Installing Multiple Linux Distributions on the Same Box

There are lots of places that tell you the general ideas on setting up a multiple boot, usually with Windows as being one of the OSs. But what is the philosophy behind it? What about some of the war stories? How many partitions? What boot partitions? While the doc says " you can", what are the ramifications of doing so? How do you set up multiple Linux installations? What are the step by step instructions? Here they are.

In this whitepaper, we're going to install a second distribution on a single machine. First we're going to install Fedora from Red Hat, then install SuSE on the same box. When we're done, we'll have a machine that will, when turned on, will display a boot screen with the option to boot into either.

Boot Loaders and Partitions

In order to configure a machine with multiple distributions, you need to know something about boot loaders and partitions and how Linux starts up.

Hardware configuration

There are two parts of a personal computer's hardware that we're going to be interacting with during this process, so you have to know something about them before we begin. These items are the Master Boot Record (MBR) and hard disk partitions.

All personal computers with a hard disk have a special area in the hard drive's first sector called a Master Boot Record that is used as part of the boot process. All PCs have one and only one MBR - it doesn't matter how many operating systems are installed on the computer.

A personal computer hard disk is divided into one to four areas called partitions. Only one OS can be installed in a partition, but an OS can extend over more than one partition. However, with Linux, multiple versions of a distribution can be contained in the same partition - they're simply stored in different directories. For example, suppose Fedora Core 2.4.22 is installed on a computer. Then, a while later, the kernel is updated to version 2.4.26. The files for this new version don't have to be contained in a separate partition. Rather, a new directory is created during the update process that contains the files specific to 2.4.26. A consequence of this architecture is that the user can choose which version to load - they're both available during the boot process. (Why would you want more than one version of a distro on a computer? In order to use software that ran well under a previous version but wasn't working under the new one.)

Contrast this with the process used to update Windows. After you update an installation of Windows with a service pack or patch, you're basically stuck with that new updated version - depending on the specific situation, the ease of loading anything other than the current version ranges from very difficult to basically impossible. If you find that the updated version doesn't work satisfactorily, either due to its own bugs or to incompatibilities with other software, your only choice often is to reinstall the entire computer from scratch.

With Linux, if you find bugs or incompatibilities, you can load an earlier version simply by selecting it from a menu that displays during startup.

How Linux starts up

Since we're installing multiple operating systems on a machine, how the machine starts up is really important information.

Generic tests

When a personal computer starts up, a few things happen regardless of what type of operating system is being used. First, the power supply checks itself to make sure it can do its own job. Next, the PC looks to a ROM address to find out where the BIOS (basic input-output system) is. The BIOS is loaded, which runs a batch of diagnostic tests. If they all pass, the BIOS determines if this start up was from scratch (power on) or a reboot (the machine was already on.) If the machine was just turned on, the POST (power-on self-test) is executed, which then runs another series of tests, the results of which are echoed to the screen. These include video and memory tests. If the machine was rebooted, the POST is skipped.

Device selection and boot loaders

The next step is for the BIOS to query the CMOS (a bit of ROM powered by a little watch battery on the motherboard) to find out which device the BIOS is supposed to try to boot from - the floppy, the hard drive, or the CD-ROM. The BIOS will run a program called a boot loader from that device that continues the booting process. How the boot loader program is constructed depends on which operating system is installed and which device is being used.

Different operating systems use different boot loader programs; there are a number of boot loaders available in the Linux world, but most Linux distributions provide two from you to choose from during installation - LILO (LInux LOader) and GRUB (GRand Unified Bootloader). LILO is older but suffers from some limitations; GRUB is newer and has additional capabilities. The popular distributions such as Red Hat, SuSE and Mandrake all use GRUB as their default boot loader. This discussion will cover GRUB.

Master Boot Record

If the CMOS indicates the first boot device is a hard disk, the boot loader is divided into at least two parts. The first part is contained in the hard drive's Master Boot Record. (How the boot loader is constructed if the boot device is something other than a hard disk doesn't concern us during this discussion.)

The MBR is a 512 byte chunk of 1's and 0's located in the sector at cylinder 0, head 0, sector 1 of a hard disk. The MBR must perform the same functions regardless of what operating system is installed, but the specific contents of the MBR depends on what operating system(s) are installed on the machine. A DOS MBR is constructed differently than a Windows MBR, which is different from an OS/2 MBR, which is different than a Linux GRUB MBR.

On a machine using Linux and GRUB, the MBR contains both a chunk of code (the first part of the boot loader) that executes when called from the BIOS and a partition table that describes how many partitions are on the hard disk and where they are located.

At least one of the partitions identified in the MBR is identified as being the bootable partition, via a pointer in the MBR. The bootable partition contains the rest of the boot loader code. The code in the MBR directs it to load the bootable partition's boot loader code.

Stage 1

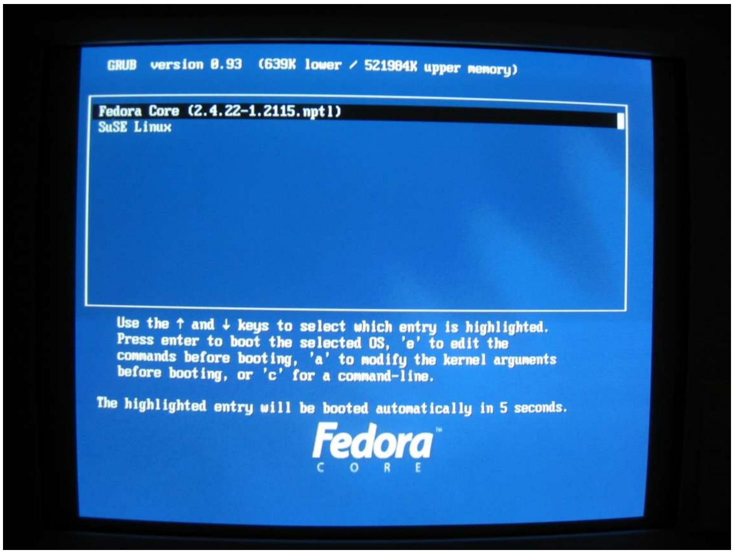
As mentioned, a boot loader program is broken into more than one part. With GRUB, it's broken into three parts, called 'stages'. These stages are named Stage 1, Stage 1.5, and Stage 2.

Stage 1 is the part contained in the MBR, and whose function is to read the partition table in the MBR. Stage 2 is called from Stage 1, although in some special instances, a third stage, Stage 1.5 interjects itself in between 1 and 2. Those instances involve the use of a reserved area by certain operating systems (like FreeBSD) and so aren't applicable in this discussion.

All we care about is that Stage 1 runs from the MBR, and calls Stage 2, which is located in the bootable partition's boot loader code.

Stage 2

Stage 2 reads a file called /boot/grub/menu.lst that provides that data needed to display the boot menu - the screen where multiple kernels of the same or different distributions are shown. Figure 1 shows a boot menu that contains options for both booting either Fedora Core or SuSE.



```
GRUB  version 0.93  (639K lower / 521984K upper memory)

Fedora Core (2.4.22-1.2115.nptl)
SuSE Linux

Use the ↑ and ↓ keys to select which entry is highlighted.
Press enter to boot the selected OS, 'e' to edit the
commands before booting, 'a' to modify the kernel arguments
before booting, or 'c' for a command-line.

The highlighted entry will be booted automatically in 5 seconds.
```

Figure 1. A boot menu with two operating system choices.

The contents of the menu.lst file that generates the boot menu shown in Figure 1 are shown in Listing 1. Note that the line after 'title SuSE Linux' starting with the word 'kernel' wrapped in this listing.

Listing 1. Contents of menu.lst for the boot menu shown in Figure 1.

```
# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to this file
# NOTICE:  You have a /boot partition.  This means that
#          all kernel and initrd paths are relative to /boot/, eg.
#          root (hd0,0)
#          kernel /vmlinuz-version ro root=/dev/hda2
#          initrd /initrd-version.img
#boot=/dev/hda
default=0
timeout=10
```

```
splashimage=(hd0,0)/grub/splash.xpm.gz
title Fedora Core (2.4.22-1.2115.nptl)
    root (hd0,0)
    kernel /vmlinuz-2.4.22-1.2115.nptl ro root=LABEL=/ rhgb
    initrd /initrd-2.4.22-1.2115.nptl.img
title SuSE Linux
    kernel (hd0,3)/vmlinuz root=/dev/hda3 vga=0x314 splash=silent desktop
hdc=ide-scsi hdclun=0 showopts
    initrd (hd0,3)/initrd
```

Figure 2 shows a boot menu that contains multiple options for booting SuSE.

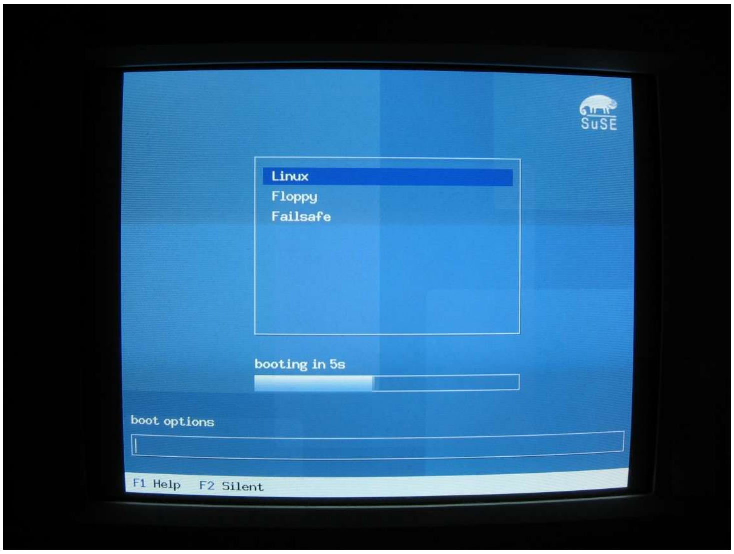


Figure 2. A boot menu with multiple options for booting SuSE.

The menu.lst file for Figure 2 is shown in Listing 2. Note that the lines starting with the word 'kernel' have wrapped in this listing.

Listing 2. Contents of menu.lst for the boot menu shown in Figure 2.

```
# Modified by YaST2. Last modification on Fri Nov 21 15:15:14 2003

color white/blue black/light-gray
default 0
timeout 8
gfxmenu (hd0,0)/message

###Don't change this comment - YaST2 identifier: Original name: linux###
title Linux
    kernel (hd0,0)/vmlinuz root=/dev/hda3 vga=0x314 splash=silent desktop
hdc=ide-scsi hdclun=0 showopts
    initrd (hd0,0)/initrd

###Don't change this comment - YaST2 identifier: Original name: floppy###
title Floppy
```

```
    root (fd0)
    chainloader +1

###Don't change this comment - YaST2 identifier: Original name: failsafe###
title Failsafe
    kernel (hd0,0)/vmlinuz root=/dev/hda3 showopts ide=nodma apm=off acpi=off
vga=normal nosmp noapic maxcpus=0 3
    initrd (hd0,0)/initrd
```

The boot menu is placed on top of a splash screen that is provided with a specific distribution - you can see that the splash screen in Figure 1 is from the Fedora distribution while the splash screen in Figure 2 is from SuSE. Selecting one of the items in the boot menu starts up the operating system selected from the boot menu. From there, the selected operating system takes over the rest of the start up process.

Stage 2 executes once you've made a choice from menu.lst (or, if you don't make a choice, after a timeout period automatically makes one of the choices for you). In the menu.lst file for the multiple SuSE options example, the standard entry refers to kernel /vmlinuz with a robust set of options while the Failsafe entry uses a different (less robust, albeit safer) set of options,

Control is now handed over to /initrd. At this point, you are now running one of the versions available - the one you chose in from the boot menu.

Splash screens

If there are simply multiple options for booting the same operating system, either in various forms or updated versions of the kernel, they can all be placed in the same partition. If, however, there are multiple operating systems available to boot into, each of them must be placed on a different partition.

Summary

In summary, then, the MBR has a partition table that identifies which partition on the hard disk is bootable. Then, the boot loader code in the MBR points to boot loader code in the bootable partition. The boot loader code in the bootable partition displays the choices found in menu.lst to the user. Selection of one of the choices in the boot menu executes the rest of the boot loader code that loads the kernel and begins start up of the operating system.

As a result, the user has the ability to have multiple operating systems on the same computer, and to be able to boot with any of them.

More details about boot loaders and GRUB in particular can be found at Glenn Holmer's excellent writeup at http://www.ameritech.net/users/gholmer/booting.html.

How to install multiple operating systems

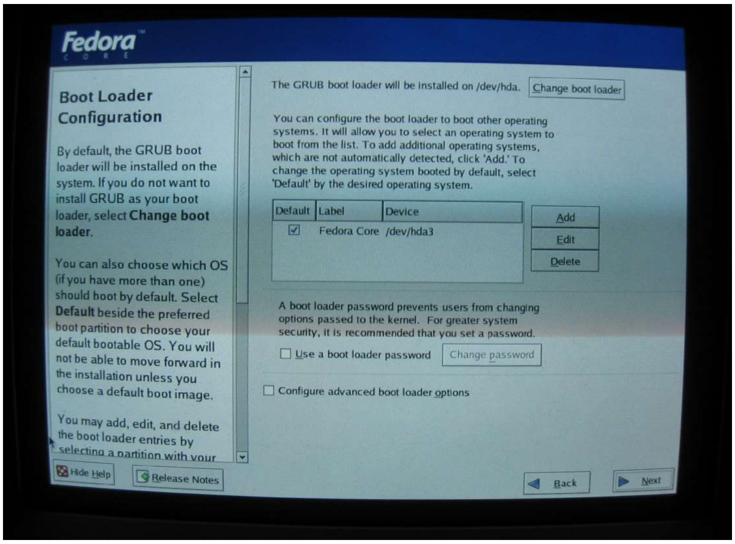
OK, so now you have an idea of how the boot process works, both with one and with multiple OSs. But how do you get there?

Suppose you have a 30 GB hard disk. First, install the first OS, and have it install its own bootloader in the MBR. Leave at least 5 GB of free space on the hard disk. At this point, your 30 GB hard disk might look like this:

```
hda1   primary  /boot    100 MB (Fedora Core)
hda2   primary  /      24000 MB (Fedora Core)
hda3   primary  swap    1024 MB (Fedora Core)
Free space              5000 MB
```

Next, install the second OS in the free space. However, during the installation, there will be a step where you can specify whether or not to install that second OS's bootloader in the MBR. Don't do it.

With Fedora Core, the step is through the "Change boot loader" command button on the Boot Loader Configuration screen, as shown in Figure 3.



Boot Loader Configuration

By default, the GRUB boot loader will be installed on the system. If you do not want to install GRUB as your boot loader, select **Change boot loader**.

You can also choose which OS (if you have more than one) should boot by default. Select **Default** beside the preferred boot partition to choose your default bootable OS. You will not be able to move forward in the installation unless you choose a default boot image.

You may add, edit, and delete the boot loader entries by selecting a partition with your

The GRUB boot loader will be installed on /dev/hda.

You can configure the boot loader to boot other operating systems. It will allow you to select an operating system to boot from the list. To add additional operating systems, which are not automatically detected, click 'Add.' To change the operating system booted by default, select 'Default' by the desired operating system.

Default	Label	Device
✓	Fedora Core	/dev/hda3

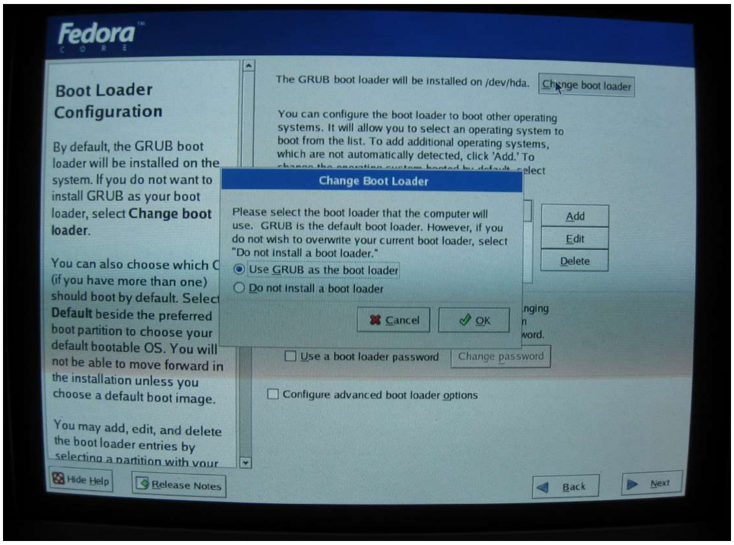
A boot loader password prevents users from changing options passed to the kernel. For greater system security, it is recommended that you set a password.

☐ Use a boot loader password

☐ Configure advanced boot loader options

Figure 3. Control of the Fedora bootloader installation is found in the Boot Loader Configuration screen.

Click on the button to bring forward the Change boot loader dialog, as shown in Figure 4.



Boot Loader Configuration

By default, the GRUB boot loader will be installed on the system. If you do not want to install GRUB as your boot loader, select **Change boot loader**.

You can also choose which C (if you have more than one) should boot by default. Select **Default** beside the preferred boot partition to choose your default bootable OS. You will not be able to move forward in the installation unless you choose a default boot image.

You may add, edit, and delete the boot loader entries by selecting a partition with your

The GRUB boot loader will be installed on /dev/hda. Change boot loader

You can configure the boot loader to boot other operating systems. It will allow you to select an operating system to boot from the list. To add additional operating systems, which are not automatically detected, click 'Add.' To

Change Boot Loader

Please select the boot loader that the computer will use. GRUB is the default boot loader. However, if you do not wish to overwrite your current boot loader, select "Do not install a boot loader."

[✓ Use GRUB as the boot loader]
Do not install a boot loader

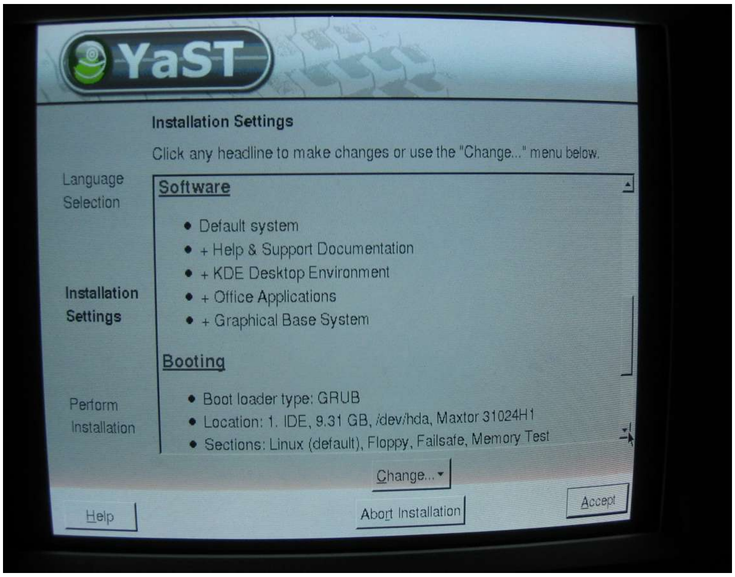
☐ Use a boot loader password

☐ Configure advanced boot loader options

Figure 4. The Change Boot Loader dialog allows you to avoid installing the boot loader in the MBR.

The description in the dialog isn't completely straightforward - there's no mention of the MBR. However, the choices are reasonably identifiable - the second option button, "Do not install a boot loader", pretty obviously tells you that nothing will be changed, which is much closer to what you are looking for.

With SuSE, the step is available through an option in the Installation Settings, as shown in Figure 5.



Installation Settings

Click any headline to make changes or use the "Change..." menu below.

Software

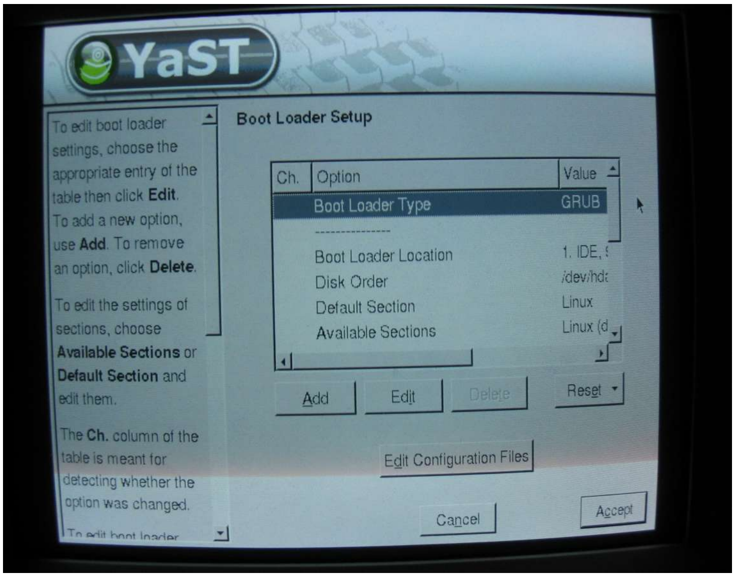
- Default system
- + Help & Support Documentation
- + KDE Desktop Environment
- + Office Applications
- + Graphical Base System

Booting

- Boot loader type: GRUB
- Location: 1. IDE, 9.31 GB, /dev/hda, Maxtor 31024H1
- Sections: Linux (default), Floppy, Failsafe, Memory Test

Figure 5. Control of the SuSE bootloader installation is found in the Installation Settings screen.

The actual location to make the changes isn't as easy to find as it was with Fedora Core - indeed, I missed it during my first few SuSE installs. Click on the Booting hyperlink in the screen or select "Booting" from the menu that displays when you click the "Change..." button. Either action will bring forward the Boot Loader Setup screen, as shown in Figure 6.



To edit boot loader settings, choose the appropriate entry of the table then click **Edit**. To add a new option, use **Add**. To remove an option, click **Delete**.

To edit the settings of sections, choose **Available Sections** or **Default Section** and edit them.

The **Ch.** column of the table is meant for detecting whether the option was changed.

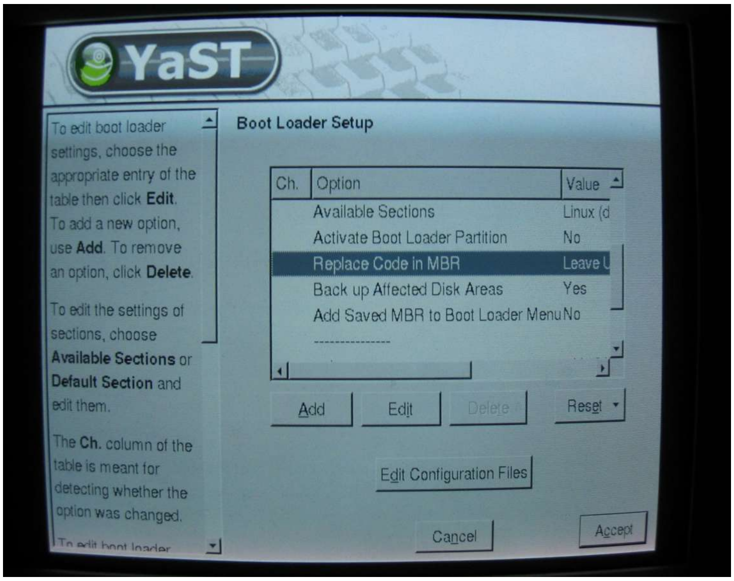
Boot Loader Setup

Ch.	Option	Value
	Boot Loader Type	GRUB

	Boot Loader Location	1. IDE, !
	Disk Order	/dev/hd:
	Default Section	Linux
	Available Sections	Linux (d

Figure 6. The Boot Loader Setup screen allows you to edit various boot loader options.

Highlight the 'Boot Loader Type' line as shown in Figure 6, then click the Edit button to bring forward the editing options shown in Figure 7.



To edit boot loader settings, choose the appropriate entry of the table then click **Edit**. To add a new option, use **Add**. To remove an option, click **Delete**.

To edit the settings of sections, choose **Available Sections** or **Default Section** and edit them.

The **Ch.** column of the table is meant for detecting whether the option was changed.

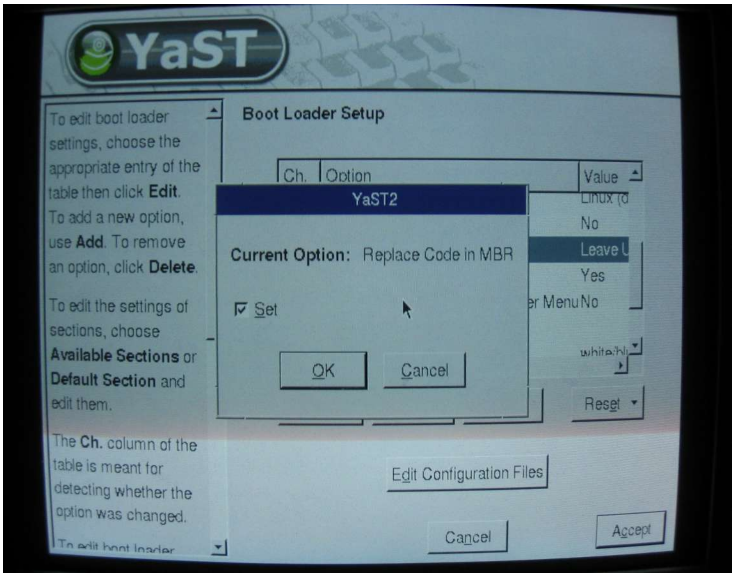
To edit boot loader

Boot Loader Setup

Ch.	Option	Value
	Available Sections	Linux (d
	Activate Boot Loader Partition	No
	Replace Code in MBR	Leave U
	Back up Affected Disk Areas	Yes
	Add Saved MBR to Boot Loader Menu	No

Figure 7. Editing boot loader options in SuSE.

This step is much better explained than with Fedora Core - select the "Replace Code in MBR" line and click the Edit button brings forward the Replace Code dialog as shown in Figure 8.



To edit boot loader settings, choose the appropriate entry of the table then click **Edit**. To add a new option, use **Add**. To remove an option, click **Delete**.

To edit the settings of sections, choose **Available Sections** or **Default Section** and edit them.

The **Ch.** column of the table is meant for detecting whether the option was changed.

To edit boot loader

Boot Loader Setup

Ch.	Option	Value
		Linux (d
		No
		Leave U
		Yes
	er Menu	No
		white/bl

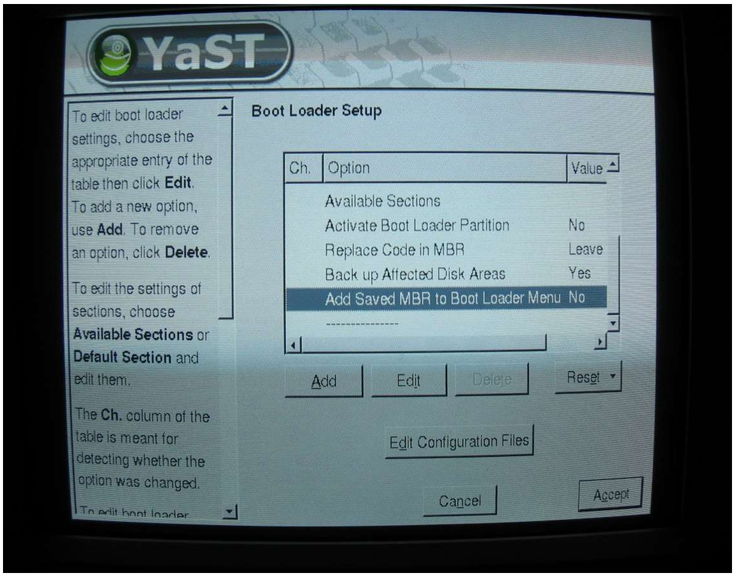
YaST2

Current Option: Replace Code in MBR

☑ Set

Figure 8. The Replace Code in MBR dialog.

Unfortunately, the dialog brought forward isn't that clear. Click the Set check box in order to execute the command to replace the code. The result is shown in Figure 9.



To edit boot loader settings, choose the appropriate entry of the table then click **Edit**. To add a new option, use **Add**. To remove an option, click **Delete**.

To edit the settings of sections, choose **Available Sections** or **Default Section** and edit them.

The **Ch.** column of the table is meant for detecting whether the option was changed.

To edit boot loader

Boot Loader Setup

Ch.	Option	Value
	Available Sections	
	Activate Boot Loader Partition	No
	Replace Code in MBR	Leave
	Back up Affected Disk Areas	Yes
	Add Saved MBR to Boot Loader Menu	No

Figure 9. After modifying the Replace Code in MBR option, the result is shown in the Boot Loader Setup screen.

*\\\ what does the "Add Saved MBR to Boot Loader Menu" option do? ///

Once you're done installing the second OS, you'll see that it has its own /boot partition that contains its own boot loader. In other words, the second (and subsequent) operating systems will only have their boot loader in their own /boot partitions - they'll all depend on the first operating system's boot loader in the MBR to get things started.

Popular distributions of Linux will see that there is already a version of Linux installed, and will use the existing swap partition instead of creating a second one.

Your 30 GB hard disk might look like this now:

```
hda1   primary    /boot    100 MB (Fedora Core)
hda2   primary    /      24000 MB (Fedora Core)
hda3   primary    swap    1024 MB (Fedora Core)
hda4   extended
hda5   logical    /boot    100 MB (SuSE)
hda6   logical    /       4900 MB (SuSE)
```

The last step is to manually modify the menu.lst for the first operating system you installed to include an entry for the new OS. See Listing 1 for an example of what you might add to a standard Fedora Core menu.lst file for SuSE. The exact values will depend on how many partitions you created, in what order, and so on.

Summary

When you boot the machine again, the MBR boot loader code will display the boot menu for the bootable partition on the hard disk - the first OS you installed. The boot menu will display options for all of the operating systems you've installed. Choosing any of those options will run the boot loader in the /boot partition for that operating system, and that's that.

Questions

I hit the wrong button. Now my second OS's boot loader is in the MBR. How do I get my first OS's boot loader back into the MBR?

The short answer is to use GRUB in interactive mode and manually issue commands that will rewrite the MBR the way you want it. Note that this process is writing to your Master Boot Record. A mistake will likely mean that you won't be able to boot your machine.

The long answer is as follows:

1. See the "Installation" topic of the GRUB info file. Type "info grub" in a terminal window. Read it. Twice.

2. Back up your MBR

```
dd if=/dev/hda of=/some/file/name bs=512 count=1
```

3. Make a boot floppy as well.

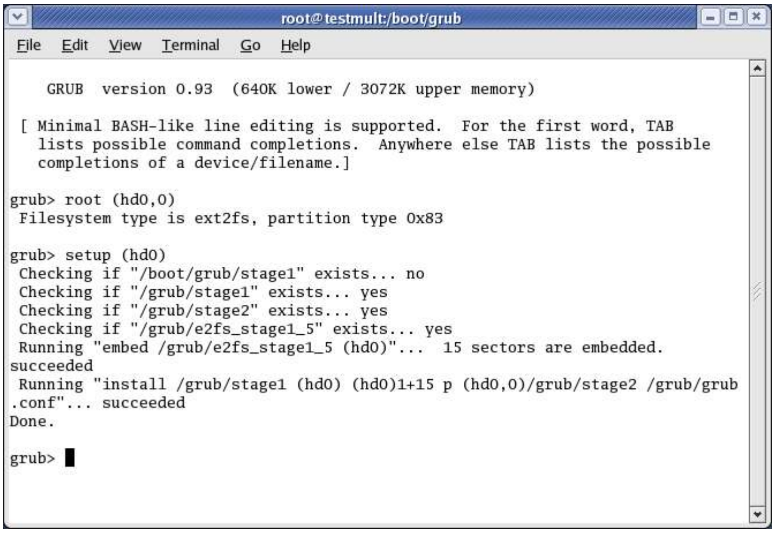
4. Log in as root, open a terminal window, and type "GRUB" in that window. You'll get a prompt like so:

```
grub>
```

5. Enter the commands similar to the following - you'll need to change the values inside the parens according to how your hard disk is set up.

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

A sample display of what you'll see in your terminal window is shown in Figure 10.



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

grub> setup (hd0)
 Checking if "/boot/grub/stage1" exists... no
 Checking if "/grub/stage1" exists... yes
 Checking if "/grub/stage2" exists... yes
 Checking if "/grub/e2fs_stage1_5" exists... yes
 Running "embed /grub/e2fs_stage1_5 (hd0)"...  15 sectors are embedded.
succeeded
 Running "install /grub/stage1 (hd0) (hd0)1+15 p (hd0,0)/grub/stage2 /grub/grub
.conf"... succeeded
Done.

grub> █
```

Figure 10. Writing to the MBR through the interactive use of GRUB.

6. After the quit command, you'll be back at a regular prompt in the terminal window. Next time you reboot (or restart) your machine, the MBR should be back where you wanted it.

So, how do you know what values to use inside the parens? If you have a copy of the menu.lst file for your first OS, you'll find the values there.

How do I change the splash screen?

Each distribution has its own splash screen, as shown in Figures 1 and 2. There is a line in the menu.lst file that fires up the boot menu that controls the splash screen. In Fedora Core, the line is

```
splashimage=(hd0,0)/grub/splash.xpm.gz
```

While in SuSE, the line is

```
gfxmenu (hd0,0)/message
```

You can modify this line to change the splash screen to something different. For instance, I changed the SuSE menu.lst file to include the Fedora Core splash screen like so:

```
# Modified by YaST2. Last modification on Fri Nov 21 15:15:14 2003

color white/blue black/light-gray
default 0
```

```
timeout 8
## gfxmenu (hd0,0)/message
splashimage=(hd0,0)/grub/splash.xpm.gz
```

The # signs in front of the gfxmenu line act to comment out the line.

The line in the boot menu for SuSE just says "Linux". When I add more distributions to my system, it'll be confusing as to which distribution that refers to. How do I change the text of the lines in the boot menu so they can be more descriptive?

You can modify the "title" line to say anything you want to. For example, the follow menu.lst file

```
title Fedora Core (2.4.22-1.2115.nptl)
    root (hd0,0)
    kernel /vmlinuz-2.4.22-1.2115.nptl ro root=LABEL=/ rhgb
    initrd /initrd-2.4.22-1.2115.nptl.img
title SuSE Linux
    kernel (hd0,3)/vmlinuz root=/dev/hda3 vga=0x314 splash=silent
    initrd (hd0,3)/initrd
```

could be changed like so (with apologies to "The Knack" and Frank Zappa):

```
title My Fedora
    root (hd0,0)
    kernel /vmlinuz-2.4.22-1.2115.nptl ro root=LABEL=/ rhgb
    initrd /initrd-2.4.22-1.2115.nptl.img
title SuSE Creamcheese Linux
    kernel (hd0,3)/vmlinuz root=/dev/hda3 vga=0x314 splash=silent
    initrd (hd0,3)/initrd
```

I need more than four partitions but there can only be four partitions on a hard disk. What do I do?

Four partitions, like 640K, seems like a ridiculously artificial limitation, yet at the time it was designed, it seemed like a reasonable decision. Fortunately, there is a mechanism that allows you to define additional partitions within one of the four partitions.

The four partitions are called "primary" partitions, because they are referenced in the partition table in the MBR. A primary partition can be defined as an 'extended' partition, which means that it can hold more partitions. These additional partitions are called 'logical' partitions.

The installation programs of popular distributions all include a partitioning tool that allows you to create more than four partitions. Once you add a fifth partition, the tool will automatically create an extended partition in the fourth partition, and then begin creating logical partitions in the extended partition.

An example of the end result could look like this:

```
hda1 primary
hda2 primary
hda3 primary
hda4 primary (extended)
hda5 logical
```

```
hda6 logical
```

Thanks to many members of the Milwaukee Linux User Group, including Joe Frost, Phil Goembel, Glenn Holmer, George Moulton and Mark P. for feedback on this whitepaper.

For updates to this whitepaper as well as other HOWTO whitepapers, please visit www.hentzenwerke.com.