Disk2
   map-drive = 0x80
      to = 0x81
   map-drive = 0x81
      to = 0x80
```

The /dev, boot, map and image entries are system specific and yours will probably be different that mine. The entries in your initial /etc/lilo.conf file should give you an accurate guide for your system. I copied the "Linux" entry from the original install generated lilo.conf file directly into the new lilo.conf. This should allow you to boot into "Linux" and modify your lilo.conf even if the "Disk2" entry fails totally.

Finally, run lilo −vvv to make sure it agrees with everything you are trying to do.

7.Cost and gotcha's:

7.1 Cost:

I found a Frame and 1 drawer removable drive mount for $20.00. It is the "SNT MOBILE RACK". Disks didn't cost me anything because I had several old or small IDE drives around from upgrades etc. Even if you have to buy them, 2GB IDE drives are cheap.

7.2 Gotcha's

1. Once you decide whether your removable drive is going to be a master or slave, make sure and jumper it properly before securing it in its case. If you forget this step it can take quit a while to trace booting problems back to an improper jumper setting.

2. Make sure that O/S installation routines can ONLY see the drive they are installing to. RedHat refused to let me install LILO to the SCSI MBR if it could see the IDE drive. So, to install to an internal SCSI drive, I had to physically remove the IDE. To install to the IDE, I had to disable SCSI support in the BIOS.

3. If an O/S installation routine tells you it is going to repartition ALL your drives and overwrite EVERYTHING believe it.

4. It is very easy to install an IDE cable backwards.

5. I only tested an IDE master/master set up. I am not sure how a master/slave would work.

6. Installing the removable disk frame does require opening up the computer case. If you are uncomfortable with this get a friend to help.

7. If you set the removable disk to a specific disk type in the BIOS it will work fine until you replace it with another disk of a different type. Then you will get errors or warnings and the system might not boot.

8. Plan everything in advance.

9. With a quick change to the BIOS, the second disk becomes your boot disk. This means that you can have a fully functional O/S available as your emergency/recovery disk.
