

Putting a GRUB Boot Loader in a Dedicated Partition Version 1.1

Overview

This document will cover the steps required to build a Linux GRUB Boot Loader in its own dedicated partition. The advantages of using this approach are:

- Stability of the boot menu.
- Retention of any customization of the boot loading menu after new operating systems or upgrades have been installed.
- Immunity to the “kernel” revision mess that occurs in the boot menu.

Intended Audience:

Users of this document should possess at least a moderate beginner's level knowledge of Linux operating systems. At a minimum, the user should be familiar with the use of the terminal mode, use of a text editor and possess the ability to manipulate hard drive partitions comfortably.

Application

This document applies to setups where there are multiple Linux operating systems as boot options. As an example, in my computer I currently have one stable Linux operating system that I use for most of my daily production work. I have at least one or more operating systems installed to test capability and features of upcoming generations of Linux. And finally, I have one version of Windows installed (Windows 2000 Pro – SP4) which I need for unique multimedia applications.

Preparation

You will need to secure a decent partition editor. My strong recommendation is to use GParted Live CD. The url for downloading and burning the iso to a CD can be found at:

<http://gparted.sourceforge.net/index.php>

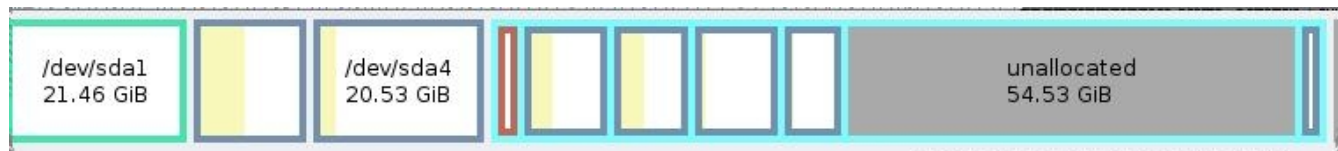
GParted is available in current versions of Ubuntu, and that is useful for quickly checking boot flags, formats, and such. However, using the Ubuntu hard-disk installed GParted makes it very difficult to work on your partitions. The operating system attempts to mount the partition you are working on in GParted, causing the partition to be locked and forcing you to unmount it (in GParted); and there are cases where GParted may issue error messages because of it. So, for easy and safe partition editing, it is recommended that you use the GParted Live CD.

Setting Up The Partition

The first step in this procedure is to set up a new partition for the GRUB boot loader. Where you place this partition is entirely a matter of personal choice. As a point of reference, I have all of my operating systems and related partitions on one hard drive (250 G SATA). My second hard drive (320 G SATA) is used to warehouse data. I also have an external 320 G SATA drive that is used exclusively for a double redundancy backup of both operating systems and data.

The GRUB partition does not have to be large. Generally speaking, a partition of 100 MB. would be more than adequate. I elected to put my GRUB partition at the very end of my hard drive as illustrated in Figure 1.

Figure 1 Operating System Harddrive Partition Map



The partition at the extreme right end of the hard drive map is the GRUB partition.

Set up the new partition as an ext3 file type.

Populating the GRUB Partition

After the partition has been established, the next step is to populate the partition. This is quite easy as we are going to use the existing default GRUB installation for the initial startup files.

Create a mount point for your new GRUB partition. I used “ /media/GRUB “. To create this point, issue these commands:

```
$ sudo mkdir /media/GRUB
$ mount /dev/sda9 /media/GRUB
```

Note : We will adjust the fstab later to mount the GRUB partition at boot.

Putting a GRUB Boot Loader in a Dedicated Partition Version 1.1

We are now ready to build the new GRUB partition. Fire up your file manager of choice (I use Krusader).

1. Navigate to your new GRUB partition.
2. Create a new folder called “/boot”.
3. Copy your active “/grub” folder to the newly created “/boot” folder.

Modifying the menu.lst file

The next step in the process involves modifying the menu.lst . This is probably the most critical aspect required for successfully using the dedicated GRUB partition.

Again, using your text editor of choice, navigate over to your new GRUB partition and open the menu.lst contained in the “/grub” folder.

Find you first Linux entry in the menu.lst . It should look something like:

```
title      Ubuntu 7.10, kernel 2.6.22-14-generic
root      (hd0,6)
kernel    /boot/vmlinuz-2.6.22-14-generic root=UUID=e210cdef-a372-4ede-a819-
cc1dfb74a272 ro quiet splash
initrd    /boot/initrd.img-2.6.22-14-generic
quiet
```

We're going to change this entry so that it looks like:

```
title      Kubuntu Gutsy (V7.10)
root      (hd0,6)
kernel    /vmlinuz root=/dev/sda7 ro quiet splash
initrd    /initrd.img
savedefault
boot
```

Note : The above menu.lst entry will only work properly IF your Linux operating system established a initrd.img type of file in the “/boot” folder of the operating system.

Putting a GRUB Boot Loader in a Dedicated Partition Version 1.1

The big change here is that we are using symbolic links (symlinks) to reference the kernel and initrd. This makes the boot menu entry independent of upgrades to the kernel. Thus, the menu.lst doesn't get cluttered with entries for kernel revisions.

An additional benefit is that the menu item will always display "Kubuntu Gutsy (V7.10)" until I elect to change it.

Note : Pay close attention to the entry made in the "root" entry. In the example "hd0" is the first hard drive. The "6" represents sda7, or the seventh partition. (Note, in GRUB notation, counting of hard drives and partitions starts from zero.)

In the same way, modify the other Linux operating system entries in your menu.lst.

If the "initrd.img" type of file is missing from a specific Linux operating system:

1. Launch your text editor of choice.
2. Open the "/boot/grub/menu.lst" file for the operating system.
3. Find the entry for the Linux operating system.
4. Copy the menu.lst entry for the operating system.
5. Open the "/media/GRUB/boot/grub/menu.lst" file.
6. Paste the entry into the Linux operating system boot options.

The next task is to include entries for memtest and a recovery mode for each Linux operating system. To keep my boot menu clean, I use a divider to group these boot options together. A divider can be inserted as follows:

```
title          Linux Operating Systems Utilities:
root
```

The following example illustrates a recovery boot entry.

```
title          Kubuntu Edgy (V6.10) Recovery Mode
root           (hd0,1)
kernel        /boot/vmlinuz-2.6.17-11-generic root=/dev/sda2 ro single
```

Putting a GRUB Boot Loader in a Dedicated Partition Version 1.1

```
initrd      /boot/initrd.img-2.6.17-11-generic
savedefault
boot
```

Note: This entry is tied directly to a specific kernel. If you later purge the old kernel, this menu.lst entry will no longer be functional.

Activating the New GRUB Partition

From the example above, the GRUB partition is located on partition 9.. Adjust the following commands to your specific configuration. This set of GRUB commands installs the GRUB located in your GRUB partition (hd0,8) to the Master Boot Record (hd0) of your first hard drive (in BIOS boot order).

```
$sudo GRUB
GRUB > root (hd0,8)
GRUB > setup (hd0)
GRUB > quit
$exit
```

You are now ready to test out your new GRUB partition. Re-boot and you should now see your menu.lst . It is recommend that you test each of the menu options to make sure the boot menu is working properly.

Adding the GRUB Partition to the Partitions Mounted at Boot Time

If you want to mount the new GRUB partition as a separate mount point in each operating system at boot time, you will need to make an entry in your /etc/fstab file for each operating system.

The first step is to create a folder in your /media directory.

```
$ sudo mkdir /media/GRUB
$ mount /dev/sda9 /media/GRUB
```

You have a choice of either making the new fstab entry by referencing the UUID number for

Putting a GRUB Boot Loader in a Dedicated Partition Version 1.1

the partition or by using the traditional mount point device naming. If you prefer using the UUID method, you will first need to find out what the UUID is for the GRUB partition. At the terminal, type in the following to get a listing of the UUIDs for all of your devices:

```
$ ls /dev/disk/by-uuid -lh
```

The next step involves modifying your fstab file. It is recommended that a backup copy of the fstab be made before editing the file.

```
$ cd /etc  
$ cp fstab fstab-bu
```

Next launch the text editor in root privileges to open the fstab file.

If you are going to use the UUID method of identifying the device, make a new entry in your fstab file like:

```
# /dev/sda9  
UUID=f7093xy52-0e89-459a-xxb7-64870e73f0f4 /media/GRUB ext3 defaults 0 2
```

Insert the appropriate UUID number after the "UUID=" . Insert the mount point you established in the space where I have "/media/GRUB". The rest of the entry should be the same.

If you are going to do a traditional fstab entry it will look like this:

```
# /dev/sda9  
/dev/sda9 /media/GRUB ext3 defaults 0 2
```

Save the file and re-boot. Your GRUB partition should now be mounted within your Linux operating system.

Maintenance

Eventually you will probably want to either replace existing operating systems with new versions or install completely new operating systems on new partitions. I typically do this at least 3-4 times each year. My strong preference is to build all active operating systems from scratch rather than using the update route.

For **replacing** one operating system with another in an existing partition:

1. Go through the normal installation. At the installation, let GRUB do it's own thing (will build a "/boot" directory within the operating system). When you boot to GRUB after the initial installation, you will be booting to the new (temporary) GRUB menu.

2. To re-gain your previous GRUB Menu, re-set your GRUB partition as the location to use for the menu.lst .

3. At the terminal :

```
$sudo GRUB
```

```
GRUB > root (hd0,8)
```

```
GRUB > setup (hd0)
```

```
GRUB > quit
```

```
$exit
```

When you re-boot, you should be back to your standard GRUB Boot Menu.

For **adding** a new operating system on a new partition:

1. Repeat #1 above.

2. Mount the GRUB partition as previously explained.

3. Edit the menu.lst in the GRUB partition and make an entry for the new operating system.

4. Repeat the instructions for re-gaining the previous GRUB Menu (above).

Acknowledgments

I would like to thank Herman at bigpond for the initial information on using links for GRUB boot points, <http://users.bigpond.net.au/hermanzone/p15.htm>

Additionally I would like to thank mikemqa for his assistance in putting this document together along with his invaluable expertise in editing the final version.

Disclaimer

I have presented this document with the intention of assisting other users in establishing and using a dedicated partition for their GRUB boot loader. It is meant to provide an outline and an example for the user to adapt to his/her own system. All information presented here is correct to the best of my knowledge, and it worked on my system. The reader of this document accepts all responsibility for any malfunctions or damage that may result from following these recommendations. The author accepts no liability for any damages that may result from following this guideline.

Tim Bonesho
aka IndyTim