20a - **LILO** Linux Loader

Types of Boot Loaders: LILO,GRUB,SILO,Loadlin,ELILO,QUIK,VMELILO,ZIPL

MBR Description

First 512 Bytes of the Hard disk.
Independent standard format for PCs. 512 Bytes

- First 446 Bytes: Boot loader program code
- 64 Bytes: Partition Table containing max 4 partitions: Addr + Type
- Last 2 Bytes: Magic MBR ID number '`0xAA55`'

Note: On SuSE GRUB *`fstype`*`_stage1_5` file resides right after the MBR

Boot sectors:

First 512 bytes of each partition except on an extended partition.
Used to hold information on OS for DOS, Windows or OS2 partitions or
a Boot Loader(Optional) for Linux partitions(with magic number '`0xAA55`' at the end).

20b - LILO - The Linux Loader

- **LILO restrictions**
  - Problems with kernel/initrd files located after cylinder 1024 for older BIOS (~1996 and older) and LILO Before Ver.21.6

- **Advantages over GRUB:**
  - **fits inside**
  - When software RAID is used
  - When VLM is running

- **LILO configuration**

  - With Yast
  - With editor
  - Monitor character size (`vga= { ask | 791 | normal | extended }`)
  - Root(`/`) and `/boot` directories

Example of `/etc/lilo.conf`:

```
boot = /dev/hda           # LILO installation target: MBR
backup = /boot/MBR.hda.050428
                          # backup file for the old MBR
                          # 2005-04-28
vga = normal              # normal text mode (80x25 chars)
read-only
menu-scheme = Wg:kw:Wg:Wg
lba32                     # Use BIOS to ignore
                          # 1024 cylinder limit
prompt
password = q99iwr4        # LILO password (example)
timeout = 80              # Wait at prompt for 8 s before
                          # default is booted
message = /boot/message   # LILO's greeting
```

```
#----------
image=/boot/vmlinuz
root=/dev/hda7          # Root partition to mount to '/'
label=linux
initrd=/boot/initrd
#------------------
other=/dev/hda1
table=/dev/hda       (Partition table)
label=win
```

- Install `lilo` in MBR by running the command: `lilo`
  Results expected:
  ```
  Added linux *
  Added win
  ```

- **LILO configuration and control**.
  Table 2.4 Keywords and their corresponding options to use in the configuration file for LILO.

Command line Options	Config file(`/etc/lilo.conf`) keywords
`-b bootdev`	`boot=`*`bootdev`*
`-c`	`compact`
`-d deciseconds`	`delay=dsec`
`-D label`	`default=`*`label`*
`-i filename`	`install=`*`bootsector`*
`-f filename`	`disktab=`*`file`*
`-l`	`linear`
`-m filename`	`map=`*`mapfile`*
`-P fix`	`fix-table`
`-P ignore`	`ignore-table`
`-s filename`	`backup=`*`file`*
`-S filename`	`force-backup=`*`file`*
`-v`	`verbose=`*`level`*

- **Selection of a default menu entry only once for the next booting.**

  `lilo -R` *`labelname`*

- **syslinux** - Boot loader for Linux from windows

  SYSLINUX is a boot loader for the Linux operating system which operates off an MS-DOS or Windows FAT file system. It is intended to simplify first-time installation of Linux and for creation of rescue and other special-purpose boot disks. `Romfs` files systems are mainly used for the initial RAM disks used during installation.

- **To Recover the DOS/Windows MBR:**
  Boot in DOS/Win95/Win98 and enter the command:
  ```
  fdisk /mbr
  ```
  or boot the Win2000/XP in manual repair mode and issue the command:
  ```
  fixmbr  or fixmbr \Device\HarDisk0
  ```
  or boot the Linux which last changed the MBR and issue the command:
  ```
  lilo -u /dev/hda
  ```

GRUB

(**GR**and **U**nified **B**ootloader)

Advantages over LILO:

- Allows to change the content of the menu (`/boot/grub/menu.lst`) or the kernel, or `initrd` without needing to rewrite the MBR or the Boot sector.

- GRUB reads directly the filesystems therefore can locate all the files needed for the boot sequence without refering to a Hard drive Geometry via the BIOS.

- GRUB can read directly the following filesystems:
  - BSD FFS
  - VFAT16 and VFAT32
  - ReiserFS
  - JFS
  - Minix
  - Ext2fs and Ext3fs
  - XFS

How does GRUB work:

- GRUB's start part (`stage1`) is normally written in the first 512 bytes (like `LILO`)
  - in the Hard Disk MBR
  - or in floppy boot sector (`stage1` and `stage2`)
  - or in a Hard disk Partition's boot sector.

- When the PC starts after the BIOS init procedure:
  - It loads GRUB by loading from MBR ,the `stage1` (which includes the physical address of *`fssys`*`_stage1_5`),
    Note: SuSE loads the *`fssys`*`_stage1_5` after the MBR on the Boot track
  - Then loads *`fssys`*`_stage1_5.`
  - Then loads `stage2` (directly from `/boot/grub/` directory).
  - Then loads the `/boot/grub/menu.lst` and presents the menu

- The *`fssys`*`_stage1_5` can be either written after the MBR on the hard disk or will be loaded by the `stage1` already in MBR.
  `fssys` filesystem type where the `stage2` file is. (eg. `e2fs_stage1_5`)
  `fssys``_stage1_5` decodes the filesystem for loading directly the second part of GRUB: file `/boot/grub/stage2`.

- GRUB understands the Hard disks as follows:

  `(hd0)` MBR of first Hard disk found
  (normally `/dev/hda` or `/dev/sda` if only SCSI disks exist)

  `(hd0,0)` Boot sector of first partitionon first Hard disk found
  (normally `/dev/hda1` or `/dev/sda1` if only SCSI disks exist)

  `(hd1)` Second Hard disk found.
  This one can be any hard disk found after the first one.
  It can be `/dev/hdc` or `/dev/sdd` etc.

  Note: The sequence of finding hard disks is:
  IDE ---> SCSI ---> others

Configuration of GRUB

GRUB uses 3 different configuration files:

- `/boot/grub/device.map`
- `/etc/grub.conf`
- `/boot/grub/menu.lst`

`/boot/grub/device.map`

The file `/boot/grub/device.map` is a hard disk mapping list for converting GRUB hard disk notation to Linux notation. If you encounter problems when booting, check if the sequence in this file corresponds to the sequence in the BIOS.
After manually changing `device.map`, execute the following command to reinstall GRUB. This command causes the file `device.map` to be reloaded and the commands listed in `grub.conf` to be executed:

```
grub --batch < /etc/grub.conf
```

`/etc/grub.conf`

This file contains the parameters and options the command **grub** needs for installing the boot loader correctly: eg.

```
root (hd0,4)
install /boot/grub/stage1 d (hd0) /boot/grub/stage2 0x8000 (hd0,4)/boot/grub/menu.lst
quit
```

Meaning of the individual entries:

`root (hd0,4)`
: This command tells GRUB to apply the following commands to the first logical partition of the first hard disk (the location of the boot files).

`install` *`parameters`*
: The command **grub** will be run with the command `install`.
Install Parameters:

`/grub/stage1 d (hd0)`
: `stage1` of the boot loader should be installed in the MBR.

`/grub/stage2 0x8000`
: `stage2` should be loaded to the memory address 0x8000

`(hd0,4)/grub/menu.lst`
: Tells GRUB where to look for the menu

`/boot/grub/menu.lst`

The configuration file of GRUB is located in `/boot/grub/menu.lst`
This file contains the equivalent of `/etc/lilo.conf` but in another format.
On SuSE 8.1 and up, GRUB comes with boot images, which are normally installed in the directory: `/usr/lib/grub/`
This is where the boot images of GRUB are. If you destroy or corrupt one the main GRUB images in /boot/grub then recopy them from here.

example:

```
gfxmenu (hd1,1)/boot/message
color white/blue black/light-gray
default 0
fallback 1
```

```
timeout 8

title linux
   kernel (hd1,1)/boot/vmlinuz root=/dev/hdc2 apic vga=791 \
     initrd (hd1,1)/boot/initrd

title linux-scsi
   kernel (hd1,1)/boot/vmlinuz root=/dev/hdc2 apic vga=791 \
     hdb=ide-scsi max_scsi_^luns=1
   initrd (hd1,1)/boot/initrd

# For booting Windows NT or Windows95
title windows
   root (hd0,0)
   makeactive
   chainloader +1

# For loading DOS if Windows NT is installed
# chainload /bootsect.dos

title floppy
   root (fd0)
   chainloader +1
```

List of possible configuration parameters:

These commands are usable in the command-line and in menu entries.
If you forget a command, you can run the command `help' (*note help::).

* `blocklist`	Get the block list notation of a file
* `boot`	Start up your operating system
* `cat`	Show the contents of a file
* `chainloader`	Chain-load another boot loader
* `cmp`	Compare two files
* `configfile`	Load a configuration file
* `debug`	Toggle the debug flag
* `displayapm`	Display APM information
* `displaymem`	Display memory configuration
* `embed`	Embed Stage 1.5
* `find`	Find a file
* `fstest`	Test a filesystem
* `geometry`	Manipulate the geometry of a drive
* `halt`	Shut down your computer
* `help`	Show help messages
* `impsprobe`	Probe SMP
* `initrd`	Load an `initrd`
* `install`	Install GRUB
* `ioprobe`	Probe I/O ports used for a drive
* `kernel`	Load a kernel
* `lock`	Lock a menu entry
* `makeactive`	Make a partition active
* `map`	Map a drive to another eg. `map (hd0,1) (hd0,0)`

`map (hd0,0) (hd0,1)`
Used to swap the assignment of physical partitions.
Useful to let windows 98 boot from a partition which is not the first one (picky fellow!!) and make it think that it is.

* `md5crypt`	Encrypt a password in MD5 format
* `module`	Load a module
* `modulenounzip`	Load a module without decompression
* `pause`	Wait for a key press
* `quit`	Exit from the grub shell
* `reboot`	Reboot your computer
* `read`	Read data from memory
* `root`	Set GRUB's root device
* `rootnoverify`	Set GRUB's root device without mounting
* `savedefault`	Save current entry as the default entry
* `setup`	Set up GRUB's installation automatically
* `testload`	Load a file for testing a filesystem
* `testvbe`	Test VESA BIOS EXTENSION
* `uppermem`	Set the upper memory size
* `vbeprobe`	Probe VESA BIOS EXTENSION

Setting-up GRUB for booting

Write GRUB on diskette:

This method allows to boot Linux from a diskette.

```
cd /boot/grub
dd if=stage1 of=/dev/fd0 bs=512 count=1
dd if=stage2 of=/dev/fd0 bs=512 seek=1
```

Write GRUB on MBR:

To install GRUB in MBR:

- Verify that the DataDir is `/boot/grub` (SuSE)

```
vi $(which grub-install)
     datadir=/boot/grub
<esc> :wq
grub-install /dev/hda
```

Or manual alternative:

```
grub
grub> find /boot/grub/stage1
 (hd0,3)
grub> root (hd0,3)
 Filesystem type is ext2fs, partition type 0x83
grub> setup (hd0)
 Checking if "/boot/grub/stage1" exists...yes
 Checking if "/boot/grub/stage2" exists...yes
 Checking if "/boot/grub/e2fs_stage1_5" exists...yes
 Running ........
..........
Done
```

In case of trouble to find the files eg. The result is:

```
Checking if "/boot/grub/stage1" exists...no
```

```
Checking if "/boot/grub/stage2" exists...no
```

then get out (CTRL-C) and try the following command:

```
grub-install /dev/hda
```

Securing GRUB:

To limit the possibility to write kernel options and commands to GRUB a password can be entered in the `/boot/grub/menu.lst` as follows:
Note: Wherever a password is written here, the md5crypt format can also be used.
eg. *`password --md5 $1$lS2dv/$JOYcdxIn7CJk9xShzzJVw/`*

Protecting only against entering kernel options:

Globally with only one password:

`password password``(clear text)` or

Protecting against booting.

Global password and individual locking:

In global section:

`password` *`password`*

Note: no " or ' should be used around the password.
These characters are seen as part of the password.

In individual section:

`lock`

immediately at the line after the title of each section that requires a password to boot.

eg.
```
password password
title linux-scsi
     lock
     kernel (hd1,1)/boot/vmlinuz.....
```

Individual password per section:

In individual section:

`password` *`password`*

eg.
```
title linux-scsi
     password mart28ty
     kernel (hd1,1)/boot/vmlinuz.....
```

Note: The user needs to press 'p' before entering the password.

Extra features:

Exchange the keyboard keys:

GRUB has a lot of features possible of which one may prove useful when typing extra options from a non-US keyboard. By entering the command `setkey` for each key translation in `/boot/grub/menu.lst` the user can use the keyboard.
eg. (To recognize part of the German keyboard)

```
setkey z y
setkey y z
```

Selection of a default menu entry only once for the next booting.

`grubonce 2` (Script in SuSE only)

or

```
echo "savedefault --stage2=/boot/grub/stage2 --default=2 --once quit" \
| grub --batch
```

The next time the computer boots(and only the next time) the default boot item from the grub menu will be the **third item** (0,1,2...).

- **Remapping the assignment of physical partitions for Windows95/98**
  Useful to let windows 98 boot from a partition which is not the first one (picky fellow!!) and make it think that it is.
  Eg. Swapping the first 2 partitions of a drive:
  ```
  map (hd0,1) (hd0,0)
  map (hd0,0) (hd0,1)
  ```

- **Creating Boot CDs**
  If problems occur booting your system using a boot manager or if the boot manager cannot be installed on the MBR of your hard disk or a floppy disk, it is also possible to create a bootable CD with all the necessary start-up files for Linux. This requires a CD writer installed in your system.

  Creating a bootable CD-ROM with GRUB merely requires a special form of *stage2* called `stage2_eltorito` and, optionally, a customized `menu.lst`.
  Note: The classic files `stage1` and `stage2` are not required.

  Create a directory in which to create the ISO image, for example, with:
  ```
  mkdir -p /tmp/iso/boot/grub
  ```

  Copy the file `stage2_eltorito` into the directory `grub`:
  ```
  cp /usr/lib/grub/stage2_eltorito /tmp/iso/boot/grub
  ```

  Also copy the kernel (`/boot/vmlinuz`), the `initrd` (`/boot/initrd`), and the file `/boot/message` to `iso/boot/`:
  ```
  cp /boot/vmlinuz /tmp/iso/boot/
  cp /boot/initrd /tmp/iso/boot/
  cp /boot/message /tmp/iso/boot/
  ```

  To make them available to GRUB, copy the file `menu.lst` to `iso/boot/grub` and adjust the path entries to make them point to a CD-ROM device.
  Do this by replacing the device name of the hard disks, listed in the format `(hd*)`, in the pathnames with the device name of the CD-ROM drive, which is `(cd)`:

```
gfxmenu (cd)/boot/message
timeout 8
default 0

title Linux
    kernel (cd)/boot/vmlinuz root=/dev/hda5 vga=794 resume=/dev/hda1
splash=verbose showopts
    initrd (cd)/boot/initrd
```

  Finally, create the ISO image with the following command:

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

  Then write the resulting file `grub.iso` to a CD using your preferred utility