GRUB tips and tricks

Spicing up a great utility for more IT fun

Jeremy Turner

The GRand Unified Boot loader, or GRUB, has all but replaced the default boot loader on many GNU/Linux distributions. It includes some conveniences over LILO, the LInux LOader. One advantage is not having to remember to run /sbin/lilo every time you make a configuration change. It also can function as a boot loader for removable media such as floppies, CD-R/W and USB flash memory keys. It is short-sighted to view GRUB only as a boot loader to be installed on a hard drive of a GNU/Linux system. Combined with a few other utilities, GRUB can be a powerful and good-looking tool for your home, organization or workplace.

Introduction

First, what exactly is GRUB? GRUB is a boot loader, which means it passes control of the boot process from the Power-On Self Test (POST) to the kernel of your GNU/Linux distribution. GRUB works in a modular, layered fashion so that any unneeded modules are not loaded. Not only does this reduce execution time, but it saves valuable resources when running from removable media. GRUB optionally loads its configuration file at run/boot time, so you don't have to type in commands manually each time. However, the command-line option is still available in case there is an error in your configuration file. So why use GRUB when there are other options out there? The beauty of free software is that you have choices. Alternatives to GRUB include LILO, syslinux and isolinux. The benefit of GRUB is that it will work in many different types of boot devices, but you only need to learn one set of menu commands. In addition, GRUB can work on other forms of bootable storage, such as CD-R/W, USB flash memory keys, floppy disks, and even via a TFTP server with PXE ROM booting.

Figure 1: you can run GRUB on a USB flash memory key!

Installing GRUB on a USB flash memory key

I got the inspiration for this article after trying DSLinux (or DSL), which is a fully graphical Linux distribution weighing in around 50 MB. After seeing an advertisement on their website for a USB flash memory drive with DSL installed, I figured I could probably learn how to set DSL up myself on my Lexar 256 MB JumpDrive. The DSL documentation pointed towards installing via syslinux and reconfiguring the cylinder/head/sector information of my JumpDrive, but I didn't have any luck trying to get my USB flash memory key to boot successfully. Finally, I tried using GRUB and I was up and running with DSL in no time!

GRUB is a boot loader, which means it passes control of the boot process from the Power-On Self Test (POST) to the kernel of your GNU/Linux distribution

First, I recommend creating a directory structure to organize your boot-related files, and keep them separate from any other files you'd like to keep on the USB flash memory key. You could create two partitions, but I couldn't get both partitions to load correctly when I inserted the key back into Windows. On my USB flash memory key, I created a root folder named boot to hold all the data necessary for USB booting (see figure 2). Under the boot folder, I created a directory named grub for GRUB-related files, images which are initial ramdisk (initrd), floppy, or disk images, and finally kernels to hold all the kernels. You may want to organize your boot folder differently, but make sure that you change the corresponding paths and directory names in GRUB's menu.lst file. The menu.lst file that I use can be found in Sidebar 1.

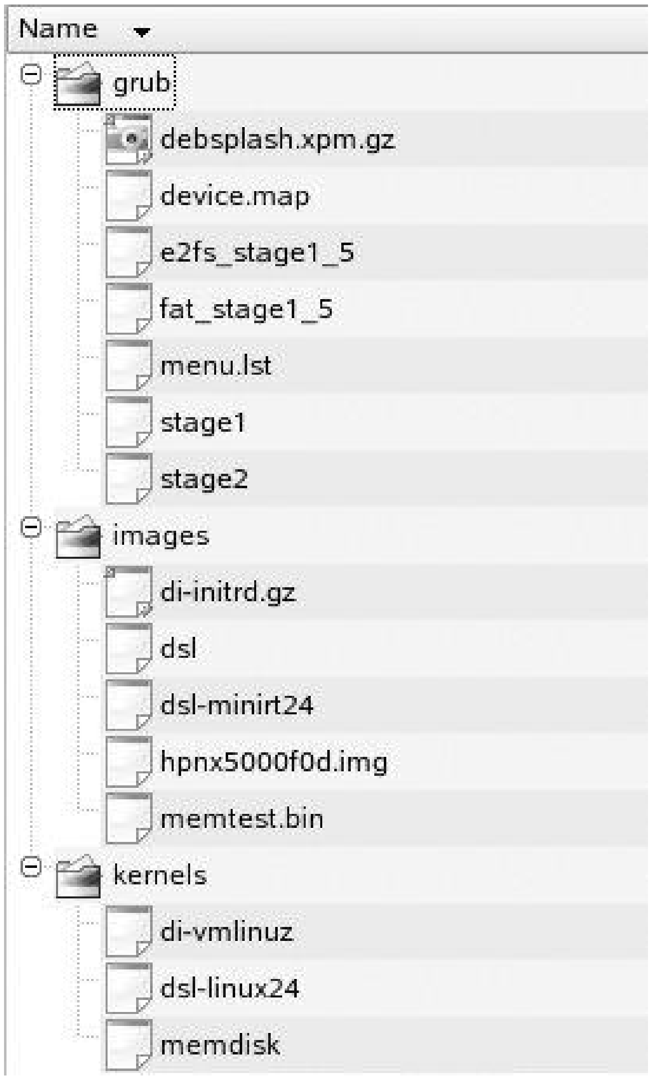
Next, you'll need to copy some of GRUB's stage files, including stage1, stage2, and fat_stage1_5, and put them into the boot/grub directory on the USB flash memory key. These will allow GRUB to boot into GNU/Linux and other operating systems. After the files are copied over, it's time to install GRUB to the Master Boot Record (MBR) of the USB flash memory key.

Luckily, it's the same process as installing to a hard drive:

```
# grub
grub> find /boot/grub/stage1
 (hd0,1)
 (hd2,0)
```

On my system, hd0 is /dev/hda and hd2 happens to be /dev/sda. Just to make sure, we can use a bash-like tab completion to look through a filesystem:

Figure 2: this is the file structure on my USB Memory Key



```
Name
grub
    debsplash.xpm.gz
    device.map
    e2fs_stage1_5
    fat_stage1_5
    menu.lst
    stage1
    stage2
images
    di-initrd.gz
    dsl
    dsl-minirt24
    hpnx5000f0d.img
    memtest.bin
kernels
    di-vmlinuz
    dsl-linux24
    memdisk
```

```
grub> find (hd2,0)/boot/im<TAB>
grub> find (hd2,0)/boot/images/
```

Since the /boot directory on /dev/hda doesn't have an images directory, I know that (hd2) is the hard drive that I want to install GRUB on:

```
grub> root (hd2,0)
 Filesystem is type fat, partition type 0xb
grub> setup (hd2)
 Checking if "/boot/grub/stage1" exists... yes
 Checking if "/boot/grub/stage2" exists... yes
 Checking if "/boot/grub/fat_stage1_5" exists... yes
 Running "embed /boot/grub/fat_stage1_5 (hd2)"...
   15 sectors are embedded.
succeeded
 Running "install /boot/grub/stage1 (hd2)
   (hd2)1+15 p (hd2,0)/boot/grub/stage2
```

```
/boot/grub/menu.lst"... succeeded
Done.
grub> quit
```

Great! Now we have GRUB in the USB flash memory key's MBR. Now, we have to put some files on the memory key to boot into and create a menu.lst file!

When you boot from the USB flash memory key, the key itself becomes hd0, even before the primary master hard drive

GRUB with disk images

One cool trick is to use GRUB and memdisk to boot floppy disk images. Using the memdisk kernel from the syslinux package (`http://syslinux.zytor.com/memdisk.php`), you can load disk images and execute them in a non-emulated environment. How might this be useful? Let's say you have an organization with several different models of desktops and laptops. You could create a CD-R/W or a bootable USB flash memory key with all of the different BIOS upgrades or hardware tests. Rather than carry around a book of floppies, you can simply copy the floppy image and boot from the CD-R/W or USB flash memory key. Using this method, you can also add Memtest86+'s floppy image to your bootable CD-R/W or USB flash memory key and have it at your disposal. Here is an example of a menu.lst snippit using memdisk to boot into Memtest86+:

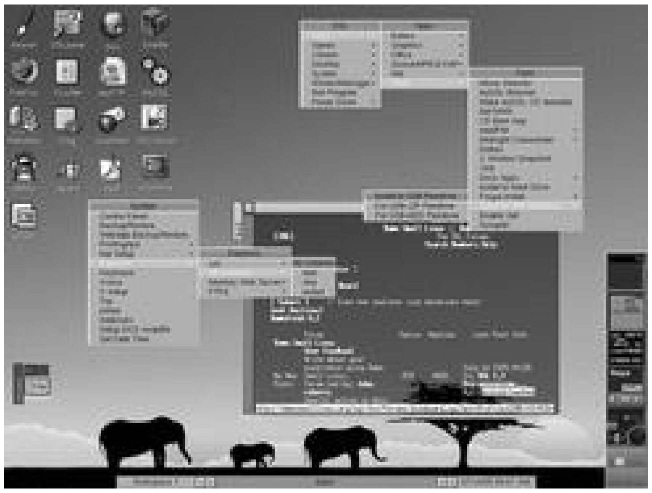
```
title MemTest86+ Ver 1.60
kernel /boot/kernels/memdisk
initrd /boot/images/memtestp.bin
```

There is nothing special about the filenames. The only important thing is that the path and name referenced matches with the actual files. Check out Sidebar 1 for more examples of disk images.

Sidebar 1: Contents of menu.lst

default=0
timeout=10
root=(hd0,0)
splashimage=/boot/grub/debsplash.xpm.gz
title DSL 1.2 (2.4.26) 1024x768 (save to RAM)
kernel /boot/kernels/dsl-linux24 ramdisk_size=100000 init=/etc/init lang=us apm=power-off vga=791 toram nomce noapic quiet knoppix_dir=images knoppix_name=dsl
initrd=/boot/images/dsl-minirt24.gz
title Debian Sarge Installer
kernel /boot/kernels/di-vmlinuz initrd=/boot/images/di-initrd.gz ramdisk_size=10240 root=/dev/rd/0 devfs=mount,dall rw
initrd /boot/images/di-initrd.gz
title HP nx5000 F0.d BIOS Upgrade
kernel /boot/kernels/memdisk
initrd /boot/images/hpnx5000f0d.img
title Memtest86+ (1.60)
kernel /boot/kernels/memdisk
initrd /boot/images/memtestp.bin

Figure 3: DSLinux is a 50 MB fully-graphical live GNU/Linux distribution



GRUB with DSLinux

So how can you boot a full GNU/Linux desktop off a USB flash memory key with GRUB? First, download the DSL ISO9660 image, and either burn it to a CD, or mount it via loopback:

```
# mkdir dsl-test
# mount -t iso9660 -o loop dsl-image.iso dsl-test
```

Next, copy the KNOPPIX file, kernel, and initial ramdisk:

```
# cp dsl-test/KNOPPIX/KNOPPIX \
  /media/usb/boot/images/dsl
# cp dsl-test/boot/isolinux/linux24 \
  /media/usb/boot/kernels/dsl-linux24
# cp dsl-test/boot/isolinux/minirt24.gz \
  /media/usb/boot/images/dsl-minirt24.gz
# sync
# umount dsl-test && rmdir dsl-test
```

This assumes that your USB flash memory key is mounted at /media/usb. Next, edit the /media/usb/boot/grub/menu.lst file and make sure it looks like the entry in Sidebar 1. You might have noticed that the root line at the top of the menu.lst file says (hd0,0) even though we used (hd2,0) earlier. When you boot from the USB flash memory key, the key itself becomes hd0, even before the primary master hard drive. Once you have the menu.lst file edited, go ahead and reboot. Make sure that your BIOS is set to USB-HDD, USB-ZIP, or USB-FLOPPY. You might need to experiment to see which one works. Once you get to the menu, select the option for DSL. If you get a GRUB error and are unable to successfully boot into a kernel, press the 'c' key to open a GRUB prompt. You can try commands like find to help locate files to boot from.

GRUB splash images

Another cool function with GRUB is putting a splash image on the boot menu screen. By default, GRUB will make the menu screen a plain black-and-white menu. There are menu options to change the black and white colors, but why stop there? Grab your favorite picture, or head to one of the URLs listed below which have splash images created for you. If you are creating your own, it will need to be in XPM format, a maximum color palate of 14 colors, and 640x480 resolution size. The GIMP can help transform your graphic to these specifications. As an alternative, you can use the ImageMagick suite of programs. The application convert can help with this conversion process. You can run it as follows:

```
$ convert -resize 640x480 -colors 14 \
    mycoolpicture.jpg mybootsplash.xpm
$ gzip mybootsplash.xpm
```

In this case, the file mycoolpicture.jpg will be resized to 640x480, reduced to 14 colors, and saved in the XPM graphical format. The second step compresses the XPM file using the gzip compression method. GRUB can display the gzipped-xpm splash images well.

If you don't want to create your own splash image, check out some of the following web sites which have them available for download:

- GNU GRUB Public Splashimage Archive (`http://ruslug.rutgers.edu/~mcgrof/grub-images/images`)
- GRUB Splash (`http://vision.featia.net/linux/grubsplash`)
- Once you have the XPM or gzipped-xpm file, you need to add a line at the top of GRUB's menu.lst to instruct GRUB to load the specified boot splash image. The line should read as follows:

```
splashimage=/boot/grub/debsplash.xpm.gz
```

In my case, I'm using the Debian boot splash image as my background.

Another cool function with GRUB is putting a splash image on the boot menu screen

GRUB on CD-R/W media

I've also mentioned several times about installing GRUB to use for a bootable CD-R/W disk. When generating the ISO image, you will use some special settings of mkisofs. The command to build a bootable GRUB CD-R/W looks like:

```
$ mkdir -p iso/boot/grub
$ cp stage2_eltorito iso/boot/grub
$ mkisofs -R -b boot/grub/stage2_eltorito \
    -no-emul-boot -boot-load-size 4 \
    -boot-info-table -o grub.iso iso
```

The key is to copy the stage2_eltorito file into the /boot/grub directory of the CD image tree, and run mkisofs with the options specified above. As mentioned earlier, you can also burn a menu.lst file along with kernels and disk images and put them all on the CD. In the menu.lst, you will need to use (cd) as the device, rather than (hd0). Splash images work well, too.

Conclusion

Congratulations! Now you have a bootable USB flash memory key, with DSL and memtest86+, and even a nice boot splash image. You even have the knowledge to add extra images, like your favorite BIOS update disk image, and others. For more information, check out some of the links in the resources section below.

Resources

- Grub Homepage (`http://www.gnu.org/software/grub`)
- Grub Manual (`http://www.gnu.org/software/grub/manual/grub.html`)
- Memdisk (`http://syslinux.zytor.com/memdisk.php`)
- Syslinux (`http://syslinux.zytor.com`)
- Memtest86+ (`http://www.memtest.org/#downiso`)
- GNU GRUB Public Splashimage Archive (`http://ruslug.rutgers.edu/~mcgrof/grub-images/images`)
- GRUB Splash (`http://vision.featia.net/linux/grubsplash`)

Copyright information

About the author

Jeremy Turner enjoys freelance writing when given the opportunity. He often plays system administrator, hardware technician, programmer, web designer, and all-around nice guy. You contact him by visiting his web site (`http://linuxwebguy.com`).