

Multiple Boots from USB disk with Linux

Since I could not find any simple explanations on the Internet to install multiple Linux distribution packages on a USB disk and make it a bootable disk, I decided to share with you what I learnt. I hope you will find this information useful.

Content

Basic Installation of Linux on Internal Disk	1
Using Virtualization – Linux and Sun VirtualBox version 2	1
Multiple Boots from USB disk with Linux	2
Introduction	2
Before you start – getting ready	2
Prepare your USB disk – Creating Linux partitions	3
Setting your BIOS to boot from USB Disk	3
Installing Linux on your USB boot Disk	3
Tune up your USB boot menu – customized Grub menu	4
Customize your main disk boot menu – accessing your USB partitions	7
MS Windows Vista Special Note	9
Conclusion	9

Basic Installation of Linux on Internal Disk

Installing Linux on a computer hard disk, as the unique operative system, is very easy and does not require any particular knowledge. All most popular distributions will do the installation simply and fast. In fact, it generally takes 10 to 15 minutes to install Linux on a computer loaded with all sort of free applications. Download the Live CD from your favorite distribution site, restart your computer with the CD and 15 minutes later, your computer is ready.

Installing Linux on dual boot with MS Windows XP (or any other OS) on an internal disk is also quite simple. Dual boot means that you will have the option to select your operative system when you boot your computer. In a dual boot installation, Linux will first create a partition on your hard disk and then install a “boot loader” on your hard disk. This boot loader will display a menu when you boot your computer, allowing you to choose your operative system.

Installing various distribution packages of Linux on your internal disk is as easy as installing a Dual Boot. You end up with various partitions on your hard disk, each holding its own Linux package.

I first tested Linux in early 2007. I was happy to see how well done were most of the distribution packages. The only common issue to most of the package was the installation of WIFI drivers. Still today, this issue requires special attention. After many hours spent on Community Forums and testing all sort of solutions, I discovered that the easiest and simplest solution for most of the distributions, especially the KDE versions – is the installation of “ndiswrapper”. This small application uses your MS Windows driver for your WIFI to allow you to connect to the net. Once you’ve got that, Linux is a piece of cake! I tried various packages and ended up with Ubuntu and PCLinuxOS 2007 on my internal hard disk.

Using Virtualization – Linux and Sun VirtualBox version 2

A few months ago, I discovered that SUN was releasing a new update of its VirtualBox software, a free application which allows the installation of Linux within MS Windows. In this case, you can run Linux without having to reboot your computer. VirtualBox runs as any Windows application and allows you to install your favorite Linux packages directly from the Live CD image. No need to burn a CD.

Although Linux runs a bit slower when using “virtualization”, this solution offers you the ability to quickly move from MS Windows to Linux as you would between Email and Web browsing. I use VirtualBox to quickly check how my web site looks under Linux when updating it, and to test the web application I am

developing with Adobe Flash. I installed a free Apache Web Server on MS Windows XP which I use as a mirror of my web site. Using VirtualBox, I can run Ubuntu or Kubuntu and access my local intranet server using the IP address of my computer. I can then test my web application with Linux while making changes under Windows before loading my latest update to my web site online.

When using VirtualBox, you don't have to worry about special support for WIFI. I am not exactly sure how it works, but it does! Under virtualization, Linux uses your Ethernet connection (not your Wireless) and connects to the net wirelessly via your Windows WIFI connection. It works brilliantly and makes installation a piece of cake.

Multiple Boots from USB disk with Linux

Introduction

My latest project was a bit trickier. I wanted to be able to boot from a USB disk and have the option to choose from various distribution packages of Linux. Finally, I also wanted to boot from my internal disk and still have the ability to run any Linux package installed on my USB disk. After 2 days of tests and tries, I got it done! In fact, when you know how to do it, it is quite simple.

My current configuration looks like this on my HP dv5000 laptop: When I start my computer, a custom screen is displayed "Welcome to Fabrice's Computer" and offers me the option to start with MS Windows XP, Ubuntu, PCLinuxOS or to choose another Linux package from my external USB disk. If my USB disk is connected, another menu is displayed when I select the latest option. If my USB disk is not connected, then a message saying that the disk is not available is displayed.

If I choose MS Windows, my computer loads MS Windows XP. I can then choose to open VirtualBox and start Ubuntu or Kubuntu. Since I have a 22" LCD monitor connected to my laptop, I can display my Linux OS on my laptop screen and Windows on my LCD monitor. It's like having 2 computers in one.

I can also boot my computer with my USB disk connected and set the BIOS of my computer to boot from the USB disk. When I do so, a menu is displayed allowing me to choose which Linux package I want to load. I also have the option to load Windows and the 2 Linux packages installed on my internal hard disk. I ended up installing on my external USB disk the following packages:

- PCLinuxOS Min 2008
- Kubuntu 8.04.1 (I later reinstalled version 8.10)
- openSUSE-KDE v.11
- openSUSE-Gnome v11
- Mandriva Spring 2008.

I wanted to install Fedora 2009 but gave up after a while. This version was refusing to get my sound working (not uncommon according to their forum). I also could not start the Mandriva 2009 Live CD. According to their forum, I am not the only one. And since I did not want to spend two days downloading their Live Free DvD, I used their Spring 2008 Live CD instead.

Let me now share with you how I got to have a multiple boots external USB disk with all these Linux packages.

Before you start – getting ready

In order to have a multiple boots external USB disk, your computer must allow you to start from a USB disk. Most computers from the past 4 or 5 years will give you this option. We'll see in a second how to do that.

Firstly, download your favorite Linux Live CD distribution packages and store them all on your favorite backup hard disk. I used 2 CD-RW to prepare my live CDs. After installing my packages, I erase the CDs and use them to prepare the next set of Live CDs.

Secondly, make a backup of your internal hard disk. If you have a dual boot installation of your hard disk, **make sure you have a backup of your MBR**. This is important since you might accidentally erase your main “boot loader” while setting up your external USB Disk. Acronis True Home Image is a great software for making backups of your partitions. This application also allows you to restore your MBR in just a few seconds.

Prepare your USB disk – Creating Linux partitions

Before you start with your installation, you want to prepare your external USB disk. Follow my advice: **partition** and **format** your external USB hard disk **before** installing Linux. You can use Gparted (free) or Acronis Disk Director.

I decided to use an external self-powered 80 GB USB disk. I bought it a while ago. It’s small and light. It does not require external power, so I can travel with it easily.

I created the following partitions:

- a) Primary partition, NTFS – 20 GB (used to store my files and exchange them between Windows and all my Linux packages).
- b) Logical partition – 2 GB – Linux Swap.
- c) Logical partitions, Ext3 – 10 GB each. I created 5 partitions
- d) Logical partition, Ext3 – 20 MB. I labeled it “Grub” and use it to store my customized “Boot Loader menu image” and a menu text file that I use to load my Linux packages (see later).

Make sure you **format** each partition and **label** them. I simply used a label that shows which Linux package I have installed. I ended with 5 partitions labeled: “PCLinuxOS”, “Kubuntu”, “openSUSE_KDE”, “openSUSE_Gnome”, and “Mandriva”.

Setting your BIOS to boot from USB Disk

Your computer must allow you to start from a USB disk. To get it done, you must start your computer and hit the key that allows you to access your BIOS settings. You should see on the bottom left corner of your screen, at boot time, which key is required for your computer. It could be F1, Del, or F10 or any other key. Pay attention, you just have 2 or 3 seconds to get it done!

When you open your computer BIOS, find the section where you can set the Disk Boot order of your computer and set it so that you will have listed: 1. CD drive; 2. USB disk; 3. Internal Disk. Save your configuration and restart your computer.

Installing Linux on your USB boot Disk

Set your BIOS to start from your USB Disk and reboot with a Live CD in your CD drive. Your computer will start loading the Linux CD. When ready, choose to install Linux and when asked where to install it, choose the **Custom Partition** option.

Most of the Linux packages will display your hard disks defining your internal hard disk as “hda” and your USB disk as “sda”. If you had a second USB disk connected, it would appear as “sdb”. A third would be “sdc”. Now, be aware... Some packages don’t follow this convention. Some instead refer to your internal hard disk as “sda” and your USB as “sdb”. So be sure to clearly identify your USB disk and make a note of how it is displayed. **For the rest of this tutorial, I will refer to the internal hard disk as “hda” and the USB disk as “sda”.**

You should see your “sda” disk with all its partitions. Select your 2 GB Linux Swap partition to act as your SWAP file and select the proper Ext3 partition to mount as “/”. This is where your Linux distribution will be installed. **Make sure that none of the other partitions are set to be formatted!**

So far, as you can see, there is nothing really special about installing Linux on your USB disk. You follow the exact same process as for installing it on your internal disk, except that here, you choose to install it on a partition on the “sda” disk.

What’s different is the installation of your “boot loader”. Please take your time here and pay close attention to where Linux will install the “boot loader”. **The default is your internal disk. We do not want this!**

Depending on your distribution package, the “boot loader” will be defined before or after the installation of the system files. So be aware. You can choose to install the “boot loader” on your internal disk (hda), on a partition of your internal hard disk (hda1,hda2,hda5,hda6, etc...), on your USB disk (sda) or on a partition of your USB disk (sda1,sda5,sda6...).

In my case, my USB disk shows me the following options:

- sda - root of the disk
- sda1 – NTFS partition
- sda5 – Linux Swap
- sda6 – PCLinuxOs – Ext3 partition
- sda7 – Kubuntu – Ext3 partition
- sda8 – openSUSE_KDE – Ext3 partition
- sda9 – Mandriva – Ext3 partition
- sda10 - openSUSE_Gnome – Ext3 partition
- sda11 – Grub – Ext3 small partition

The bottom line, you must install your “boot loader” on “sda”.

Repeat the same step with each distribution that you wish to install. Recreate a “boot loader” on “sda” after each installation. We will customize our boot loader when we have installed all the distribution packages.

Tune up your USB boot menu – customized Grub menu

Now that you’ve installed your favorite distribution packages, let’s take a few minutes to tune up your boot menu. I decided to use Mandriva or PCLinuxOS to manage my boot loader. I found their tool (which is the same) very easy to use.

Since you are starting your computer from your USB disk, the Grub menu will identify your USB disk as “hd0” and your internal disk as “hd1”. In my case, the partitions of my USB disks are defined as:

- (hd0,5) – PCLinuxOs
- (hd0,6) – Kubuntu
- (hd0,7) – openSUSE_KDE
- (hd0,8) – Mandriva
- (hd0,9) - openSUSE_Gnome
- (hd0,10) – Grub – Not needed but nice to have!

Please note that Linux see my PCLinuxOs partition as “/dev/sda6” and Grub sees it as (hd0,5).

Firstly, I created 2 folders on my Grub small partition: “boot” (see later) and “grubimage”. I copied the gfxmenu file from PCLinuxOS called “message” into the “grubimage” folder of my Grub partition. This partition is identify as “/dev/sb11” by Linux and defined in the Grub menu as (hd0,11).

To customize your Grub image, follow these steps:

Grub uses a special file as the background image of your boot menu (on KDE packages). The location of the file in use is specified at the top of your grub menu. See sample below from PCLinuxOS:

```
timeout 15
color black/cyan yellow/cyan
gfxmenu (hd0,10) /usr/share/gfxboot/themes/pclinuxos/boot/message
default 0
```

1. Find the gfxmenu file used by your distribution package and copy it into a “menu” folder in your “Home” folder. Let’s assume that the file is named “message” – as it is the case with PCLinuxOS. (Mandriva calls it “gfxmenu”). **Rename the file “messageOri”** (original)

2. Create a folder “new” inside the “menu” folder.

3. Open a terminal as a super user and go to the “new” folder. Type the following:

```
su
Enter your password.
cd /home/YOURNAME/menu/new
```

4. Extract the content of your gfxmenu file. It will be extracted into the “new” folder:

```
cat /home/YOURNAME/menu/messageOri | cpio -i
```

5. Find your favorite background image (800 x 600) and copy it into your “menu” folder. **Rename it “back.jpg”**.

6. Edit your new “back.jpg” image file with Gimp adding a title if your wish and save the file as a jpg image.

7. Now, copy the “back.jpg” image into the “new” folder and replace the existing file with this same name.

8. Finally, rebuild your gfxmenu file.

```
find . | cpio -o > /home/YOURNAME/menu/message
```

9. You now have your own personal gfxmenu file. Great! Place this “message” file into to folder as defined in your grub menu. As said previously, I store mine on my Grub partition inside the folder “grubimage”

Customize your Grub menu – list all your Linux distributions

Before we rebuild the MBR (redefine the “boot loader”), it’s a good idea to review your grub menu. The simplest way to do it is to open the terminal and then open your menu. If you have a KDE package installed (such as PCLinuxOS or Mandriva), open the terminal and type:

```
su
Enter your password).
Kwrite /boot/grub/menu.lst
```

This will open the KWrite editor and display your current menu. See below the menu I have for my USB disk:

```
timeout 15
color black/cyan yellow/cyan
gfxmenu (hd0,10)/grubimage/message
default 0

title 1. MS Windows XP Media Center
```

```

root (hd0,0)
map (hd0) (hd1)
map (hd1) (hd0)
chainloader +1

title 2. Mandriva 2.6.24.7
kernel (hd0,8)/boot/vmlinuz-2.6.24.7-desktop586-1mnb
BOOT_IMAGE=2._Mandriva_2.6.24.7 root=UUID=aa801cca-5ec1-5f6c-468f-
85d0cb6fd401 splash=silent vga=788
initrd (hd0,8)/boot/initrd-2.6.24.7-desktop586-1mnb.img

title 3. Kubuntu 8.04.1 - 2.6.24-21
kernel (hd0,7)/boot/vmlinuz-2.6.24-21-generic
BOOT_IMAGE=3._Kubuntu_8.04.1_-_2.6.24-21 root=UUID=9d038e24-30fa-4d8c-
aa64-6d74937cc532 ro quiet splash=silent
initrd (hd0,7)/boot/initrd.img-2.6.24-21-generic

title 4. PCLinuxOS 2008 - 2.6.22.15
kernel (hd0,5)/boot/vmlinuz-2.6.22.15.tex2
BOOT_IMAGE=4._PCLinuxOS_2008_-_2.6.22.15 root=/dev/sda6 acpi=on
splash=silent vga=788
initrd (hd0,5)/boot/initrd-2.6.22.15.tex2.img

title 5. openSUSE 11.0 KDE
kernel (hd0,6)/boot/vmlinuz-2.6.25.18-0.2-default
BOOT_IMAGE=5._openSUSE_11.0_KDE root=/dev/sda7 showopts splash=silent
vga=0x317
initrd (hd0,6)/boot/initrd-2.6.25.18-0.2-default

title 6. openSUSE 11.0 Gnome
kernel (hd0,9)/boot/vmlinuz-2.6.25.18-0.2-default
BOOT_IMAGE=6._openSUSE_11.0_Gnome root=/dev/sda10 showopts splash=silent
vga=0x317
initrd (hd0,9)/boot/initrd-2.6.25.18-0.2-default

title 7. PCLinuxOS 2007 - Main Disk
kernel (hd1,2)/boot/vmlinuz-2.6.18.8.tex5 BOOT_IMAGE=7._PCLinuxOS_2007_-_
_Main_Disk root=/dev/hda3 acpi=on resume=/dev/hda4 splash=silent vga=788
initrd (hd1,2)/boot/initrd.img

title 8. Ubuntu 8.04.1 - Main Disk
kernel (hd1,1)/boot/vmlinuz-2.6.24-19-generic
BOOT_IMAGE=8._Ubuntu_8.04.1_-_Main_Disk root=UUID=d9ba69bf-a2b1-4812-
affa-4db8f9f8b010 ro quiet splash=silent
initrd (hd1,1)/boot/initrd.img-2.6.24-19-generic

```

Notice that some package such as Ubuntu and Kubuntu like to refer to the root using the UUID of the partition. DO NOT CHANGE THIS.

Make sure that each menu section (title section) refers to the latest version of the installed **kernel** and **initrd**. You might need to update these after installation of the current updates. The key here is to check inside the "/boot/grub/" folder of your Linux distribution the most recent version of each. I find it simpler to select the file and choose to rename it (but I do not change its name); then copy its current name and paste it into the menu. This assures me from making any typos!

Now that you have a nice grub menu, you need to reinstall your "boot loader". With PCLinux or Mandriva (or most of the KDE packages), go to the System Configuration Tool and find the Boot or Boot Loader section. Take time to familiarize yourself with this tool and find the place where you can define the location of your "boot loader". **Choose to install is on "sda"**.

I also used the same tool to install additional “boot loader” on each one of the partition of my USB disk holding a Linux package: sda6, sda7, sda8, sda9, sda10. So I ended up with a main boot loader on “sda” (the root of the USB disk) and a copy on each Linux partition. The copy is not needed but I like it this way.

I finally copied the “menu.lst” file that I carefully edited into the “/boot/grub/” folder of each Linux partition. I even saved a backup of the menu on my Grub partition, just in case!

You are now done!! Reboot your computer from your USB disk and you should see you nice new grub menu.

Customize your main disk boot menu – accessing your USB partitions

Now that we have a bootable USB disk with its own menu, we want to also get access to these partitions when we start our computer from the main disk.

Since I have a Linux partition on my internal disk, I can already start Linux without using my USB disk. I only need to add a section to my internal disk Grub menu to access my USB partitions. Here is my main (internal disk) grub menu:

```
timeout 10
color black/cyan yellow/cyan
gfxmenu (hd0,2)/usr/share/gfxboot/themes/pclinuxos/boot/message
default 0

title 1. MS Windows XP Media Center
root (hd0,0)
makeactive
chainloader +1

title 2. Ubuntu 8.04.1 - Kernel 2.6.24-21
kernel (hd0,1)/boot/vmlinuz-2.6.24-21-generic root=UUID=d9ba69bf-a2b1-4812-affa-4db8f9f8b010 ro quiet splashh
initrd (hd0,1)/boot/initrd.img-2.6.24-21-generic
quiet

title 3. PCLinuxOS 2007 - Kernel 2.6.18-8
kernel (hd0,2)/boot/vmlinuz-2.6.18.8.tex5 BOOT_IMAGE=3._PCLinuxOS_ -
_Kernel_2.6.18-8 root=/dev/hda3 acpi=on resume=/dev/hda4 splash=silent
vga=788
initrd (hd0,2)/boot/initrd.img
quiet
savedefault
boot

title 4. Other Linux Systems on USB Disk
rootnoverify (hd1,10)
configfile (hd1,10)/boot/menu.lst
```

Notice that the ‘title 4’ section opens a menu that I stored on my USB Grub partition (hd1,10). Remember, since I start from my internal disk, the internal disk is defined as “hd0” and my USB disk is defined as “hd1”.

Since the current boot menu on my USB disk refers to the USB disk as “hd0”, I cannot use the same menu that I created for my bootable USB disk. So I created another menu referring to the USB disk as “hd1”. I stored that menu on my Grub special partition.

Warning: For the “title 4” section to work, I must also have on my (hd1,10) partition (the Grub partition) a “device.map” file that defines the disks to Grub. Grub looks for the following file: “/boot/grub /device.map”.

So, on my Grub partition ([hd1,10](#)), I have the following:

- Folder “boot”
- Inside the folder “boot”, I have a “menu.lst” file (see below)
- Inside the folder “boot”, I have a folder “grub”
- Inside “/boot/grub”, I have the file “device.map”.

Here is the content of the file “device.map”:

```
(hd0) /dev/hda
(hd1) /dev/sda
```

Here is the content of the file “menu.lst”. This special file gives me access to the USB Linux partition starting my computer from my internal hard disk:

```
timeout 15
color black/cyan yellow/cyan
gfxmenu (hd1,10)/grubimage/message2
default 0

title 1. Mandriva 2.6.24.7
kernel (hd1,8)/boot/vmlinuz-2.6.24.7-desktop586-1mnb
BOOT_IMAGE=2. Mandriva 2.6.24.7 root=UUID=aa801cca-5ec1-5f6c-468f-
85d0cb6fd401 splash=silent vga=788
initrd (hd1,8)/boot/initrd-2.6.24.7-desktop586-1mnb.img

title 2. Kubuntu 8.04.1 - 2.6.24-21
kernel (hd1,7)/boot/vmlinuz-2.6.24-21-generic root=UUID=9d038e24-30fa-
4d8c-aa64-6d74937cc532 ro quiet splash=silent
initrd (hd1,7)/boot/initrd.img-2.6.24-21-generic

title 3. PCLinuxOS 2008 - 2.6.22.15
kernel (hd1,5)/boot/vmlinuz-2.6.22.15.tex2 root=/dev/sda6 acpi=on
splash=silent vga=788
initrd (hd1,5)/boot/initrd-2.6.22.15.tex2.img

title 4. openSUSE 11.0 KDE
kernel (hd1,6)/boot/vmlinuz-2.6.25.18-0.2-default root=/dev/sda7
splash=silent showopts vga=0x317
initrd (hd1,6)/boot/initrd-2.6.25.18-0.2-default

title 5. openSUSE 11.0 Gnome
root (hd1,9)
kernel (hd1,9)/boot/vmlinuz-2.6.25.18-0.2-default root=/dev/sda10
splash=silent showopts vga=0x317
initrd (hd1,9)/boot/initrd-2.6.25.18-0.2-default

title 6. Return to previous menu
rootnoverify (hd0,2)
configfile (hd0,2)/boot/grub/menu.lst
```

See that we now refer to the USB disk as “hd1”. Also notice the last section and how we get back to the previous menu.

MS Windows Vista Special Note

If you have problems with the bootloader running MS Windows Vista, you should check **NeoSmart EasyBCD 1.7.2**. NeoSmart is a non-profit organization dedicated to all fields of technology. Here is an extract from NeoSmart's web site at <http://neosmart.net> :

"With EasyBCD, setting up and configuring Windows boot entries is simple, and there is no easier way to quickly boot right into Linux, Mac OS X, or BSD straight from the Windows Vista bootloader - on the fly, no expert knowledge needed!"

"EasyBCD is geared for users of all kinds. Whether you just want to add an entry to your old XP partition or want to create a duplicate for testing purposes; if you're interested in debugging the Windows Kernel or septuple-booting your seven test operating systems, EasyBCD is the key."

Conclusion

I truly hope that this information will help you with your Linux multi boot installation on USB disk. I am not an expert in Linux at all, although I do have over 22 years of experience working with computers. Linux is a bit different and there are a few things that one has to learn. At the end of the day, it's the same with any other operative system.

I don't play computer games; I prefer to learn how to use new things! That's my way to entertain myself!

Best wishes to you.

Fabrice Menoyot
<http://freeEducationCenter.com>