Building a casper-rw USB for Kubuntu/Ubuntu

Section 1 - Some File Types
Everything in Linux is a File or a Process.

ISO Image Files
A *.iso image is an **archive file** also called an *image file* when it contains the contents of a CD or DVD.

The most important features of an ISO image file is that it is a compressed block by block copy of the original file(s) on a hard drive that can be easily burned to a DVD or CD.

Device Files (or Special Files)
/dev/*file_name* is an interface for a device driver; it appears in the file system as if it were an ordinary file in the directory /dev.
There are three types of Device Node files.

Character devices (also called Serial Devices)
Character special files or character devices are device nodes used to drive physical devices (such as mice& keyboards) one byte (character) at a time..

Block Devices
Block special files or block devices correspond to devices through which the system moves data in the form of random addressable data blocks (e.g. 512 byte disk blocks). These device nodes are often used to drive devices such as hard disks, CD-ROM drives, or memory-regions.

Pseudo-Devices
Device nodes on are not always associated with physical devices. Nodes that lack an association with a physical device are called pseudo-devices. They provide various functions handled by the operating system. Some of the most commonly-used (character-based) pseudo-devices include:

/dev/null	Accepts and discards all input; produces no output.
/dev/zero	Produces a continuous stream of NUL (zero value) bytes.
/dev/random	Produces a variable-length stream of random numbers.

Loop devices (http://en.wikipedia.org/wiki/Loop_device)
A loop device, is a pseudo-device that makes a file accessible as a block device.
Sometimes, when using 'bash speak' the loop device is incorrectly referred to as a 'loopback' device. In 'bash speak' loopback is reserved for networking devices. However to add to the confusion in 'grub speak' loopback is the command to make a virtual device from a file. Why is it so? I do not know. I'm not smart enough to invent this stuff, I'm just dumb enough to use it.

Section 2 - Grub2 Boot Process

The first Block, LBA 0 (or Sector CHS 0/0/1) of a formatted hard drive contains:

- the Master Boot Record (MBR)
- and the Primary or Master Partition Table (MPT)

The Grub2 Two stage Boot Loader
Grub2 is a two stage boot loader. The first stage is primitive, it does not understand disk formatting

(partitions and file systems) and can only address the hard drive using LBA (Logical Block addressing) The first part of stage 1 is incorporated into the MBR, the remainder is normally *embeded* in unformatted disk space following the MBR; however it can be embeded in any suitable unformatted disk space or an an unformatted BIOS partition. When formatting a disk or USB drive always leave at least 1MiB of free space following the MBR for grub stage 1.

Grub stage 2 can understand formatting, has its' own command/scripting language similar to Bash and is installed on either its own boot partition or the boot partition for the drive.

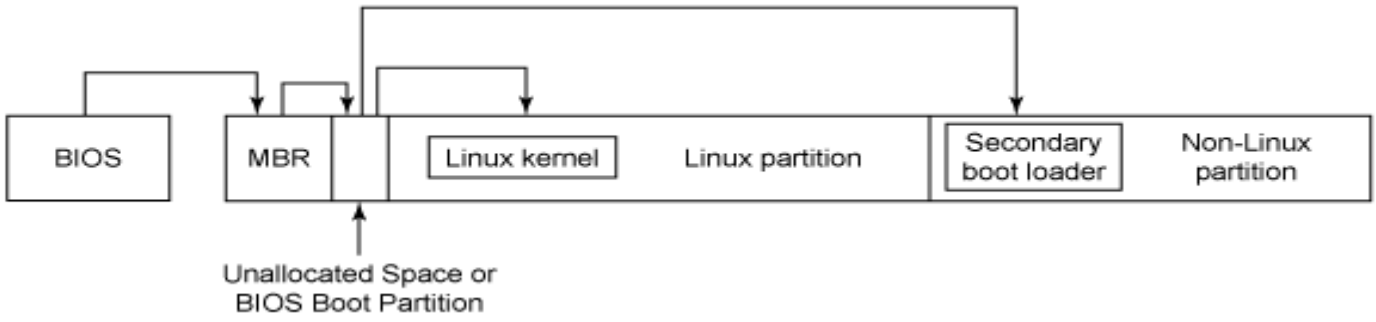
Grub2 Installation.
The **`grub-install`** command creates a directory **`/boot/grub`** on the nominated boot partition and installs into **`/boot/grub`** the grub stage 2 files . It then installs a Grub MBR on LBA 0 (CHS 0,0,1) and '*embeds'* the remainder of the first stage boot loader, in the following unformatted blocks.

Booting a Grub2 Linux System.
On Boot-up, BIOS executes and initiates the Grub2 MBR loader which loads and runs the grub stage 1; this in turn loads and runs grub stage 2, which uses the data in the **`/boot/grub/grub.cnf`** file data to complete the boot process.
`grub.cnf` contains information on the available boot operating systems and their booting parameters. Two of those parameters are

1. the location of the Kernel (vmlinuz)
2. the location of the RAM-drive (initrd)

The Kernel is a program that constitutes the central core of an Operating System and when loaded into memory and started it remains in memory for the whole time that the computer is in operation. **`vmlinuz`** may be a compressed version of the Linux kernel or a link to a compressed or uncompressed Kernel.

`initrd` (initial RAM drive) and **`initramfs`** (initial RAM file-system) refer to two different methods used by '*live*' devices to load a temporary root file system into memory that facilitates booting the kernel. **`initrd`** is the method used by the Grub2, it is loaded into 'early user space', and contains executables and that do hardware detection, device discovery and the module decompression/loading (vmlinuz + .mod) necessary to get the Linux root file system mounted. Once this is done initrd is unmounted and its memory freed.



Section 3 - Making a Bootable Casper EXT2 USB

Casper
Casper is an Ubuntu hook for **`initrd`**, (set by the **`grub.cfg`** kernel parameter boot prompt

`boot=casper`)
A 'hook' is a technique for altering or augmenting the behaviour of a software by intercepting, inspecting and redirecting certain messages that are passed between two software components.

Casper is one of several different methods of initiating a process capable of booting a '*live*' OS from an *.iso cd/dvd or USB Flash drive. You can examine a casper based *.iso using 'Ark' or a similar program to decompress the *.iso, it will show a directory named 'casper' that contains the necessary files to boot the '*live*' device.

A Persistent USB
A persistent '*live*' Linux USB install is one that is capable of saving data changes to the USB device instead of leaving the data in system RAM and consequently loosing the data when the '*live*'device is removed. Some changes are not saved persistently, e.g. graphic and network card settings. Using persistence can be very convenient, but it is less secure.

Partitioning the USB
Note: The partition sizes given in the following description are for guidance based on an 8GB USB and can be modified according to the users needs.

Insert the USB and use Gparted, or similar to partition the 8GiB USB Flash drive into an three primary partitions one fat32 (label **`FAT32`**) and two ext2 partitions Casper requires that you label these partitionslabels **`casper-rw`** & **`home-rw`**. ext2 is a non journaling file-system and results in less flash-memory wear than the journaling systems ext3/4.

** Do not forget to leave at least 1MiB of unpartitioned space for Grub stage1 immediatly following the MBR

Partition#1 is about 1.5 GB partition labelled **FAT32**. This partition will be visible to MS Windows and provides a convenient means of transferring files to/from Windows. If this partition is not required then omit this step.

Partition#2 is ext2 and about 4GB labelled **`casper-rw`**, it will contain the 'guts' of the operating system

Partition#3 is ext2 and will use the remainder of the USB and is labeled **`home-rw`**. This partition will hold the users data files

Close Gparted, remove the USB stick, reinsert the usb stick.

Some Useful Disk Commands
What devices are available?
```
ls /dev | grep ^sd
```

How do I identify these devices.
```
fdisk -l
```
&/or
```
blkid
```
&/or
```
df -hT |grep ^[F/] | sort
```

What devices are mounted?
`mount -l |grep ^/`

Mount the casper-rw partition as /media/casper-rw

Access a terminal and use the command **`sudo blkid`** or **`df -h`** to identify your device, example:- part of the response will contain information similar to /dev/sdb2: LABEL="casper-rw"
The remainder of this description assumes your device is **`/dev/sdb2`** and is mounted at **`/media/casper-rw`**

Install grub2
Use **Linux Partions only** e.g. ext2, not Microsoft eg fat32.
The following command installs grub2 and creates a new /boot and /boot/grub/ directory on your USB
`sudo grub-install --no-floppy --root-directory=/media/casper-rw /dev/sdb`
response **`password for user:`** input your password
response **`Installation finished. No error reported.`**

The following command creates the directory into which the Kubuntu/Ubuntu iso's will be placed.
`sudo mkdir /media/casper-rw/boot/isos`

Change (the User Access Control) **mode** for **`casper-rw`** - it makes your experimentation a lot easier and it is essential if you are going to use persistence.
`sudo chmod 777 -vR /media/casper-rw`

At the completion of the above operations the directory **`/media/casper-rw/`** will contain two sub-directories
`/boot/grub`. which is now populated with a whole pile of grub stuff.
`/boot/isos`. is currently empty. Use the file manager to copy the required Kubuntu/Ubuntu .iso's into this directory.
Check the properties of the imported *.iso files, if the owner is root, change the UAC mode again.
`sudo chmod 777 -vR /media/casper-rw`

Create a text file called **`grub.cfg`** in **`/media/casper-rw/boot/grub`**
This is the grub configuration file, and is read and used by grub when it starts.
Create atext file labeled **`grub.cfg`** znd use the following text as guide.
Only ONE of the OS can be made persistent

Please Note the character ¶ is an indicator only, for the end of a line (Enter).
This is to avoid confusion if text editors or word processors auto-wrap lines.
Do Not include ¶ in your final **`grub.cfg`** text file.
```
######################
set timeout=100 ¶
set default=0¶
¶
menuentry "LinuxMint 15 persistent" { ¶
loopback loop (hd0,2)/boot/isos/linuxmint-15-cinnamon-dvd-32bit.iso ¶
set root=(loop) ¶
linux /casper/vmlinuz boot=casper iso-scan/filename=/boot/isos/linuxmint-15-cinnamon-dvd-32bit.iso
persistent splash -- ¶
initrd /casper/initrd.lz ¶
```

} ¶
¶
menuentry "Kubuntu 12.10" { ¶
loopback loop (hd0,2)/boot/isos/kubuntu.iso ¶
set root=(loop) ¶
linux /casper/vmlinuz boot=casper iso-scan/filename=/boot/isos/kub.iso splash noeject noprompt -- ¶
initrd /casper/initrd.lz ¶
no persistence ¶
} ¶
¶
##################

In 'grub speak' (hd0,2) is equivalent to 'bash speak' sda2
The information on the line following the command '**linux**' are called *kernel parameter boot prompts*, *boot time parameters*, *boot prompts* or *cheat codes* they are passed to the kernel at boot time to modify the kernel installation.
Some common boot prompts can be seen using the terminal command **man bootparam**

You are now ready to boot Kubuntu/Ubuntu from your casper-rw.

Section 4 - Odds 'n Sods

Some Grub Commands.
A Linux **loop device** is a pseudo-device (e.g. /dev/loop0) that makes a file accessible as a block device.
Before use, a loop device must be connected to an existing file in the filesystem.
If the file contains file system *.iso, it may then be mounted as if it were a disk device.
Loopback example:-**loopback loop (hd0,1)/boot/isos/kub.iso**
Grub command: **loopback** [-d] *device file*
Make the device named *device* correspond to the contents of the filesystem image in *file*. For example:

```
loopback loop0 /path/to/image
ls (loop0)/
```

With the -d option, delete a device previously created using this command.

Set example:- **set root=(loop)**
Grub command: **set** [*envvar=value*]

> Set the environment variable *envvar* to *value*. If invoked with no arguments, print all environment variables with their values.

Linux example:- **linux /casper/vmlinuz**
Grub command: **linux** *file ...*
Load a Linux kernel image from *file*. The rest of the line is passed verbatim as the kernel boot time parameters.. Any initrd must be reloaded after using this command (see initrd).

> On x86 systems, the kernel will be booted using the 32-bit boot protocol. Note that this means that the 'vga=' boot option will not work; if you want to set a special video mode, you will need to use GRUB commands such as 'set gfxpayload=1024x768' or 'set gfxpayload=keep' (to keep the same mode as used in GRUB) instead. GRUB can automatically detect some uses of 'vga=' and translate them to appropriate settings of 'gfxpayload'. The linux16 command (see linux16) avoids this restriction.

Kernel Parameters example:- **`boot=casper iso-scan/filename=/boot/isos/kub.iso splash noeject noprompt --`**
The entries on the line following the command '**`linux`**' are called,
boot parameters, boot time parameters, *boot prompts* or *cheat codes.* They are passed to the kernel at boot time to modify the kernel installation. refer '**`man bootparam`**'

Initrd example:- **`initrd /casper/initrd.lz`**
Grub command: **`initrd`** *file*

> Load an initial ramdisk for a Linux kernel image, and set the appropriate parameters in the Linux setup area in memory. This may only be used after the `linux` command (see linux) has been run.

Install the Grub emulator grub-emu
Experiment with grub commands in a normal Linux terminal
`sudo apt-get install grub-emu` when installed read man grub-emu
Some Hints for first use of grub or grub-emu
To boot into the grub on power-up if you do not normally get a grub menu press 'Alt' first then power on and keep 'Alt' pressed untill the grub boot screen appears.

`grub-emu`	# starts grub emu
`set pager=1`	# scroll one page at a time (space-bar) or one line at a time (enter).
`help`	# view all the available commands. Space bar pages one screen, enter pages one line.
`command-name -hu`	#view command usage and function

The up/down arrow keys will allow you to scroll through your command history.

If your computer is like me, a little on the ancient side, it may not have a BIOS option to 'Boot from USB' if that is the case get a copy of Plop boot CD. Boot into plop on a CD and that will give you the option to boot from USB.

Plop usb boot cd
http://www.plop.at/en/bootmanager.html

Any questions? I'm at Sturt Street on Friday 'arvos.