System Startup and Shutdown

System Startup

Boot Loader

The most commonly used boot loader is called **GRUB** (**G**rand **U**nified **B**oot **L**oader), however historically there have been others:

- LILO (**LI**nux **LO**ader)
- SysLinux
- LoadLin
- …

BIOS Systems

The Boot Loader is generally placed in the MBR (**M**aster **B**oot **Record**) of the disk on which the system to-be-booted resides. The MBR format is as follows:

- 446 bytes occupied by the boot loader,
- 64 bytes for the Partition Table. In other words 16 bytes per primary partition,
- 2 bytes with a fixed hexadecimal value of **AA55**.

<note imporatant> Note that you can also install the boot loader in what is known as the **PBR** (**P**artition **B**oot **R**ecord). </note>

EFI Systems

Since 2011, the BIOS is being steadily replaced by **UEFI** (**U**nified **E**xtensible **F**irmware **I**nterface). Systems using a CPU other than the x86 or the x86-64 use non-BIOS software such as **OpenFirmware** or **EFI** .

EFI relies on boot loaders stored in a disk partition called the **EFI System Partition** or **ESP**. This partition is normally mounted by Linux at **/boot/efi**. The boot loaders reside in files having a .efi extension stored in subdirectories named after the OS to be booted.

The EFI firmware includes a boot manager that enables you to choose which OS to boot. In order for EFI to work each boot loader must be registered with the firmware.

GRUB

GRUB currently exists in two versions:

- GRUB LEGACY
- GRUB 2

GRUB LEGACY no longer benefits from any development. Indeed the last stable version is **0.97**.

If **GRUB** is not installed in the MBR, it can be by using either one of the following commands :

- grub-install /dev/<special-file>
- grub-install '(hdX)'

where **special-file** represents the device, for example sda or hda, or **hdX** represents the hard disk number, as seen by GRUB, where GRUB needs to be installed.

<note important> To un-install GRUB, you can either use the Linux **dd** command or a DOS boot floppy (DOS or W9X) with the following command **A> fdisk /mbr**. </note>

GRUB LEGACY on Red Hat/CentOS

GRUB LEGACY has it's configuration in the **/boot/grub/menu.lst** file.

<note important> Some distributions do not have a **/boot/grub/menu.lst** file but rather a **/boot/grub/grub.conf** file. </note>

To view the contents of the **/boot/grub/menu.lst** file, use the following command:

```
[root@centos ~]# cat /boot/grub/menu.lst
# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to this file
# NOTICE:  You have a /boot partition.  This means that
#          all kernel and initrd paths are relative to /boot/, eg.
#          root (hd0,0)
#          kernel /vmlinuz-version ro root=/dev/sda2
#          initrd /initrd-[generic-]version.img
#boot=/dev/sda
default=0
timeout=5
splashimage=(hd0,0)/grub/splash.xpm.gz
hiddenmenu
title CentOS (2.6.32-358.23.2.el6.i686)
    root (hd0,0)
    kernel /vmlinuz-2.6.32-358.23.2.el6.i686 ro root=UUID=c7b1d3e8-6471-4cba-947b-430db974e774 rd_NO_LUKS
KEYBOARDTYPE=pc KEYTABLE=fr LANG=en_US.UTF-8 rd_NO_MD SYSFONT=latarcyrheb-sun16 crashkernel=128M rd_NO_LVM
rd_NO_DM rhgb quiet
    initrd /initramfs-2.6.32-358.23.2.el6.i686.img
title CentOS (2.6.32-358.el6.i686)
    root (hd0,0)
    kernel /vmlinuz-2.6.32-358.el6.i686 ro root=UUID=c7b1d3e8-6471-4cba-947b-430db974e774 rd_NO_LUKS
KEYBOARDTYPE=pc KEYTABLE=fr LANG=en_US.UTF-8 rd_NO_MD SYSFONT=latarcyrheb-sun16 crashkernel=128M rd_NO_LVM
rd_NO_DM rhgb quiet
    initrd /initramfs-2.6.32-358.el6.i686.img
```

The directives in the above file are as follows:

Directive	**Description**
default	Indicates the default OS to boot. A value of 0 indicates the OS referenced by the first stanza commencing with the **title** keyword.
timeout	Indicates the time to wait for user input prior to booting the default OS.
splashimage	Indicates the graphics file that will be displayed during the boot process.

Directive	Description
hiddenmenu	Indicates that the GRUB menu will be hidden unless a key is pushed.
title	Indicates the beginning of a per image stanza.
root	Indicates the position of GRUB's root partition. In this case the first partition of the first disk.
kernel	Indicates the location of the Linux kernel and any options that need to be passed to it. This location is relative to GRUB's root partition.
initrd	Indicates the location of the **INIT**ial **R**am **D**isk.

GRUB LEGACY on OpenSUSE

GRUB LEGACY is very similar on openSUSE to the Red Hat/CentOS version :

```
opensuse:~ # cat /boot/grub/menu.lst
# Modified by YaST2. Last modification on Tue May 17 15:03:57 CEST 2011
# THIS FILE WILL BE PARTIALLY OVERWRITTEN by perl-Bootloader
# Configure custom boot parameters for updated kernels in /etc/sysconfig/bootloader

default 0
timeout 8
##YaST - generic_mbr
gfxmenu (hd0,0)/message
##YaST - activate

###Don't change this comment - YaST2 identifier: Original name: linux###
title Desktop -- openSUSE 11.4 - 2.6.37.6-0.5
    root (hd0,0)
    kernel /vmlinuz-2.6.37.6-0.5-desktop root=/dev/sda2 resume=/dev/sda3 splash=silent quiet showopts vga=0x314
    initrd /initrd-2.6.37.6-0.5-desktop

###Don't change this comment - YaST2 identifier: Original name: failsafe###
title Failsafe -- openSUSE 11.4 - 2.6.37.6-0.5
    root (hd0,0)
    kernel /vmlinuz-2.6.37.6-0.5-desktop root=/dev/sda2 showopts apm=off noresume nosmp maxcpus=0 edd=off
powersaved=off nohz=off highres=off processor.max_cstate=1 nomodeset x11failsafe vga=0x314
```

```
    initrd /initrd-2.6.37.6-0.5-desktop
```

GRUB 2 on Debian Squeeze

GRUB 2 is a complete rewrite of GRUB LEGACY. It is possible to upgrade from GRUB LEGACY to GRUB 2 by using the following command:

```
# upgrade-from-grub-legacy
```

GRUB 2 is modular in design. The modules can be found in the **/boot/grub** directory:

```
root@debian:~# ls /boot/grub
915resolution.mod        gcry_seed.mod     part_sunpc.mod
acpi.mod             gcry_serpent.mod     parttool.lst
affs.mod             gcry_sha1.mod     parttool.mod
afs_be.mod           gcry_sha256.mod     password.mod
afs.mod              gcry_sha512.mod     password_pbkdf2.mod
aout.mod             gcry_tiger.mod     pbkdf2.mod
ata.mod              gcry_twofish.mod     pci.mod
ata_pthru.mod            gcry_whirlpool.mod  play.mod
at_keyboard.mod          gettext.mod     png.mod
befs_be.mod          gfxmenu.mod     probe.mod
befs.mod             gfxterm.mod     pxeboot.img
biosdisk.mod             gptsync.mod     pxecmd.mod
bitmap.mod           grldr.img         pxe.mod
bitmap_scale.mod         grub.cfg         raid5rec.mod
blocklist.mod            grubenv         raid6rec.mod
boot.img             gzio.mod         raid.mod
boot.mod             halt.mod         read.mod
bsd.mod              handler.lst     reboot.mod
bufio.mod            hashsum.mod     regexp.mod
cat.mod              hdparm.mod         reiserfs.mod
cdboot.img           hello.mod         relocator.mod
```

```
chain.mod            help.mod         scsi.mod
cmostest.mod             hexdump.mod     search_fs_file.mod
cmp.mod              hfs.mod         search_fs_uuid.mod
command.lst          hfsplus.mod     search_label.mod
configfile.mod           iorw.mod         search.mod
core.img             iso9660.mod     serial.mod
cpio.mod             jfs.mod         setjmp.mod
cpuid.mod            jpeg.mod         setpci.mod
crc.mod              kernel.img        sfs.mod
crypto.lst           keystatus.mod     sleep.mod
crypto.mod           linux16.mod     tar.mod
cs5536.mod           linux.mod         terminal.lst
datehook.mod             lnxboot.img     terminal.mod
date.mod             loadenv.mod     terminfo.mod
datetime.mod             locale         test.mod
device.map           loopback.mod     tga.mod
diskboot.img             lsmmap.mod         trig.mod
dm_nv.mod            ls.mod         true.mod
drivemap.mod             lspci.mod         udf.mod
echo.mod             lvm.mod         ufs1.mod
efiemu32.o           mdraid.mod        ufs2.mod
efiemu64.o           memdisk.mod     uhci.mod
efiemu.mod           memrw.mod         usb_keyboard.mod
elf.mod              minicmd.mod     usb.mod
example_functional_test.mod  minix.mod       usbms.mod
ext2.mod             mmap.mod         usbtest.mod
extcmd.mod           moddep.lst        vbeinfo.mod
fat.mod              msdospart.mod     vbe.mod
font.mod             multiboot2.mod    vbetest.mod
fshelp.mod           multiboot.mod     vga.mod
fs.lst               nilfs2.mod        vga_text.mod
functional_test.mod      normal.mod         video_bochs.mod
gcry_arcfour.mod         ntfscomp.mod     video_cirrus.mod
gcry_blowfish.mod        ntfs.mod         video_fb.mod
```

```
gcry_camellia.mod        ohci.mod         video.lst
gcry_cast5.mod           part_acorn.mod     video.mod
gcry_crc.mod             part_amiga.mod     videotest.mod
gcry_des.mod             part_apple.mod     xfs.mod
gcry_md4.mod             part_bsd.mod     xnu.mod
gcry_md5.mod             part_gpt.mod     xnu_uuid.mod
gcry_rfc2268.mod         partmap.lst     zfsinfo.mod
gcry_rijndael.mod        part_msdos.mod     zfs.mod
gcry_rmd160.mod          part_sun.mod
```

Grub2 reads its entries from the **/boot/grub/grub.cfg** file:

```
root@debian:~# cat /boot/grub/grub.cfg
#
# DO NOT EDIT THIS FILE
#
# It is automatically generated by grub-mkconfig using templates
# from /etc/grub.d and settings from /etc/default/grub
#

### BEGIN /etc/grub.d/00_header ###
if [ -s $prefix/grubenv ]; then
  load_env
fi
set default="0"
if [ "${prev_saved_entry}" ]; then
  set saved_entry="${prev_saved_entry}"
  save_env saved_entry
  set prev_saved_entry=
  save_env prev_saved_entry
  set boot_once=true
fi

function savedefault {
  if [ -z "${boot_once}" ]; then
    saved_entry="${chosen}"
    save_env saved_entry
  fi
}

function load_video {
  insmod vbe
  insmod vga
  insmod video_bochs
  insmod video_cirrus
}

insmod part_msdos
insmod ext2
set root='(hd0,msdos1)'
search --no-floppy --fs-uuid --set a42a1ddd-14bc-4dde-a537-e6c1b984a782
if loadfont /usr/share/grub/unicode.pf2 ; then
  set gfxmode=640x480
  load_video
  insmod gfxterm
fi
terminal_output gfxterm
insmod part_msdos
insmod ext2
set root='(hd0,msdos1)'
search --no-floppy --fs-uuid --set a42a1ddd-14bc-4dde-a537-e6c1b984a782
set locale_dir=($root)/boot/grub/locale
set lang=fr
insmod gettext
set timeout=5
### END /etc/grub.d/00_header ###

### BEGIN /etc/grub.d/05_debian_theme ###
```

```
insmod part_msdos
insmod ext2
set root='(hd0,msdos1)'
search --no-floppy --fs-uuid --set a42a1ddd-14bc-4dde-a537-e6c1b984a782
insmod png
if background_image /usr/share/images/desktop-base/spacefun-grub.png; then
  set color_normal=light-gray/black
  set color_highlight=white/black
else
  set menu_color_normal=cyan/blue
  set menu_color_highlight=white/blue
fi
### END /etc/grub.d/05_debian_theme ###

### BEGIN /etc/grub.d/10_linux ###
menuentry 'Debian GNU/Linux, avec Linux 2.6.32-5-686' --class debian --class gnu-linux --class gnu --class os {
    insmod part_msdos
    insmod ext2
    set root='(hd0,msdos1)'
    search --no-floppy --fs-uuid --set a42a1ddd-14bc-4dde-a537-e6c1b984a782
    echo    'Chargement de Linux 2.6.32-5-686 ...'
    linux    /boot/vmlinuz-2.6.32-5-686 root=UUID=a42a1ddd-14bc-4dde-a537-e6c1b984a782 ro  quiet
    echo    'Chargement du disque mémoire initial ...'
    initrd    /boot/initrd.img-2.6.32-5-686
}
menuentry 'Debian GNU/Linux, avec Linux 2.6.32-5-686 (mode de dépannage)' --class debian --class gnu-linux --
class gnu --class os {
    insmod part_msdos
    insmod ext2
    set root='(hd0,msdos1)'
    search --no-floppy --fs-uuid --set a42a1ddd-14bc-4dde-a537-e6c1b984a782
    echo    'Chargement de Linux 2.6.32-5-686 ...'
    linux    /boot/vmlinuz-2.6.32-5-686 root=UUID=a42a1ddd-14bc-4dde-a537-e6c1b984a782 ro single
    echo    'Chargement du disque mémoire initial ...'
    initrd    /boot/initrd.img-2.6.32-5-686
}
### END /etc/grub.d/10_linux ###

### BEGIN /etc/grub.d/20_linux_xen ###
### END /etc/grub.d/20_linux_xen ###

### BEGIN /etc/grub.d/30_os-prober ###
### END /etc/grub.d/30_os-prober ###

### BEGIN /etc/grub.d/40_custom ###
# This file provides an easy way to add custom menu entries.  Simply type the
# menu entries you want to add after this comment.  Be careful not to change
# the 'exec tail' line above.
### END /etc/grub.d/40_custom ###

### BEGIN /etc/grub.d/41_custom ###
if [ -f  $prefix/custom.cfg ]; then
  source $prefix/custom.cfg;
fi
### END /etc/grub.d/41_custom ###
```

In this file, the section that is of the most interest is:

```
...
### BEGIN /etc/grub.d/10_linux ###
menuentry 'Debian GNU/Linux, avec Linux 2.6.32-5-686' --class debian --class gnu-linux --class gnu --class os {
    insmod part_msdos
    insmod ext2
    set root='(hd0,msdos1)'
    search --no-floppy --fs-uuid --set a42a1ddd-14bc-4dde-a537-e6c1b984a782
    echo    'Chargement de Linux 2.6.32-5-686 ...'
    linux    /boot/vmlinuz-2.6.32-5-686 root=UUID=a42a1ddd-14bc-4dde-a537-e6c1b984a782 ro  quiet
    echo    'Chargement du disque mémoire initial ...'
    initrd    /boot/initrd.img-2.6.32-5-686
}
menuentry 'Debian GNU/Linux, avec Linux 2.6.32-5-686 (mode de dépannage)' --class debian --class gnu-linux --
class gnu --class os {
    insmod part_msdos
    insmod ext2
    set root='(hd0,msdos1)'
    search --no-floppy --fs-uuid --set a42a1ddd-14bc-4dde-a537-e6c1b984a782
    echo    'Chargement de Linux 2.6.32-5-686 ...'
    linux    /boot/vmlinuz-2.6.32-5-686 root=UUID=a42a1ddd-14bc-4dde-a537-e6c1b984a782 ro single
    echo    'Chargement du disque mémoire initial ...'
    initrd    /boot/initrd.img-2.6.32-5-686
}
...
```

<note important> Note that in the above example an opening curly bracket follows the menu title and each entry is ended with a closing curly bracket. Equally important and also very confusing is that disk numbers now start at **1** as opposed to **0** with LEGACY GRUB whilst partition numbers **still** start at **0** as in the case of LEGACY GRUB. </note>

By comparision with the previous **/boot/grub/menu.lst** file, it is possible to identify the following correspondances between LEGACY GRUB and GRUB 2:

GRUB LEGACY	GRUB 2	Comments
title	menuentry	Menu entry in GRUB 2 is enclosed in quotation marks (single or double)
root	set root	With GRUB 2 an equal sign seperates **root** from the partition specification

This file must note be edited manually. It is automatically created by using the **update-grub** command. When executed this command uses the information from several other files:

The /boot/grub/device.map file

```
root@debian:~# cat /boot/grub/device.map
```

```
(hd0)   /dev/disk/by-id/ata-VBOX_HARDDISK_VB5bb9d489-1757bae6
```

<note important> NOte that the **/boot/grub/device.map** file can be manually edited or automatically configured by using the **grub-mkdevicemap** command. </note>

The /etc/default/grub file

This file contains the default global configuration for GRUB 2:

```
root@debian:~# cat /etc/default/grub
# If you change this file, run 'update-grub' afterwards to update
# /boot/grub/grub.cfg.

GRUB_DEFAULT=0
GRUB_TIMEOUT=5
GRUB_DISTRIBUTOR=`lsb_release -i -s 2> /dev/null || echo Debian`
GRUB_CMDLINE_LINUX_DEFAULT="quiet"
GRUB_CMDLINE_LINUX=""

# Uncomment to enable BadRAM filtering, modify to suit your needs
# This works with Linux (no patch required) and with any kernel that obtains
# the memory map information from GRUB (GNU Mach, kernel of FreeBSD ...)
#GRUB_BADRAM="0x01234567,0xfefefefe,0x89abcdef,0xefefefef"

# Uncomment to disable graphical terminal (grub-pc only)
#GRUB_TERMINAL=console

# The resolution used on graphical terminal
# note that you can use only modes which your graphic card supports via VBE
# you can see them in real GRUB with the command `vbeinfo'
#GRUB_GFXMODE=640x480

# Uncomment if you don't want GRUB to pass "root=UUID=xxx" parameter to Linux
```

```
#GRUB_DISABLE_LINUX_UUID=true

# Uncomment to disable generation of recovery mode menu entries
#GRUB_DISABLE_LINUX_RECOVERY="true"

# Uncomment to get a beep at grub start
#GRUB_INIT_TUNE="480 440 1"
```

<note important> Any change made to this file requires the execution of the **update-grub** command in order for the changes to become effective. </note>

The directives in the above file are as follows:

Directive	**Description**
GRUB_DEFAULT	Indicates the default OS to boot. A value of 0 indicates the OS referenced by the first stanza commencing with the **menuentry** keyword
GRUB_TIMEOUT	Indicates the time to wait for user input prior to booting the default OS
GRUB_DISTRIBUTOR	Set by distributors of GRUB to their identifying name. This is used to generate more informative menu entry titles.
GRUB_CMDLINE_LINUX_DEFAULT	Unless ‘GRUB_DISABLE_LINUX_RECOVERY’ is set to ‘true’, two menu entries will be generated for each Linux kernel: one default entry and one entry for recovery mode. This option lists command-line arguments to add only to the default menu entry, after those listed in ‘GRUB_CMDLINE_LINUX’.
GRUB_CMDLINE_LINUX	Command-line arguments to add to menu entries for the Linux kernel.
GRUB_TERMINAL	Indicates the terminal that will beused for input/output.
GRUB_GFXMODE	Set the resolution used on the ‘gfxterm’ graphical terminal. Note that you can only use modes which your graphics card supports via VESA BIOS Extensions (VBE), so for example native LCD panel resolutions may not be available. The default is ‘auto’, which tries to select a preferred resolution.
GRUB_DISABLE_LINUX_UUID	Normally, grub-mkconfig will generate menu entries that use universally-unique identifiers (UUIDs) to identify the root filesystem to the Linux kernel, using a ‘root=UUID=…’ kernel parameter. This is usually more reliable, but in some cases it may not be appropriate. To disable the use of UUIDs, set this option to ‘true’.
GRUB_DISABLE_LINUX_RECOVERY	If this option is set to ‘true’, disable the generation of recovery mode menu entries.
GRUB_INIT_TUNE	Play a tune on the speaker when GRUB starts. This is particularly useful for users unable to see the screen. The value of this option is passed directly to play.
GRUB_BADRAM	If this option is set, GRUB will issue a badram command to filter out specified regions of RAM.

Files in the /etc/grub.d directory

The files in this directory are executed in a numerical order and are used by the **update-grub** command to build the stanzas in the **/boot/grub/grub.cfg** file:

```
root@debian:~# ls -l /etc/grub.d
total 52
-rwxr-xr-x 1 root root 6433 18 janv.  2011 00_header
-rwxr-xr-x 1 root root 5343 17 janv.  2011 05_debian_theme
-rwxr-xr-x 1 root root 4284 18 janv.  2011 10_linux
-rwxr-xr-x 1 root root 4925 18 janv.  2011 20_linux_xen
-rwxr-xr-x 1 root root 5789 18 janv.  2011 30_os-prober
-rwxr-xr-x 1 root root  214 18 janv.  2011 40_custom
-rwxr-xr-x 1 root root   95 18 janv.  2011 41_custom
-rw-r--r-- 1 root root  483 18 janv.  2011 README
```

The /etc/grub.d/10_Linux file

This script looks for Linux kernels present on the system and builds the appropriate stanzas:

```
root@debian:~# cat /etc/grub.d/10_linux
#! /bin/sh
set -e

# grub-mkconfig helper script.
# Copyright (C) 2006,2007,2008,2009,2010  Free Software Foundation, Inc.
#
# GRUB is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# GRUB is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with GRUB.  If not, see <http://www.gnu.org/licenses/>.

prefix=/usr
exec_prefix=${prefix}
bindir=${exec_prefix}/bin
libdir=${exec_prefix}/lib
. ${libdir}/grub/grub-mkconfig_lib

export TEXTDOMAIN=grub
export TEXTDOMAINDIR=${prefix}/share/locale

CLASS="--class gnu-linux --class gnu --class os"

if [ "x${GRUB_DISTRIBUTOR}" = "x" ] ; then
  OS=GNU/Linux
else
  OS="${GRUB_DISTRIBUTOR} GNU/Linux"
  CLASS="--class $(echo ${GRUB_DISTRIBUTOR} | tr '[A-Z]' '[a-z]' | cut -d' ' -f1) ${CLASS}"
fi

# loop-AES arranges things so that /dev/loop/X can be our root device, but
# the initrds that Linux uses don't like that.
case ${GRUB_DEVICE} in
  /dev/loop/*|/dev/loop[0-9])
    GRUB_DEVICE=`losetup ${GRUB_DEVICE} | sed -e "s/^[^(]*(\([^)]\+\)).*/\1/"`
  ;;
esac

if [ "x${GRUB_DEVICE_UUID}" = "x" ] || [ "x${GRUB_DISABLE_LINUX_UUID}" = "xtrue" ] \
    || ! test -e "/dev/disk/by-uuid/${GRUB_DEVICE_UUID}" \
    || uses_abstraction "${GRUB_DEVICE}" lvm; then
  LINUX_ROOT_DEVICE=${GRUB_DEVICE}
else
  LINUX_ROOT_DEVICE=UUID=${GRUB_DEVICE_UUID}
fi

linux_entry ()
{
  os="$1"
  version="$2"
  recovery="$3"
  args="$4"
  if ${recovery} ; then
    title="$(gettext_quoted "%s, with Linux %s (recovery mode)")"
  else
    title="$(gettext_quoted "%s, with Linux %s")"
  fi
  printf "menuentry '${title}' ${CLASS} {\n" "${os}" "${version}"
  save_default_entry | sed -e "s/^/\t/"

  # Use ELILO's generic "efifb" when it's known to be available.
  # FIXME: We need an interface to select vesafb in case efifb can't be used.
  if [ "x$GRUB_GFXPAYLOAD_LINUX" != x ]; then
      cat << EOF
    set gfxpayload=$GRUB_GFXPAYLOAD_LINUX
EOF
  fi

  if [ -z "${prepare_boot_cache}" ]; then
    prepare_boot_cache="$(prepare_grub_to_access_device ${GRUB_DEVICE_BOOT} | sed -e "s/^/\t/")"
  fi
  printf '%s\n' "${prepare_boot_cache}"
  message="$(gettext_printf "Loading Linux %s ..." ${version})"
  cat << EOF
    echo    '$message'
    linux    ${rel_dirname}/${basename} root=${linux_root_device_thisversion} ro ${args}
EOF
  if test -n "${initrd}" ; then
    message="$(gettext_printf "Loading initial ramdisk ...")"
    cat << EOF
    echo    '$message'
    initrd    ${rel_dirname}/${initrd}
EOF
  fi
  cat << EOF
}
EOF
}

list=`for i in /boot/vmlinu[zx]-* /vmlinu[zx]-* ; do
        if grub_file_is_not_garbage "$i" ; then echo -n "$i " ; fi
      done`
prepare_boot_cache=

while [ "x$list" != "x" ] ; do
  linux=`version_find_latest $list`
  echo "Found linux image: $linux" >&2
  basename=`basename $linux`
  dirname=`dirname $linux`
  rel_dirname=`make_system_path_relative_to_its_root $dirname`
  version=`echo $basename | sed -e "s,^[^0-9]*-,,g"`
  alt_version=`echo $version | sed -e "s,\.old$,,g"`
  linux_root_device_thisversion="${LINUX_ROOT_DEVICE}"

  initrd=
  for i in "initrd.img-${version}" "initrd-${version}.img" \
       "initrd-${version}" "initramfs-${version}.img" \
       "initrd.img-${alt_version}" "initrd-${alt_version}.img" \
       "initrd-${alt_version}" "initramfs-${alt_version}.img"; do
    if test -e "${dirname}/${i}" ; then
      initrd="$i"
      break
    fi
  done
  if test -n "${initrd}" ; then
    echo "Found initrd image: ${dirname}/${initrd}" >&2
  else
    # "UUID=" magic is parsed by initrds.  Since there's no initrd, it can't work here.
    linux_root_device_thisversion=${GRUB_DEVICE}
  fi

  linux_entry "${OS}" "${version}" false \
      "${GRUB_CMDLINE_LINUX} ${GRUB_CMDLINE_LINUX_DEFAULT}"
  if [ "x${GRUB_DISABLE_LINUX_RECOVERY}" != "xtrue" ]; then
    linux_entry "${OS}" "${version}" true \
    "single ${GRUB_CMDLINE_LINUX}"
  fi

  list=`echo $list | tr ' ' '\n' | grep -vx $linux | tr '\n' ' '`
done
```

Le fichier /etc/grub.d/30_os-prober

This script looks for other operating systems present on the system and builds the appropriate stanzas:

```
root@debian:~# cat /etc/grub.d/30_os-prober
#! /bin/sh
set -e

# grub-mkconfig helper script.
# Copyright (C) 2006,2007,2008,2009  Free Software Foundation, Inc.
```

```
#
# GRUB is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# GRUB is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with GRUB.  If not, see <http://www.gnu.org/licenses/>.

prefix=/usr
exec_prefix=${prefix}
libdir=${exec_prefix}/lib

. ${libdir}/grub/grub-mkconfig_lib

if [ "x${GRUB_DISABLE_OS_PROBER}" = "xtrue" ]; then
  exit 0
fi

if [ -z "`which os-prober 2> /dev/null`" -o -z "`which linux-boot-prober 2> /dev/null`" ] ; then
  # missing os-prober and/or linux-boot-prober
  exit 0
fi

OSPROBED="`os-prober | tr ' ' '^' | paste -s -d ' '`"
if [ -z "${OSPROBED}" ] ; then
  # empty os-prober output, nothing doing
  exit 0
fi
```

```
osx_entry() {
        cat << EOF
menuentry "${LONGNAME} (${2}-bit) (on ${DEVICE})" {
EOF
    save_default_entry | sed -e "s/^/\t/"
    prepare_grub_to_access_device ${DEVICE} | sed -e "s/^/\t/"
    cat << EOF
        load_video
        set do_resume=0
        if [ /var/vm/sleepimage -nt10 / ]; then
           if xnu_resume /var/vm/sleepimage; then
             set do_resume=1
           fi
        fi
        if [ \$do_resume = 0 ]; then
           xnu_uuid ${OSXUUID} uuid
           if [ -f /Extra/DSDT.aml ]; then
              acpi -e /Extra/DSDT.aml
           fi
           $1 /mach_kernel boot-uuid=\${uuid} rd=*uuid
           if [ /System/Library/Extensions.mkext -nt /System/Library/Extensions ]; then
              xnu_mkext /System/Library/Extensions.mkext
           else
              xnu_kextdir /System/Library/Extensions
           fi
           if [ -f /Extra/Extensions.mkext ]; then
              xnu_mkext /Extra/Extensions.mkext
           fi
           if [ -d /Extra/Extensions ]; then
              xnu_kextdir /Extra/Extensions
           fi
           if [ -f /Extra/devprop.bin ]; then
              xnu_devprop_load /Extra/devprop.bin
           fi
           if [ -f /Extra/splash.jpg ]; then
              insmod jpeg
              xnu_splash /Extra/splash.jpg
           fi
           if [ -f /Extra/splash.png ]; then
              insmod png
              xnu_splash /Extra/splash.png
           fi
           if [ -f /Extra/splash.tga ]; then
              insmod tga
              xnu_splash /Extra/splash.tga
           fi
        fi
}
EOF
}

for OS in ${OSPROBED} ; do
  DEVICE="`echo ${OS} | cut -d ':' -f 1`"
  LONGNAME="`echo ${OS} | cut -d ':' -f 2 | tr '^' ' '`"
  LABEL="`echo ${OS} | cut -d ':' -f 3 | tr '^' ' '`"
  BOOT="`echo ${OS} | cut -d ':' -f 4`"

  if [ -z "${LONGNAME}" ] ; then
    LONGNAME="${LABEL}"
  fi

  echo "Found ${LONGNAME} on ${DEVICE}" >&2

  case ${BOOT} in
    chain)

      cat << EOF
menuentry "${LONGNAME} (on ${DEVICE})" {
```

```
EOF
      save_default_entry | sed -e "s/^/\t/"
      prepare_grub_to_access_device ${DEVICE} | sed -e "s/^/\t/"

      case ${LONGNAME} in
    Windows\ Vista*|Windows\ 7*)
    ;;
    *)
      cat << EOF
    drivemap -s (hd0) \${root}
EOF
    ;;
      esac

      cat <<EOF
    chainloader +1
}
EOF
    ;;
    linux)
      LINUXPROBED="`linux-boot-prober ${DEVICE} 2> /dev/null | tr ' ' '^' | paste -s -d ' '`"
      prepare_boot_cache=

      for LINUX in ${LINUXPROBED} ; do
        LROOT="`echo ${LINUX} | cut -d ':' -f 1`"
        LBOOT="`echo ${LINUX} | cut -d ':' -f 2`"
        LLABEL="`echo ${LINUX} | cut -d ':' -f 3 | tr '^' ' '`"
        LKERNEL="`echo ${LINUX} | cut -d ':' -f 4`"
        LINITRD="`echo ${LINUX} | cut -d ':' -f 5`"
        LPARAMS="`echo ${LINUX} | cut -d ':' -f 6- | tr '^' ' '`"

        if [ -z "${LLABEL}" ] ; then
          LLABEL="${LONGNAME}"
        fi
    if [ "${LROOT}" != "${LBOOT}" ]; then
      LKERNEL="${LKERNEL#/boot}"
      LINITRD="${LINITRD#/boot}"
    fi

        cat << EOF
menuentry "${LLABEL} (on ${DEVICE})" {
EOF
    save_default_entry | sed -e "s/^/\t/"
    if [ -z "${prepare_boot_cache}" ]; then
      prepare_boot_cache="$(prepare_grub_to_access_device ${LBOOT} | sed -e "s/^/\t/")"
    fi
    printf '%s\n' "${prepare_boot_cache}"
    cat <<  EOF
    linux ${LKERNEL} ${LPARAMS}
EOF
        if [ -n "${LINITRD}" ] ; then
          cat << EOF
    initrd ${LINITRD}
EOF
        fi
        cat << EOF
}
EOF
      done
    ;;
    macosx)
      OSXUUID="`grub-probe --target=fs_uuid --device ${DEVICE} 2> /dev/null`"
      osx_entry xnu_kernel 32
      osx_entry xnu_kernel64 64
    ;;
    hurd)
      cat << EOF
menuentry "${LONGNAME} (on ${DEVICE})" {
```

```
EOF
      save_default_entry | sed -e "s/^/\t/"
      prepare_grub_to_access_device ${DEVICE} | sed -e "s/^/\t/"
      grub_device="`${grub_probe} --device ${DEVICE} --target=drive`"
      mach_device="`echo "${grub_device}" | tr -d '()' | tr , s`"
      grub_fs="`${grub_probe} --device ${DEVICE} --target=fs`"
      case "${grub_fs}" in
    *fs)    hurd_fs="${grub_fs}" ;;
    *)    hurd_fs="${grub_fs}fs" ;;
      esac
      cat << EOF
    multiboot /boot/gnumach.gz root=device:${mach_device}
    module /hurd/${hurd_fs}.static ${hurd_fs} --readonly \\
            --multiboot-command-line='\${kernel-command-line}' \\
            --host-priv-port='\${host-port}' \\
            --device-master-port='\${device-port}' \\
            --exec-server-task='\${exec-task}' -T typed '\${root}' \\
            '\$(task-create)' '\$(task-resume)'
    module /lib/ld.so.1 exec /hurd/exec '\$(exec-task=task-create)'
}
EOF
    ;;
    *)
      echo "  ${LONGNAME} is not yet supported by grub-mkconfig." >&2
    ;;
  esac
done
```

The /etc/grub.d/40_custom and /etc/grub.d/41_custom files

These two scripts are supplied so that thay can be personalised in order to build the appropriate stanzas:

```
root@debian:~# cat /etc/grub.d/40_custom
#!/bin/sh
```

```
exec tail -n +3 $0
# This file provides an easy way to add custom menu entries.  Simply type the
# menu entries you want to add after this comment.  Be careful not to change
# the 'exec tail' line above.
root@debian:~# cat /etc/grub.d/41_custom
#!/bin/sh
cat <<EOF
if [ -f  \$prefix/custom.cfg ]; then
  source \$prefix/custom.cfg;
fi
EOF
```

<note important> We will look closely at personalising one of these files in the **Initramfs on Debian** section of this unit. </note>

Initramfs

The **Initramfs** *INITial Ram File System* file is a minimal system image which is initialised upon system boot.

The file format is **cramFS** that is to say archived using **cpio** and compressed with **gzip**.

Initramfs on Redhat / CentOS

To examine the current image, first copy it to /tmp whilst renaming it to **custom.gz** and then uncompress it:

```
[root@centos ~]# cp /boot/initramfs-2.6.32-358.23.2.el6.i686.img /tmp/custom.gz
[root@centos ~]# gunzip /tmp/custom.gz
```

Now extract the cpio archive as follows:

```
[root@centos ~]# cd /tmp
[root@centos tmp]# mkdir initrd
```

```
[root@centos tmp]# cd initrd/
[root@centos initrd]# cpio -cid -I ../custom
71576 blocks
```

Now install the **tree** package using **yum**:

```
[root@centos initrd]# yum install tree
Loaded plugins: fastestmirror, refresh-packagekit, security
Loading mirror speeds from cached hostfile
 * atomic: atomic.mirror.uber.com.au
 * base: centos.crazyfrogs.org
 * epel: mirror.vutbr.cz
 * extras: centos.mirror.fr.planethoster.net
 * rpmforge: apt.sw.be
 * updates: centos.crazyfrogs.org
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package tree.i686 0:1.5.3-2.el6 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

=================================================================================================================
=======================================================
 Package                               Arch                                  Version
Repository                             Size
=================================================================================================================
=======================================================
Installing:
 tree                                  i686                                  1.5.3-2.el6
base                                   36 k

Transaction Summary
```

```
=================================================================================================================
=======================================================
Install       1 Package(s)

Total download size: 36 k
Installed size: 63 k
Is this ok [y/N]: y
Downloading Packages:
tree-1.5.3-2.el6.i686.rpm
|  36 kB     00:00
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing : tree-1.5.3-2.el6.i686
1/1
  Verifying  : tree-1.5.3-2.el6.i686
1/1

Installed:
  tree.i686 0:1.5.3-2.el6

Complete!
```

Now use the tree commande to examine the contents of the image:

```
[root@centos initrd]# tree | more
.
├── bin
│   ├── basename
│   ├── cat
│   ├── cp
│   ├── dash
│   ├── dd
│   ├── dmesg
│   ├── grep
│   ├── gzip
│   ├── ln
│   ├── loadkeys
│   ├── ls
│   ├── mkdir
│   ├── mknod
│   ├── mount
│   ├── mv
│   ├── plymouth
│   ├── plymouthd
│   ├── readlink
│   ├── rm
│   ├── sed
│   ├── setfont
│   ├── sh -> dash
│   ├── sleep
│   ├── umount
│   └── uname
├── cmdline
│   ├── 01parse-kernel.sh
│   ├── 01version.sh
│   ├── 10parse-resume.sh
│   ├── 10parse-root-opts.sh
--More--
```

As you can see the image contains a minimal Linux tree:

```
[root@centos initrd]# ls
bin      dev                 emergency  init       initqueue-finished  initqueue-timeout  mount      pre-trigger
proc  sys      tmp  var
cmdline  dracut-004-303.el6  etc        initqueue  initqueue-settled   lib                pre-pivot  pre-udev
sbin  sysroot  usr
```

The init Script

The init script is executed once the image has been loaded into memory:

```
[root@centos initrd]# more init
#!/bin/sh
#
# Licensed under the GPLv2
#
# Copyright 2008-2009, Red Hat, Inc.
# Harald Hoyer <harald@redhat.com>
# Jeremy Katz <katzj@redhat.com>

wait_for_loginit()
{
    if getarg rdinitdebug; then
    set +x
    exec 0<>/dev/console 1<>/dev/console 2<>/dev/console
    # wait for loginit
    i=0
    while [ $i -lt 10 ]; do
        j=$(jobs)
        [ -z "$j" ] && break
        [ -z "${j##*Running*}" ] || break
        sleep 0.1
        i=$(($i+1))
    done
    [ $i -eq 10 ] && kill %1 >/dev/null 2>&1

        while pidof -x /sbin/loginit >/dev/null 2>&1; do
            for pid in $(pidof -x /sbin/loginit); do
                kill $HARD $pid >/dev/null 2>&1
            done
            HARD="-9"
        done
    set -x
    fi
--More--(7%)
```

The dracut Command

The **dracut** command is used under Red Hat/CentOS to easily create an initramfs image. Dracut's configuration file is **/etc/dracut.conf**. This file can be edited in order to inlcude in the new image those modules that are required. For example, uncomment the **add_drivers** directive and add the usb modules to the line:

/etc/dracut.conf

```
# Sample dracut config file

# Specific list of dracut modules to use
#dracutmodules+=""

# Dracut modules to omit
#omit_dracutmodules+=""

# Dracut modules to add to the default
#add_dracutmodules+=""

# additional kernel modules to the default
add_drivers+="ehci-hcd ohci-hcd usd-storage scsi_mod sd_mod"

# list of kernel filesystem modules to be included in the generic initramfs
#filesystems+=""

# build initrd only to boot current hardware
#hostonly="yes"
```

```
#

# install local /etc/mdadm.conf
mdadmconf="yes"

# install local /etc/lvm/lvm.conf
lvmconf="yes"
```

Now use the dracut command to generate a new image called **usbinitramfs**:

```
# dracut -v usbinitramfs
```

Move the generated image to the /boot directory:

```
[root@centos initrd]# mv usbinitramfs /boot
```

Now edit the **/boot/grub/menu.lst** file and **add** a **new** section as stanza 0 which uses the newly generated image:

```
...
hiddenmenu
title CentOS Linux (usbinitramfs)
    root (hd0,0)
    kernel /vmlinuz-2.6.32-358.23.2.el6.i686 ro root=UUID=c7b1d3e8-6471-4cba-947b-430db974e774 rd_NO_LUKS
KEYBOARDTYPE=pc KEYTABLE=fr LANG=en_US.UTF-8 rd_NO_MD SYSFONT=latarcyrheb-sun16 crashkernel=128M rd_NO_LVM
rd_NO_DM rhgb quiet
    initrd /usbinitramfs
title CentOS (2.6.32-358.el6.i686)
...
```

<note important> **Never** overwrite an existing stanza. In the case of a problem on reboot, you will not be able rollback. </note>

Finally, the dracut command can also be configured directly on the command line:

```
[root@centos initrd]# dracut --help
Usage: /sbin/dracut [OPTION]... <initramfs> <kernel-version>
Creates initial ramdisk images for preloading modules

  -f, --force           Overwrite existing initramfs file.
  -m, --modules [LIST]  Specify a space-separated list of dracut modules to
                         call when building the initramfs. Modules are located
                         in /usr/share/dracut/modules.d.
  -o, --omit [LIST]     Omit a space-separated list of dracut modules.
  -a, --add [LIST]      Add a space-separated list of dracut modules.
  -d, --drivers [LIST]  Specify a space-separated list of kernel modules to
                        exclusively include in the initramfs.
  --add-drivers [LIST]  Specify a space-separated list of kernel
                        modules to add to the initramfs.
  --filesystems [LIST]  Specify a space-separated list of kernel filesystem
                        modules to exclusively include in the generic
                        initramfs.
  -k, --kmoddir [DIR]   Specify the directory, where to look for kernel
                        modules
  --fwdir [DIR]         Specify additional directories, where to look for
                        firmwares, separated by :
  --kernel-only         Only install kernel drivers and firmware files
  --no-kernel           Do not install kernel drivers and firmware files
  --strip               Strip binaries in the initramfs
  --nostrip             Do not strip binaries in the initramfs (default)
  --mdadmconf           Include local /etc/mdadm.conf
  --nomdadmconf         Do not include local /etc/mdadm.conf
  --lvmconf             Include local /etc/lvm/lvm.conf
  --nolvmconf             Do not include local /etc/lvm/lvm.conf
  -h, --help            This message
  --debug               Output debug information of the build process
  -v, --verbose         Verbose output during the build process
  -c, --conf [FILE]     Specify configuration file to use.
                         Default: /etc/dracut.conf
  -l, --local           Local mode. Use modules from the current working
                         directory instead of the system-wide installed in
                         /usr/share/dracut/modules.d.
                         Useful when running dracut from a git checkout.
  -H, --hostonly          Host-Only mode: Install only what is needed for
                         booting the local host instead of a generic host.
  -i, --include [SOURCE] [TARGET]
                        Include the files in the SOURCE directory into the
                         Target directory in the final initramfs.
  -I, --install [LIST]  Install the space separated list of files into the
                         initramfs.
```

<note> Restart the Virtual Machine and log back into it. </note>

Initramfs on Debian Squeeze

To examine the current image, first copy it to /tmp whilst renaming it to **custom.gz** and then uncompress it:

```
root@debian:~# cp /boot/initrd.img-2.6.32-5-686 /tmp/custom.gz
root@debian:~# cd /tmp
root@debian:/tmp# gunzip custom.gz
```

Now extract the cpio archive as follows:

```
root@debian:/tmp# mkdir initrd
root@debian:/tmp# cd initrd
root@debian:/tmp/initrd# cpio -idvB < ../custom
...
```

Now install the **tree** package using **apt-get**:

```
root@debian:/tmp/initrd# apt-get install tree
```

```
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les NOUVEAUX paquets suivants seront installés :
  tree
0 mis à jour, 1 nouvellement installés, 0 à enlever et 4 non mis à jour.
Il est nécessaire de prendre 31,2 ko dans les archives.
Après cette opération, 94,2 ko d'espace disque supplémentaires seront utilisés.
Réception de : 1 http://ftp.fr.debian.org/debian/ squeeze/main tree i386 1.5.3-1 [31,2 kB]
31,2 ko réceptionnés en 0s (40,1 ko/s)
Sélection du paquet tree précédemment désélectionné.
(Lecture de la base de données... 144283 fichiers et répertoires déjà installés.)
Dépaquetage de tree (à partir de .../archives/tree_1.5.3-1_i386.deb) ...
Traitement des actions différées (« triggers ») pour « man-db »...
Paramétrage de tree (1.5.3-1) ...
```

Now use the tree commande to examine the contents of the image:

```
root@debian:/tmp/initrd# tree | more
.
├── bin
│   ├── busybox
│   ├── cat
│   ├── chroot
│   ├── cpio
│   ├── dd
│   ├── dmesg
│   ├── false
│   ├── fstype
│   ├── gunzip
│   ├── gzip
│   ├── halt
│   ├── insmod
│   ├── ipconfig
│   ├── kill
│   ├── ln
│   ├── losetup
│   ├── ls
│   ├── minips
│   ├── mkdir
│   ├── mkfifo
│   ├── mknod
│   ├── mount
│   ├── nfsmount
│   ├── nuke
│   ├── pivot_root
│   ├── poweroff
--More--
```

As you can see the image contains a minimal Linux tree:

```
root@debian:/tmp/initrd# ls
bin  conf  etc  init  lib  sbin  scripts
```

The init Script

The init script is executed once the image has been loaded into memory:

```
root@debian:/tmp/initrd# more init
#!/bin/sh

echo "Loading, please wait..."

[ -d /dev ] || mkdir -m 0755 /dev
[ -d /root ] || mkdir -m 0700 /root
[ -d /sys ] || mkdir /sys
[ -d /proc ] || mkdir /proc
[ -d /tmp ] || mkdir /tmp
mkdir -p /var/lock
mount -t sysfs -o nodev,noexec,nosuid none /sys
mount -t proc -o nodev,noexec,nosuid none /proc

# Note that this only becomes /dev on the real filesystem if udev's scripts
# are used; which they will be, but it's worth pointing out
tmpfs_size="10M"
if [ -e /etc/udev/udev.conf ]; then
    . /etc/udev/udev.conf
fi
if ! mount -t devtmpfs -o mode=0755 none /dev; then
    echo "W: devtmpfs not available, falling back to tmpfs for /dev"
    mount -t tmpfs -o size=$tmpfs_size,mode=0755 udev /dev
    [ -e /dev/console ] || mknod -m 0600 /dev/console c 5 1
    [ -e /dev/null ] || mknod /dev/null c 1 3
fi
mkdir /dev/pts
mount -t devpts -o noexec,nosuid,gid=5,mode=0620 none /dev/pts || true
> /dev/.initramfs-tools
--More--(15%)
```

The mkinitramfs Command

The **mkinitramfs** command is used under Debian to easily create an initramfs image. Mkinitramfs's module configuration file is **/etc/initramfs-tools/modules**. This file can be edited in order to include in the new image those modules that are required. For example, add the usb modules to the file as shown below:

```
# List of modules that you want to include in your initramfs.
# They will be loaded at boot time in the order below.
#
# Syntax:  module_name [args ...]
```

```
#
# You must run update-initramfs(8) to effect this change.
#
# Examples:
#
# raid1
# sd_mod
ehci-hcd
uhci-mod
ohci-mod
usb-storage
scsi-mod
sd-mod
```

Now use the mkinitramfs command to generate a new image called **usbinitramfs.img**:

```
# mkinitramfs -o usbinitramfs.img
```

Move the generated image to the /boot directory:

```
root@debian:/tmp/initrd# mv usbinitramfs.img /boot
```

Now create a file called **/etc/grub.d/09_usbdebian** as follows:

```
#!/bin/sh -e
cat << EOF
menuentry "Debian GNU/Linux, using usbinitramfs" {
set root=(hd0,msdos1)
search --no-floppy --fs-uuid --set a42a1ddd-14bc-4dde-a537-e6c1b984a782
echo    'Chargement de Linux 2.6.32-5-686 ...'
linux   /boot/vmlinuz-2.6.32-5-686 root=UUID=a42a1ddd-14bc-4dde-a537-e6c1b984a782 ro  selinux=1 quiet
echo    'Chargement du disque mémoire initial ...'
initrd  /boot/usbinitramfs.img
}
```

```
EOF
```

Make the file executable:

```
root@debian:/etc/grub.d# chmod +x 09_usbdebian
```

Use the **update-grub** command to include the information from **/etc/grub.d/09_usbdebian** in the **/boot/grub/grub.cfg** file:

```
root@debian:/etc/grub.d# update-grub
Generating grub.cfg ...
Found background image: /usr/share/images/desktop-base/desktop-grub.png
Found linux image: /boot/vmlinuz-2.6.32-5-686
Found initrd image: /boot/initrd.img-2.6.32-5-686
done
```

Check that a stanza has been included in the **boot/grub/grub.cfg** file:

```
...
### END /etc/grub.d/05_debian_theme ###

### BEGIN /etc/grub.d/09_usbdebian ###
menuentry "Debian GNU/Linux, using usbinitramfs" {
set root=(hd0,msdos1)
search --no-floppy --fs-uuid --set a42a1ddd-14bc-4dde-a537-e6c1b984a782
echo    'Chargement de Linux 2.6.32-5-686 ...'
linux   /boot/vmlinuz-2.6.32-5-686 root=UUID=a42a1ddd-14bc-4dde-a537-e6c1b984a782 ro  selinux=1 quiet
echo    'Chargement du disque mémoire initial ...'
initrd  /boot/usbinitramfs.img
}
### END /etc/grub.d/09_usbdebian ###

### BEGIN /etc/grub.d/10_linux ###
...
```

<note> Restart the Virtual Machine and log back into it. </note>

Initramfs on openSUSE

To examine the current image, first copy it to /tmp whilst renaming it to **custom.gz** and then uncompress it:

```
opensuse:~ # cp /boot/initrd /tmp/custom.gz
opensuse:~ # gunzip /tmp/custom.gz
```

Now extract the cpio archive as follows:

```
opensuse:~ # cd /tmp
opensuse:/tmp # mkdir initrd
opensuse:/tmp # cd initrd/
opensuse:/tmp/initrd # cpio -cid -I ../custom
44146 blocks
```

Now install the **tree** package using **zypper**:

```
opensuse:/tmp/initrd # zypper install tree
Loading repository data...
Reading installed packages...
Resolving package dependencies...

The following NEW package is going to be installed:
  tree

1 new package to install.
Overall download size: 33.0 KiB. After the operation, additional 74.0 KiB will
be used.
Continue? [y/n/?] (y): y
Retrieving package tree-1.5.3-7.1.i586 (1/1), 33.0 KiB (74.0 KiB unpacked)
```

```
Retrieving: tree-1.5.3-7.1.i586.rpm [done (86.2 KiB/s)]
Installing: tree-1.5.3-7.1 [done]
```

Now use the tree commande to examine the contents of the image:

```
opensuse:/tmp/initrd # tree | more
.
├── bin
│   ├── awk -> /etc/alternatives/awk
│   ├── bash
│   ├── cat
│   ├── chmod
│   ├── cp
│   ├── date
│   ├── grep
│   ├── ipconfig
│   ├── ipconfig.sh
│   ├── linuxrc
│   ├── ln
│   ├── logger
│   ├── ls
│   ├── mkdir
│   ├── mknod
│   ├── mount
│   ├── mv
│   ├── on_ac_power
│   ├── rm
│   ├── run-init
│   ├── sed
--More--
```

As you can see the image contains a minimal Linux tree:

```
[root@centos initrd]# ls
```

```
opensuse:/tmp/initrd # ls
bin   bootsplash  dev  init  mkinitrd.config  root        sbin  tmp  var
boot  config      etc  lib   proc             run_all.sh  sys   usr
```

The init Script

The init script is executed once the image has been loaded into memory:

```
opensuse:/tmp/initrd # more init
#!/bin/bash

##################################################
# add_module_param $kernelmodule $value
# remembers parameters for the given kernel module
# e.g. add_module_param rtl8193too debug=1
##################################################
add_module_param() {
    echo "options $1 $2" >> /etc/modprobe.d/options.conf
}

##################################################
# load_modules
# loads all kernelmodules that belong to the
# current init module
# this is also done automatically when the
# init module is done
# e.g. load_modules
##################################################
load_modules() {
    local module
    for module in $(eval echo $modules)
    do
```

```
--More--(32%)
```

<note> Passez en revue le contenu du script. </note>

The mkinitrd Command

The **mkinitrd** command is used under openSUSE to easily create an initramfs image. Mkinitrd's configuration file is **/etc/sysconfig/kernel**. This file can be edited in order to include in the new image those modules that are required. For example, edit the **INITRD_MODULES** directive and add the usb modules:

```
...
# (like drivers for scsi-controllers, for lvm or reiserfs)
#
INITRD_MODULES="ahci ata_piix ata_generic thermal processor fan ehci-hcd ohci-hcd usb-storage scsi_mod sd_mod"

## Type:        string(yes)
...
```

Navigate to the **/boot** directory and run the **mkinitrd** command:

```
opensuse:/boot # mkinitrd -v

Kernel image:   /boot/vmlinuz-2.6.37.6-0.5-desktop
Initrd image:   /boot/initrd-2.6.37.6-0.5-desktop
Root device:    /dev/disk/by-id/ata-VBOX_HARDDISK_VBb97f4bdd-e14419fa-part2 (/dev/sda2) (mounted on / as ext3)
Resume device:  /dev/disk/by-id/ata-VBOX_HARDDISK_VBb97f4bdd-e14419fa-part3 (/dev/sda3)
[BLOCK] /dev/sda -> ata_piix
[BLOCK] /dev/sda -> sd_mod
[MODULES]   02-start.sh: ahci ata_piix ata_generic thermal processor fan ehci-hcd ohci-hcd usb-storage scsi_mod
sd_mod
[MODULES]   02-start.sh:
[MODULES]   03-rtc.sh: rtc_cmos
```

```
[MODULES]   03-storage.sh:
[MODULES]   11-block.sh: ata_piix sd_mod
[MODULES]   11-usb.sh: usbcore
[MODULES]   11-usb.sh: ohci_hcd
[MODULES]   11-usb.sh: uhci-hcd
[MODULES]   11-usb.sh: ehci_hcd
[MODULES]   11-usb.sh: usbhid
[MODULES]   'modinfo -k "2.6.37.6-0.5-desktop" -F supported'  returned with an error.
Kernel Modules: ata_generic thermal_sys thermal processor fan
[MOUNT] Root:   /dev/disk/by-id/ata-VBOX_HARDDISK_VBb97f4bdd-e14419fa-part2
Features:       block usb resume.userspace resume.kernel
Bootsplash: openSUSE (800x600)
Shared libs:    /lib/udev/bluetooth.sh is a script
/lib/udev/bluetooth_serial is a script
/lib/udev/collect_lvm is a script
/lib/udev/findkeyboards is a script
/lib/udev/idedma.sh is a script
/lib/udev/iwlwifi-led.sh is a script
/lib/udev/keyboard-force-release.sh is a script
/lib/udev/kpartx_id is a script
/lib/udev/udev-add-printer is a script
/lib/udev/usb_modeswitch is a script
/lib/udev/write_cd_rules is a script
/lib/udev/write_net_rules is a script
/lib/mkinitrd/bin/ipconfig.sh is a script
/sbin/ifup is a script
/lib/mkinitrd/bin/ipconfig.sh is a script
/lib/mkinitrd/bin/linuxrc is a script
/usr/bin/on_ac_power is a script
lib/ld-2.11.3.so lib/libacl.so.1.1.0 lib/libattr.so.1.1.0 lib/libblkid.so.1.1.0 lib/libc-2.11.3.so
lib/libcap.so.2.19 lib/libcom_err.so.2.1 lib/libcrypto.so.1.0.0 lib/libdevmapper.so.1.02 lib/libdl-2.11.3.so
lib/libe2p.so.2.3 lib/libext2fs.so.2.4 lib/libgcrypt.so.11.6.0 lib/libgio-2.0.so.0.2800.0
lib/libglib-2.0.so.0.2800.0 lib/libgmodule-2.0.so.0.2800.0 lib/libgobject-2.0.so.0.2800.0 lib/libgpg-
error.so.0.7.0 lib/libgthread-2.0.so.0.2800.0 lib/libkeyutils-1.3.so lib/liblzo2.so.2.0.0 lib/libm-2.11.3.so
```

```
lib/libmount.so.1.1.0 lib/libncurses.so.5.7 lib/libpcre.so.0.0.1 lib/libpthread-2.11.3.so lib/libreadline.so.6.1
lib/libresolv-2.11.3.so lib/librt-2.11.3.so lib/libselinux.so.1 lib/libsepol.so.1 lib/libssl.so.1.0.0
lib/libudev.so.0.10.0 lib/libutil-2.11.3.so lib/libuuid.so.1.3.0 lib/libz.so.1.2.5 usr/lib/libatasmart.so.4.0.3
usr/lib/libcups.so.2 usr/lib/libdal-0.3.so.0.0.0 usr/lib/libdirect-1.4.so.5.0.0 usr/lib/libdirectfb-1.4.so.5.0.0
usr/lib/libfusion-1.4.so.5.0.0 usr/lib/libgdk_pixbuf-2.0.so.0.2200.1 usr/lib/libgnutls.so.26.14.12
usr/lib/libgpod.so.4.3.1 usr/lib/libgssapi_krb5.so.2.2 usr/lib/libimobiledevice.so.1.0.4
usr/lib/libk5crypto.so.3.1 usr/lib/libkrb5.so.3.3 usr/lib/libkrb5support.so.0.1 usr/lib/libmtp.so.8.3.6
usr/lib/libparted.so.0.0.1 usr/lib/libplist.so.1.1.3 usr/lib/libreiserfs-0.3.so.0.0.0
usr/lib/libsgutils2.so.2.0.0 usr/lib/libsplashy.so.1.0.0 usr/lib/libsplashycnf.so.1.0.0
usr/lib/libsqlite3.so.0.8.6 usr/lib/libtasn1.so.3.1.5 usr/lib/libusb-0.1.so.4.4.4 usr/lib/libusb-1.0.so.0.0.0
usr/lib/libusbmuxd.so.1.0.7 usr/lib/libxml2.so.2.7.8 lib/libnss_dns-2.11.3.so lib/libnss_dns.so.2
lib/libnss_files-2.11.3.so lib/libnss_files.so.2 lib/libgcc_s.so.1
44147 blocks
```

Note the presence of the new image:

```
opensuse:/boot # ls -l | grep initrd
lrwxrwxrwx 1 root root       27 Oct 19 14:13 initrd -> initrd-2.6.37.6-0.5-desktop
-rw-r--r-- 1 root root 10473839 Oct 19 14:13 initrd-2.6.37.6-0.5-desktop
```

<note> Modify the **/boot/grub/menu.lst** file in order to boot using the new initramfs image. </note>

Finally, the mkinitrd command can also be configured directly on the command line:

```
opensuse:/boot # mkinitrd -h




MKINITRD
       Create initial ramdisk images that contain all kernel modules needed in
       the early boot process, before the root file system becomes  available.
       This  usually  includes  SCSI and/or RAID modules, a file system module
       for the root file system, or a  network  interface  driver  module  for
       dhcp.
       mkinitrd [options]


       -f feature list
              Features  to  be enabled when generating initrd.  Available fea-
              tures are: iscsi, md, multipath, lvm, lvm2, ifup

       -k kernel list
              List of kernel  images  for  which  initrd  files  are  created.
              Defaults to all kernels found in /boot.

       -i initrd list
              List  of file names for the initrd; position have match to "ker-
              nel list". Defaults to all kernels found in /boot.

       -l lib_dir
              mkinitrd directory. Defaults to /lib/mkinitrd.

       -b boot_dir
              Boot directory. Defaults to /boot.

       -M map System.map file to use.

       -A     Create a so called "monster initrd" which includes all  features
              and modules possible.

       -B     Do not update bootloader configuration.

       -v     Verbose mode.

       -R     Print release (version).

       -L     Disable logging.
       -h     This help screen.

       -m module list
              Modules  to  include  in  initrd. Defaults to the INITRD_MODULES
              variable in /etc/sysconfig/kernel

       -u DomU module list
              Modules to include in initrd. Defaults to  the  DOMU_INITRD_MOD-
              ULES variable in /etc/sysconfig/kernel.

       -d root_device
              Root  device.  Defaults  to  the device from which / is mounted.
              Overrides the rootdev enviroment variable if set.

       -j device
              Journal device

       -S     Don't include all libata drivers.

       -D interface
              Run dhcp on the specified interface.

       -I interface
              Configure the specified interface statically.

       -a acpi_dsdt
              Attach compiled ACPI DSDT (Differentiated System Description Ta-
              ble)  to initrd. This replaces the DSDT of the BIOS. Defaults to
              the ACPI_DSDT variable in /etc/sysconfig/kernel.

       -s size
              Add splash animation and bootscreen to initrd.

       -V script
```

```
              Vendor specific script to run in linuxrc (deprecated).
```

<note> Restart the Virtual Machine and log back into it. </note>

Kernel Booting Process

The Kernel Booting Process is divided into 6 stages:

Stage	**Description**
Kernel loader loading, setup and configuration	In this step, the bootsect.s file is loaded into the memory by the BIOS. When the bootsect.s file sets up, it loads the rest of the kernel into the memory.
Parameter setup and switch to 32-bit mode	When the kernel has been loaded, the boot.s file sets up a temporary **IDT** and **GDT** and handles the switch to 32-bit mode.
Kernel decompression	The head.s file decompresses the kernel.
Kernel setup	After the kernel is decompressed, the real GDT and IDT are created by the head.s (second file).
Kernel and memory initialisation	In this step, the kernel sets up all memory constraints and virtual memory is completely set up.
Init process creation	In the final step of booting, the init process is created, which switches a Linux computer to different runlevels.

The **init_post()** function then tries to execute one of the following in the order shown:

- /sbin/init
- /etc/init
- /bin/init
- /bin/sh

An error at this stage results in a **Kernel Panic**.

The Init Process

As stated above, the first process launched is **init**. Init's role is to initialise the system. Init:

- mounts the /proc and the /sys filesystems,
- configures the kernel by using the **/etc/sysctl.conf** file,
- activates SELinux,
- updates the system time,
- sets up the text consoles,
- defines the system name,
- detects any USB peripherals,
- sets up RAID and LVM if appropriate,
- implements disk quotas if any,
- mounts the relevant filesystems,
- re-mounts the root filesystem in read/write mode,
- sets up swap space,
- launches syslog, syslog-ng or rsyslog dependant upon which package is installed,
- loads all necessary kernel modules,
- cleans up any temporary files,
- defines system environmental variables such as PATH and RUNLEVEL.

RUNLEVELS

Linux has 8 Runlevels of which 4 are common to Red Hat/CentOS, Debian and openSUSE:

RUNLEVEL	Description
0	System halt
1	Single user mode
6	System reboot
S or s	Single user mode with only the root partition mounted

The other runlevels are defined by each distribution. For Red Hat/CentOS and openSUSE these are:

RUNLEVEL	Description
2	Multi-user mode without NFS
3	Multi-user mode with NFS

RUNLEVEL	Description
4	Not used
5	Multi-user mode with graphical login

For Debian these are:

RUNLEVEL	Description
2	Multi-user mode with NFS
3	Not used
4	Not used
5	Not used

There are also 3 pseudo-runlevels **a**, **b** et **c**. These are used by init to isolate tasks without changing principal runlevels.

The current runlevel can be identified by using the **runlevel** command:

```
[root@centos ~]# runlevel
N 5
```

```
root@debian:~# runlevel
N 2
```

The letter **N** indicates that the system has not changed runlevels since it was booted. The figure **5** indicates that the system is currently in runlevel 5.

To change runlevels, use the **init** or **telinit** commands followed by the destination runlevel.

Option	Description
Q or q	Tells init to re-read its configuration file - /etc/inittab.
-t	Indicates a grace time in seconds between the SIGTERM signal and the SIGKILL signal.

Unix System V Startup Scripts

Debian Squeeze

Inittab

The **/etc/inittab** specifies which services are started in which runlevels:

```
root@debian:~# cat /etc/inittab
# /etc/inittab: init(8) configuration.
# $Id: inittab,v 1.91 2002/01/25 13:35:21 miquels Exp $

# The default runlevel.
id:2:initdefault:

# Boot-time system configuration/initialization script.
# This is run first except when booting in emergency (-b) mode.
si::sysinit:/etc/init.d/rcS

# What to do in single-user mode.
~~:S:wait:/sbin/sulogin

# /etc/init.d executes the S and K scripts upon change
# of runlevel.
#
# Runlevel 0 is halt.
# Runlevel 1 is single-user.
# Runlevels 2-5 are multi-user.
# Runlevel 6 is reboot.

l0:0:wait:/etc/init.d/rc 0
l1:1:wait:/etc/init.d/rc 1
l2:2:wait:/etc/init.d/rc 2
l3:3:wait:/etc/init.d/rc 3
l4:4:wait:/etc/init.d/rc 4
```

```
l5:5:wait:/etc/init.d/rc 5
l6:6:wait:/etc/init.d/rc 6
# Normally not reached, but fallthrough in case of emergency.
z6:6:respawn:/sbin/sulogin

# What to do when CTRL-ALT-DEL is pressed.
ca:12345:ctrlaltdel:/sbin/shutdown -t1 -a -r now

# Action on special keypress (ALT-UpArrow).
#kb::kbrequest:/bin/echo "Keyboard Request--edit /etc/inittab to let this work."

# What to do when the power fails/returns.
pf::powerwait:/etc/init.d/powerfail start
pn::powerfailnow:/etc/init.d/powerfail now
po::powerokwait:/etc/init.d/powerfail stop

# /sbin/getty invocations for the runlevels.
#
# The "id" field MUST be the same as the last
# characters of the device (after "tty").
#
# Format:
#  <id>:<runlevels>:<action>:<process>
#
# Note that on most Debian systems tty7 is used by the X Window System,
# so if you want to add more getty's go ahead but skip tty7 if you run X.
#
1:2345:respawn:/sbin/getty 38400 tty1
2:23:respawn:/sbin/getty 38400 tty2
3:23:respawn:/sbin/getty 38400 tty3
4:23:respawn:/sbin/getty 38400 tty4
5:23:respawn:/sbin/getty 38400 tty5
6:23:respawn:/sbin/getty 38400 tty6
```

```
# Example how to put a getty on a serial line (for a terminal)
#
#T0:23:respawn:/sbin/getty -L ttyS0 9600 vt100
#T1:23:respawn:/sbin/getty -L ttyS1 9600 vt100

# Example how to put a getty on a modem line.
#
#T3:23:respawn:/sbin/mgetty -x0 -s 57600 ttyS3
```

In the above example, each uncommented line contains four fields separated by a colon:

Field	Name	Description
1	ID	A 1 to 4 character unique identifier for the line
2	RUN LEVELS	The runlevels concerned by the line
3	ACTION	The method used to run the command in the 4th field
4	COMMAND	The command to execute

The **action** field can take one of the following directives:

Directive	Description
respawn	The process is automatically re-started if stopped
mingetty	Manages text terminals
once	The command is only executed once
wait	The command is only executed once. Init waits for the process to complete before going to the next line
boot	The command is executed at boot time. The runlevel field is not read
bootwait	The command is executed at boot time. The runlevel field is not read. Init waits for the process to complete before going to the next line
off	Has the same effect as commenting out the line
ondemand	Identical to respawn except that the process is executed in runlevel a,b or c
initdefault	Indicates the default runlevel destination upon boot
sysinit	Indicates the command that is executed before boot and bootwait
powerfail	The command is executed when the system receives the SIGPWR signal from a UPS
powerwait	The command is executed when the system receives the SIGPWR signal from a UPS. Init waits for the process to complete before going to the next line

Directive	Description
powerokwait	The command is executed when the system receives a signal from a UPS that power has been re-established
powerfailnow	The command that is executed when the system receives a notificartion that the UPS' battery is nearly empty
ctrlaltdel	The command that is executed when init receives a SIGINT signal generated by simultaneously hitting the [CTRL] [ALT] [DEL] keys
kbrequest	A command that is executed when a pre-defined combination of keys are pressed

The /etc/init.d directory

This directory contains scripts required to launch each service:

```
root@debian:/etc# cd init.d
root@debian:/etc/init.d# ls
acpid          hwclockfirst.sh      rc.local
alsa-utils     hwclock.sh          rcS
anacron        ifupdown          README
atd        ifupdown-clean      reboot
avahi-daemon       kerneloops          rmnologin
binfmt-support     keyboard-setup      rsyslog
bluetooth      killprocs          saned
bootlogd       lm-sensors          sendsigs
bootlogs       loadcpufreq          single
bootmisc.sh    module-init-tools      skeleton
checkfs.sh     mountall-bootclean.sh  stop-bootlogd
checkroot.sh       mountall.sh          stop-bootlogd-single
console-screen.sh  mountdevsubfs.sh   sudo
console-setup      mountkernfs.sh      udev
cpufrequtils       mountnfs-bootclean.sh  udev-mtab
cron           mountnfs.sh          umountfs
cups           mountoverflowtmp      umountnfs.sh
dbus           mtab.sh          umountroot
exim4          networking          unattended-upgrades
fancontrol     network-manager      urandom
fuse           nfs-common          vboxadd
```

```
gdm3           portmap          vboxadd-service
halt           pppd-dns          vboxadd-x11
hdparm         procps          x11-common
hostname.sh    rc
```

Le script rc.S

The script /etc/init.d/rcS script is executed when the system first boots. This scripts role is to call all S??* scripts in /etc/rcS.d/ in numerical/alphabetical order :

```
root@debian:/etc/init.d# cat /etc/init.d/rcS
#! /bin/sh
#
# rcS
#
# Call all S??* scripts in /etc/rcS.d/ in numerical/alphabetical order
#

exec /etc/init.d/rc S
```

The /etc/rcX.d Directories

The directories numbered from **/etc/rc0.d** through **/etc/rc6.d** contains soft links pointing to scripts in /etc/init.d. For example:

```
root@debian:/etc/init.d# for rep in /etc/rc[2345].d; do echo "dans $rep : "; ls $rep/S*; done
dans /etc/rc2.d :
/etc/rc2.d/S01vboxadd          /etc/rc2.d/S18kerneloops
/etc/rc2.d/S02vboxadd-service  /etc/rc2.d/S18loadcpufreq
/etc/rc2.d/S14portmap          /etc/rc2.d/S19avahi-daemon
/etc/rc2.d/S15nfs-common       /etc/rc2.d/S19bluetooth
/etc/rc2.d/S17binfmt-support   /etc/rc2.d/S19cpufrequtils
```

```
/etc/rc2.d/S17fancontrol       /etc/rc2.d/S19network-manager
/etc/rc2.d/S17rsyslog          /etc/rc2.d/S20cups
/etc/rc2.d/S17sudo         /etc/rc2.d/S20gdm3
/etc/rc2.d/S18acpid        /etc/rc2.d/S20saned
/etc/rc2.d/S18anacron          /etc/rc2.d/S21bootlogs
/etc/rc2.d/S18atd          /etc/rc2.d/S22rc.local
/etc/rc2.d/S18cron         /etc/rc2.d/S22rmnologin
/etc/rc2.d/S18dbus         /etc/rc2.d/S22stop-bootlogd
/etc/rc2.d/S18exim4
dans /etc/rc3.d :
/etc/rc3.d/S01vboxadd          /etc/rc3.d/S18kerneloops
/etc/rc3.d/S02vboxadd-service  /etc/rc3.d/S18loadcpufreq
/etc/rc3.d/S14portmap          /etc/rc3.d/S19avahi-daemon
/etc/rc3.d/S15nfs-common       /etc/rc3.d/S19bluetooth
/etc/rc3.d/S17binfmt-support   /etc/rc3.d/S19cpufrequtils
/etc/rc3.d/S17fancontrol       /etc/rc3.d/S19network-manager
/etc/rc3.d/S17rsyslog          /etc/rc3.d/S20cups
/etc/rc3.d/S17sudo         /etc/rc3.d/S20gdm3
/etc/rc3.d/S18acpid        /etc/rc3.d/S20saned
/etc/rc3.d/S18anacron          /etc/rc3.d/S21bootlogs
/etc/rc3.d/S18atd          /etc/rc3.d/S22rc.local
/etc/rc3.d/S18cron         /etc/rc3.d/S22rmnologin
/etc/rc3.d/S18dbus         /etc/rc3.d/S22stop-bootlogd
/etc/rc3.d/S18exim4
dans /etc/rc4.d :
/etc/rc4.d/S01vboxadd          /etc/rc4.d/S18kerneloops
/etc/rc4.d/S02vboxadd-service  /etc/rc4.d/S18loadcpufreq
/etc/rc4.d/S14portmap          /etc/rc4.d/S19avahi-daemon
/etc/rc4.d/S15nfs-common       /etc/rc4.d/S19bluetooth
/etc/rc4.d/S17binfmt-support   /etc/rc4.d/S19cpufrequtils
/etc/rc4.d/S17fancontrol       /etc/rc4.d/S19network-manager
/etc/rc4.d/S17rsyslog          /etc/rc4.d/S20cups
/etc/rc4.d/S17sudo         /etc/rc4.d/S20gdm3
/etc/rc4.d/S18acpid        /etc/rc4.d/S20saned
```

```
/etc/rc4.d/S18anacron          /etc/rc4.d/S21bootlogs
/etc/rc4.d/S18atd          /etc/rc4.d/S22rc.local
/etc/rc4.d/S18cron         /etc/rc4.d/S22rmnologin
/etc/rc4.d/S18dbus         /etc/rc4.d/S22stop-bootlogd
/etc/rc4.d/S18exim4
dans /etc/rc5.d :
/etc/rc5.d/S01vboxadd          /etc/rc5.d/S18kerneloops
/etc/rc5.d/S02vboxadd-service  /etc/rc5.d/S18loadcpufreq
/etc/rc5.d/S14portmap          /etc/rc5.d/S19avahi-daemon
/etc/rc5.d/S15nfs-common       /etc/rc5.d/S19bluetooth
/etc/rc5.d/S17binfmt-support   /etc/rc5.d/S19cpufrequtils
/etc/rc5.d/S17fancontrol       /etc/rc5.d/S19network-manager
/etc/rc5.d/S17rsyslog          /etc/rc5.d/S20cups
/etc/rc5.d/S17sudo         /etc/rc5.d/S20gdm3
/etc/rc5.d/S18acpid        /etc/rc5.d/S20saned
/etc/rc5.d/S18anacron          /etc/rc5.d/S21bootlogs
/etc/rc5.d/S18atd          /etc/rc5.d/S22rc.local
/etc/rc5.d/S18cron         /etc/rc5.d/S22rmnologin
/etc/rc5.d/S18dbus         /etc/rc5.d/S22stop-bootlogd
/etc/rc5.d/S18exim4
```

<note important> Each directory corresponds to a runlevel. The letter **S** indicates that the **rc** executable should execute the script concerned by passing it a **start** switch. The number following the letter S indicates the order in which rc will execute the scripts. If two links have the same number, they are executed in alphabetical order.The soft link **S22rc.local** points to a script called **rc.local** which root can edit in order to launch any program at boot time. </note>

Certain links start with the letter **K**:

```
root@debian:/etc/init.d# for rep in /etc/rc[016].d; do echo "dans $rep :"; ls $rep/K*; done
dans /etc/rc0.d :
/etc/rc0.d/K01alsa-utils       /etc/rc0.d/K02avahi-daemon
/etc/rc0.d/K01anacron          /etc/rc0.d/K02vboxadd
/etc/rc0.d/K01atd          /etc/rc0.d/K03sendsigs
/etc/rc0.d/K01bluetooth        /etc/rc0.d/K04rsyslog
```

```
/etc/rc0.d/K01exim4        /etc/rc0.d/K05umountnfs.sh
/etc/rc0.d/K01fuse         /etc/rc0.d/K06nfs-common
/etc/rc0.d/K01gdm3         /etc/rc0.d/K06portmap
/etc/rc0.d/K01kerneloops       /etc/rc0.d/K07hwclock.sh
/etc/rc0.d/K01network-manager      /etc/rc0.d/K07networking
/etc/rc0.d/K01saned        /etc/rc0.d/K08ifupdown
/etc/rc0.d/K01unattended-upgrades  /etc/rc0.d/K09umountfs
/etc/rc0.d/K01urandom          /etc/rc0.d/K10umountroot
/etc/rc0.d/K01vboxadd-service      /etc/rc0.d/K11halt
dans /etc/rc1.d :
/etc/rc1.d/K01alsa-utils  /etc/rc1.d/K01network-manager
/etc/rc1.d/K01anacron     /etc/rc1.d/K01saned
/etc/rc1.d/K01atd     /etc/rc1.d/K01vboxadd-service
/etc/rc1.d/K01bluetooth   /etc/rc1.d/K02avahi-daemon
/etc/rc1.d/K01cups    /etc/rc1.d/K02vboxadd
/etc/rc1.d/K01exim4   /etc/rc1.d/K04rsyslog
/etc/rc1.d/K01gdm3    /etc/rc1.d/K06nfs-common
/etc/rc1.d/K01kerneloops  /etc/rc1.d/K06portmap
dans /etc/rc6.d :
/etc/rc6.d/K01alsa-utils       /etc/rc6.d/K02avahi-daemon
/etc/rc6.d/K01anacron          /etc/rc6.d/K02vboxadd
/etc/rc6.d/K01atd          /etc/rc6.d/K03sendsigs
/etc/rc6.d/K01bluetooth        /etc/rc6.d/K04rsyslog
/etc/rc6.d/K01exim4        /etc/rc6.d/K05umountnfs.sh
/etc/rc6.d/K01fuse         /etc/rc6.d/K06nfs-common
/etc/rc6.d/K01gdm3         /etc/rc6.d/K06portmap
/etc/rc6.d/K01kerneloops       /etc/rc6.d/K07hwclock.sh
/etc/rc6.d/K01network-manager      /etc/rc6.d/K07networking
/etc/rc6.d/K01saned        /etc/rc6.d/K08ifupdown
/etc/rc6.d/K01unattended-upgrades  /etc/rc6.d/K09umountfs
/etc/rc6.d/K01urandom          /etc/rc6.d/K10umountroot
/etc/rc6.d/K01vboxadd-service      /etc/rc6.d/K11reboot
```

In this case the principal is the same, however this time, the **rc** executable should execute the script concerned by passing it a **stop** switch.

The update-rc.d Command

The **update-rc.d** is used to manage the links in the /etc/rcX.d directories:

```
update-rc.d <service> start <start priority> <start runlevels> . stop <stop priority> <stop runlevels> .
```

For example the following command creates the **S** (start) links in runlevels 2, 3, 4 and 5 for ssh with a priority of 20 whilst creating the **K** (stop) links in runlevels 0, 1 and 6 for ssh with a priority of 20:

```
# update-rc.d ssh start 20 2 3 4 5 . stop 20 0 1 6 . [Entrée]
```

<note important> **update-rc.d** can also be launched with the **default** argument. The **default** setup is exactly he same as command line above. </note>

To delete links for ssh you would use the following command:

```
# update-rc.d -f ssh remove [Entrée]
```

<note important> This command does not delete the script situated in /etc/init.d. </note>

update-rc.d's command line switches are as follows:

```
root@debian:/etc/init.d# update-rc.d --help
update-rc.d: using dependency based boot sequencing
update-rc.d: error: --help
usage: update-rc.d [-n] [-f] <basename> remove
       update-rc.d [-n] <basename> defaults [NN | SS KK]
       update-rc.d [-n] <basename> start|stop NN runlvl [runlvl] [...] .
       update-rc.d [-n] <basename> disable|enable [S|2|3|4|5]
        -n: not really
        -f: force

The disable|enable API is not stable and might change in the future.
```

The chkconfig Command

To get an overall picture of which services are running in which runlevels, you need to install and use the **chkconfig** command:

```
root@debian:/etc/init.d# apt-get install chkconfig
```

```
root@debian:/etc/init.d# chkconfig --list
acpid                     0:off  1:off  2:on   3:on   4:on   5:on   6:off
alsa-utils                0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
anacron                   0:off  1:off  2:on   3:on   4:on   5:on   6:off
atd                       0:off  1:off  2:on   3:on   4:on   5:on   6:off
avahi-daemon              0:off  1:off  2:on   3:on   4:on   5:on   6:off
binfmt-support            0:off  1:off  2:on   3:on   4:on   5:on   6:off
bluetooth                 0:off  1:off  2:on   3:on   4:on   5:on   6:off
bootlogd                  0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
bootlogs                  0:off  1:on   2:on   3:on   4:on   5:on   6:off
bootmisc.sh               0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
checkfs.sh                0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
checkroot.sh              0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
console-screen.sh         0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
console-setup             0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
cpufrequtils              0:off  1:off  2:on   3:on   4:on   5:on   6:off
cron                      0:off  1:off  2:on   3:on   4:on   5:on   6:off
cups                      0:off  1:off  2:on   3:on   4:on   5:on   6:off
dbus                      0:off  1:off  2:on   3:on   4:on   5:on   6:off
exim4                     0:off  1:off  2:on   3:on   4:on   5:on   6:off
fancontrol                0:off  1:off  2:on   3:on   4:on   5:on   6:off
fuse                      0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
gdm3                      0:off  1:off  2:on   3:on   4:on   5:on   6:off
hdparm                    0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
hostname.sh               0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
hwclock.sh                0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
hwclockfirst.sh           0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
```

```
ifupdown                  0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
ifupdown-clean            0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
kerneloops                0:off  1:off  2:on   3:on   4:on   5:on   6:off
keyboard-setup            0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
killprocs                 0:off  1:on   2:off  3:off  4:off  5:off  6:off
lm-sensors                0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
loadcpufreq               0:off  1:off  2:on   3:on   4:on   5:on   6:off
module-init-tools         0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
mountall-bootclean.sh     0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
mountall.sh               0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
mountdevsubfs.sh          0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
mountkernfs.sh            0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
mountnfs-bootclean.sh     0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
mountnfs.sh               0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
mountoverflowtmp          0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
mtab.sh                   0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
network-manager           0:off  1:off  2:on   3:on   4:on   5:on   6:off
networking                0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
nfs-common                0:off  1:off  2:on   3:on   4:on   5:on   6:off  S:on
portmap                   0:off  1:off  2:on   3:on   4:on   5:on   6:off  S:on
pppd-dns                  0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
procps                    0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
rc.local                  0:off  1:off  2:on   3:on   4:on   5:on   6:off
rcS                       0:off  1:off  2:off  3:off  4:off  5:off  6:off
rmnologin                 0:off  1:off  2:on   3:on   4:on   5:on   6:off
rsyslog                   0:off  1:off  2:on   3:on   4:on   5:on   6:off
saned                     0:off  1:off  2:on   3:on   4:on   5:on   6:off
sendsigs                  0:off  1:off  2:off  3:off  4:off  5:off  6:off
stop-bootlogd             0:off  1:off  2:on   3:on   4:on   5:on   6:off
stop-bootlogd-single      0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
sudo                      0:off  1:off  2:on   3:on   4:on   5:on   6:off
udev                      0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
udev-mtab                 0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
umountfs                  0:off  1:off  2:off  3:off  4:off  5:off  6:off
```

```
umountnfs.sh              0:off  1:off  2:off  3:off  4:off  5:off  6:off
umountroot                0:off  1:off  2:off  3:off  4:off  5:off  6:off
unattended-upgrades       0:off  1:off  2:off  3:off  4:off  5:off  6:off
urandom                   0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
vboxadd                   0:off  1:off  2:on   3:on   4:on   5:on   6:off
vboxadd-service           0:off  1:off  2:on   3:on   4:on   5:on   6:off
vboxadd-x11               0:off  1:off  2:off  3:off  4:off  5:off  6:off
x11-common                0:off  1:off  2:off  3:off  4:off  5:off  6:off  S:on
```

chkconfig's command line switches are as follows:

```
root@debian:/tmp# chkconfig --help
usage:
        chkconfig -A|--allservices              (together with -l: show all services)
        chkconfig -t|--terse [names]            (shows the links)
        chkconfig -e|--edit  [names]            (configure services)
        chkconfig -s|--set   [name state]...    (configure services)
        chkconfig -l|--list [--deps] [names]    (shows the links)
        chkconfig -c|--check name [state]       (check state)
        chkconfig -a|--add   [names]            (runs insserv)
        chkconfig -d|--del   [names]            (runs insserv -r)
        chkconfig -h|--help                     (print usage)
        chkconfig -f|--force ...                (call insserv with -f)

        chkconfig [name]           same as chkconfig -t
        chkconfig name state...    same as chkconfig -s name state
```

openSUSE

Under openSUSE the service scripts can be found in /etc/rc.d which is a symbolic link pointing to /etc/init.d:

```
opensuse:/etc # cd /etc/rc.d
```

```
opensuse:/etc/rc.d # ls
.depend.boot         boot.ipconfig      haveged           rc5.d
.depend.halt         boot.klog          inputattach       rc6.d
.depend.start        boot.ldconfig      jexec             rcS.d
.depend.stop         boot.loadmodules   joystick          reboot
SuSEfirewall2_init   boot.local         kbd               rpcbind
SuSEfirewall2_setup  boot.localfs       kexec             rpmconfigcheck
aaeventd             boot.localnet      lirc              rsyncd
acpid                boot.lvm           mdadmd            setserial
after.local          boot.md            microcode.ctl     single
alsasound            boot.multipath     multipathd        skeleton
atd                  boot.proc          network           skeleton.compat
auditd               boot.rootfsck      network-remotefs  smartd
autofs               boot.startpreload  nfs               smb
autoyast             boot.swap          nmb               smolt
avahi-daemon         boot.sysctl        nscd              smpppd
avahi-dnsconfd       boot.udev          ntp               splash
before.local         boot.udev_retry    openvpn           splash_early
bluez-coldplug       cifs               pm-profiler       sshd
boot                 cpufreq            postfix           stoppreload
boot.apparmor        cron               powerd            syslog
boot.cgroup          cups               powerfail         vboxadd
boot.cleanup         dbus               random            vboxadd-service
boot.clock           dnsmasq            raw               vboxadd-x11
boot.crypto          earlysyslog        rc                xdm
boot.crypto-early    earlyxdm           rc0.d             xfs
boot.cycle           fbset              rc1.d             xinetd
boot.d               gpm                rc2.d             ypbind
boot.device-mapper   halt               rc3.d
boot.dmraid          halt.local         rc4.d
```

In that directory can be found:

- the service scripts,

- the **boot.d** directory containing scripts that are called by the **boot** script. The boot replaces the rc.S script under Debian,
- the **boot.local** which is similar to DOS' autoexec.bat,
- the **boot.setup** script which is called when moving from runlevel 1 to any higher runlevel. This script defines such things as the keyboard and text terminals.

Upstart Startup Scripts

Red Hat/CentOS 6

When using upstart scripts, the /etc/inittab contains only the **initdefault** directive:

```
[root@centos ~]# cat /etc/inittab
# inittab is only used by upstart for the default runlevel.
#
# ADDING OTHER CONFIGURATION HERE WILL HAVE NO EFFECT ON YOUR SYSTEM.
#
# System initialization is started by /etc/init/rcS.conf
#
# Individual runlevels are started by /etc/init/rc.conf
#
# Ctrl-Alt-Delete is handled by /etc/init/control-alt-delete.conf
#
# Terminal gettys are handled by /etc/init/tty.conf and /etc/init/serial.conf,
# with configuration in /etc/sysconfig/init.
#
# For information on how to write upstart event handlers, or how
# upstart works, see init(5), init(8), and initctl(8).
#
# Default runlevel. The runlevels used are:
#   0 - halt (Do NOT set initdefault to this)
#   1 - Single user mode
#   2 - Multiuser, without NFS (The same as 3, if you do not have networking)
#   3 - Full multiuser mode
#   4 - unused
#   5 - X11
#   6 - reboot (Do NOT set initdefault to this)
#
id:5:initdefault:
```

All other configuration is handled by scripts found in the **/etc/init/** directory:

```
[root@centos ~]# cd /etc/init
[root@centos init]# ls
control-alt-delete.conf  plymouth-shutdown.conf  rc.conf            readahead-collector.conf         serial.conf
tty.conf
init-system-dbus.conf    prefdm.conf             rcS.conf           readahead.conf                   splash-
manager.conf
kexec-disable.conf       quit-plymouth.conf      rcS-sulogin.conf   readahead-disable-services.conf  start-
ttys.conf
```

System Initialisation

System Initialisation is configured by the **/etc/init/rcS.conf** file:

```
[root@centos init]# cat /etc/init/rcS.conf
# rcS - runlevel compatibility
#
# This task runs the old sysv-rc startup scripts.

start on startup

stop on runlevel

task
```

```
# Note: there can be no previous runlevel here, if we have one it's bad
# information (we enter rc1 not rcS for maintenance).  Run /etc/rc.d/rc
# without information so that it defaults to previous=N runlevel=S.
console output
exec /etc/rc.d/rc.sysinit
post-stop script
    if [ "$UPSTART_EVENTS" = "startup" ]; then
        [ -f /etc/inittab ] && runlevel=$(/bin/awk -F ':' '$3 == "initdefault" && $1 !~ "^#" { print $2 }'
/etc/inittab)
        [ -z "$runlevel" ] && runlevel="3"
        for t in $(cat /proc/cmdline); do
            case $t in
                -s|single|S|s) runlevel="S" ;;
                [1-9])       runlevel="$t" ;;
            esac
        done
        exec telinit $runlevel
    fi
end script
```

Runlevels

Runlevels are configured by the **/etc/init/rc.conf** file:

```
[root@centos init]# cat /etc/init/rc.conf
# rc - System V runlevel compatibility
#
# This task runs the old sysv-rc runlevel scripts.  It
# is usually started by the telinit compatibility wrapper.

start on runlevel [0123456]

stop on runlevel [!$RUNLEVEL]
```

```
task

export RUNLEVEL
console output
exec /etc/rc.d/rc $RUNLEVEL
```

[CTL]-[ALT]-[DEL]

The command that is executed when init receives a SIGINT signal generated by simultaneously pressing the [CTRL] [ALT] [DEL] keys is configured in the **/etc/init/control-alt-delete.conf** file:

```
[root@centos init]# cat /etc/init/control-alt-delete.conf
# control-alt-delete - emergency keypress handling
#
# This task is run whenever the Control-Alt-Delete key combination is
# pressed.  Usually used to shut down the machine.

start on control-alt-delete

exec /sbin/shutdown -r now "Control-Alt-Delete pressed"
```

mingetty

Text terminal configuration can be found in **/etc/init/tty.conf**, **/etc/init/serial.conf** and **/etc/sysconfig/init**:

```
[root@centos init]# cat /etc/init/tty.conf
# tty - getty
#
# This service maintains a getty on the sepcified device.

stop on runlevel [016]
```

```
respawn
instance $TTY
exec /sbin/mingetty $TTY
[root@centos rc.d]# cat /etc/init/serial.conf
# Automatically start a configured serial console
#
# How this works:
#
# On boot, a udev helper examines /dev/console. If a serial console is the
# primary console (last console on the commandline in grub),  the event
# 'fedora.serial-console-available <port name> <speed>' is emitted, which
# triggers this script. It waits for the runlevel to finish, ensures
# the proper port is in /etc/securetty, and starts the getty.
#
# If your serial console is not the primary console, or you want a getty
# on serial even if it's not the console, create your own event by copying
# /etc/init/tty.conf, and changing the getty line in that file.

start on fedora.serial-console-available DEV=* and stopped rc RUNLEVEL=[2345]
stop on runlevel [016]

instance $DEV
respawn
pre-start exec /sbin/securetty $DEV
exec /sbin/agetty /dev/$DEV $SPEED vt100-nav
```

```
[root@centos init]# cat /etc/sysconfig/init
# color => new RH6.0 bootup
# verbose => old-style bootup
# anything else => new style bootup without ANSI colors or positioning
BOOTUP=color
# column to start "[  OK  ]" label in
RES_COL=60
# terminal sequence to move to that column. You could change this
# to something like "tput hpa ${RES_COL}" if your terminal supports it
MOVE_TO_COL="echo -en \\033[${RES_COL}G"
# terminal sequence to set color to a 'success' color (currently: green)
SETCOLOR_SUCCESS="echo -en \\033[0;32m"
# terminal sequence to set color to a 'failure' color (currently: red)
SETCOLOR_FAILURE="echo -en \\033[0;31m"
# terminal sequence to set color to a 'warning' color (currently: yellow)
SETCOLOR_WARNING="echo -en \\033[0;33m"
# terminal sequence to reset to the default color.
SETCOLOR_NORMAL="echo -en \\033[0;39m"
# Set to anything other than 'no' to allow hotkey interactive startup...
PROMPT=yes
# Set to 'yes' to allow probing for devices with swap signatures
AUTOSWAP=no
# What ttys should gettys be started on?
ACTIVE_CONSOLES=/dev/tty[1-6]
# Set to '/sbin/sulogin' to prompt for password on single-user mode
# Set to '/sbin/sushell' otherwise
SINGLE=/sbin/sushell
```

rc.sysinit

As seen in the **/etc/init/rcS.d** file the **rc.sysinit** script is executed when the system is booted:

```
[root@redhat ~]# cd /etc/rc.d
[root@redhat rc.d]# ls
init.d  rc0.d  rc2.d  rc4.d  rc6.d      rc.sysinit
rc      rc1.d  rc3.d  rc5.d  rc.local
```

The /etc/rc.d/init.d Directory

Service scripts are stored in **/etc/rc.d/init.d**:

```
[root@centos rc.d]# ls init.d/*
init.d/abrtd         init.d/cpuspeed   init.d/iptables       init.d/netfs           init.d/portreserve
init.d/sandbox    init.d/vboxadd
init.d/acpid         init.d/crond      init.d/irqbalance     init.d/network         init.d/postfix
init.d/saslauthd  init.d/vboxadd-service
init.d/atd           init.d/cups       init.d/jexec          init.d/NetworkManager  init.d/psacct
init.d/single     init.d/vboxadd-x11
init.d/auditd        init.d/dnsmasq    init.d/kdump          init.d/nfs             init.d/rdisc
init.d/smartd     init.d/wpa_supplicant
init.d/autofs        init.d/firstboot  init.d/killall        init.d/nfslock         init.d/restorecond
init.d/snmpd      init.d/ypbind
init.d/avahi-daemon  init.d/functions  init.d/lvm2-monitor   init.d/nscd            init.d/rpcbind
init.d/snmptrapd
init.d/bluetooth     init.d/haldaemon  init.d/mdmonitor      init.d/nslcd           init.d/rpcgssd
init.d/sshd
init.d/certmonger    init.d/halt       init.d/messagebus     init.d/ntpd            init.d/rpcidmapd
init.d/sssd
init.d/cgconfig      init.d/httpd      init.d/microcode_ctl  init.d/ntpdate         init.d/rpcsvcgssd
init.d/sysstat
init.d/cgred         init.d/ip6tables  init.d/netconsole     init.d/oddjobd         init.d/rsyslog
init.d/udev-post
```

The /etc/rc.d/rcX.d Directories

The directories numbered from **/etc/rc.d/rc0.d** through **/etc/rc.d/rc6.d** contains soft links pointing to scripts in /etc/rc.d/init.d. For example:

```
[root@centos rc.d]# for rep in rc[345].d; do echo "dans $rep :"; ls $rep/S*; done
dans rc3.d :
rc3.d/S00microcode_ctl  rc3.d/S10network      rc3.d/S13irqbalance      rc3.d/S24avahi-daemon  rc3.d/S25netfs
rc3.d/S30vboxadd          rc3.d/S80postfix
rc3.d/S01sysstat        rc3.d/S11auditd       rc3.d/S13rpcbind         rc3.d/S24nfslock       rc3.d/S26acpid
```

```
rc3.d/S30vboxadd-x11      rc3.d/S82abrtd
rc3.d/S02lvm2-monitor   rc3.d/S11portreserve  rc3.d/S15mdmonitor       rc3.d/S24rpcgssd       rc3.d/S26haldaemon
rc3.d/S35vboxadd-service  rc3.d/S90crond
rc3.d/S08ip6tables      rc3.d/S12rsyslog      rc3.d/S22messagebus      rc3.d/S24rpcidmapd     rc3.d/S26udev-post
rc3.d/S50bluetooth        rc3.d/S95atd
rc3.d/S08iptables       rc3.d/S13cpuspeed     rc3.d/S23NetworkManager  rc3.d/S25cups          rc3.d/S28autofs
rc3.d/S55sshd             rc3.d/S99local
dans rc4.d :
rc4.d/S00microcode_ctl  rc4.d/S10network      rc4.d/S13irqbalance      rc4.d/S24avahi-daemon  rc4.d/S25netfs
rc4.d/S30vboxadd          rc4.d/S90crond
rc4.d/S01sysstat        rc4.d/S11auditd       rc4.d/S13rpcbind         rc4.d/S24nfslock       rc4.d/S26acpid
rc4.d/S35vboxadd-service  rc4.d/S95atd
rc4.d/S02lvm2-monitor   rc4.d/S11portreserve  rc4.d/S15mdmonitor       rc4.d/S24rpcgssd       rc4.d/S26haldaemon
rc4.d/S50bluetooth        rc4.d/S99local
rc4.d/S08ip6tables      rc4.d/S12rsyslog      rc4.d/S22messagebus      rc4.d/S24rpcidmapd     rc4.d/S26udev-post
rc4.d/S55sshd
rc4.d/S08iptables       rc4.d/S13cpuspeed     rc4.d/S23NetworkManager  rc4.d/S25cups          rc4.d/S28autofs
rc4.d/S80postfix
dans rc5.d :
rc5.d/S00microcode_ctl  rc5.d/S10network      rc5.d/S13irqbalance      rc5.d/S24avahi-daemon  rc5.d/S25netfs
rc5.d/S30vboxadd          rc5.d/S80postfix
rc5.d/S01sysstat        rc5.d/S11auditd       rc5.d/S13rpcbind         rc5.d/S24nfslock       rc5.d/S26acpid
rc5.d/S30vboxadd-x11      rc5.d/S82abrtd
rc5.d/S02lvm2-monitor   rc5.d/S11portreserve  rc5.d/S15mdmonitor       rc5.d/S24rpcgssd       rc5.d/S26haldaemon
rc5.d/S35vboxadd-service  rc5.d/S90crond
rc5.d/S08ip6tables      rc5.d/S12rsyslog      rc5.d/S22messagebus      rc5.d/S24rpcidmapd     rc5.d/S26udev-post
rc5.d/S50bluetooth        rc5.d/S95atd
rc5.d/S08iptables       rc5.d/S13cpuspeed     rc5.d/S23NetworkManager  rc5.d/S25cups          rc5.d/S28autofs
rc5.d/S55sshd             rc5.d/S99local
```

<note important> Each directory corresponds to a runlevel. The letter **S** indicates that the **rc** executable should execute the script concerned by passing it a **start** switch. The number following the letter S indicates the order in which rc will execute the scripts. If two links have the same number, they are executed in alphabetical order.The soft link **S99local** points to a script called **rc.local** which root can edit in order to launch any program at boot time. </note>

Certain links start with the letter **K**:

```
[root@centos rc.d]# for rep in rc[345].d; do echo "dans $rep :"; ls $rep/K*; done
dans rc3.d :
rc3.d/K01certmonger  rc3.d/K10saslauthd   rc3.d/K50snmpd       rc3.d/K73ypbind   rc3.d/K80kdump
rc3.d/K87restorecond  rc3.d/K95firstboot
rc3.d/K01smartd      rc3.d/K15httpd       rc3.d/K50snmptrapd   rc3.d/K74nscd     rc3.d/K80sssd
rc3.d/K88nslcd
rc3.d/K02oddjobd     rc3.d/K50dnsmasq     rc3.d/K60nfs         rc3.d/K74ntpd     rc3.d/K84wpa_supplicant
rc3.d/K89rdisc
rc3.d/K10psacct      rc3.d/K50netconsole  rc3.d/K69rpcsvcgssd  rc3.d/K75ntpdate  rc3.d/K86cgred
rc3.d/K95cgconfig
dans rc4.d :
rc4.d/K01certmonger  rc4.d/K10saslauthd  rc4.d/K50netconsole  rc4.d/K69rpcsvcgssd   rc4.d/K74ntpd
rc4.d/K84wpa_supplicant  rc4.d/K89rdisc
rc4.d/K01smartd      rc4.d/K15httpd      rc4.d/K50snmpd       rc4.d/K70vboxadd-x11  rc4.d/K75ntpdate
rc4.d/K86cgred           rc4.d/K95cgconfig
rc4.d/K02oddjobd     rc4.d/K16abrtd      rc4.d/K50snmptrapd   rc4.d/K73ypbind       rc4.d/K80kdump
rc4.d/K87restorecond     rc4.d/K95firstboot
rc4.d/K10psacct      rc4.d/K50dnsmasq    rc4.d/K60nfs         rc4.d/K74nscd         rc4.d/K80sssd
rc4.d/K88nslcd
dans rc5.d :
rc5.d/K01certmonger  rc5.d/K10saslauthd   rc5.d/K50snmpd       rc5.d/K73ypbind   rc5.d/K80kdump
rc5.d/K87restorecond  rc5.d/K95firstboot
rc5.d/K01smartd      rc5.d/K15httpd       rc5.d/K50snmptrapd   rc5.d/K74nscd     rc5.d/K80sssd
rc5.d/K88nslcd
rc5.d/K02oddjobd     rc5.d/K50dnsmasq     rc5.d/K60nfs         rc5.d/K74ntpd     rc5.d/K84wpa_supplicant
rc5.d/K89rdisc
rc5.d/K10psacct      rc5.d/K50netconsole  rc5.d/K69rpcsvcgssd  rc5.d/K75ntpdate  rc5.d/K86cgred
rc5.d/K95cgconfig
```

In this case the principal is the same, however this time, the **rc** executable should execute the script concerned by passing it a **stop** switch.

La commande chkconfig

To get an overall picture of which services are running in which runlevels, you need to use the **chkconfig** command:

```
[root@centos rc.d]# chkconfig --list
NetworkManager  0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
abrtd           0:arrêt    1:arrêt    2:arrêt    3:marche    4:arrêt    5:marche    6:arrêt
acpid           0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
atd             0:arrêt    1:arrêt    2:arrêt    3:marche    4:marche    5:marche    6:arrêt
auditd          0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
autofs          0:arrêt    1:arrêt    2:arrêt    3:marche    4:marche    5:marche    6:arrêt
avahi-daemon    0:arrêt    1:arrêt    2:arrêt    3:marche    4:marche    5:marche    6:arrêt
bluetooth       0:arrêt    1:arrêt    2:arrêt    3:marche    4:marche    5:marche    6:arrêt
certmonger      0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
cgconfig        0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
cgred           0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
cpuspeed        0:arrêt    1:marche    2:marche    3:marche    4:marche    5:marche    6:arrêt
crond           0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
cups            0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
dnsmasq         0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
firstboot       0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
haldaemon       0:arrêt    1:arrêt    2:arrêt    3:marche    4:marche    5:marche    6:arrêt
httpd           0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
ip6tables       0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
iptables        0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
irqbalance      0:arrêt    1:arrêt    2:arrêt    3:marche    4:marche    5:marche    6:arrêt
kdump           0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
lvm2-monitor    0:arrêt    1:marche    2:marche    3:marche    4:marche    5:marche    6:arrêt
mdmonitor       0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
messagebus      0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
microcode_ctl   0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
netconsole      0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
netfs           0:arrêt    1:arrêt    2:arrêt    3:marche    4:marche    5:marche    6:arrêt
```

```
network         0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
nfs             0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
nfslock         0:arrêt    1:arrêt    2:arrêt    3:marche    4:marche    5:marche    6:arrêt
nscd            0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
nslcd           0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
ntpd            0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
ntpdate         0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
oddjobd         0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
portreserve     0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
postfix         0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
psacct          0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
rdisc           0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
restorecond     0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
rpcbind         0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
rpcgssd         0:arrêt    1:arrêt    2:arrêt    3:marche    4:marche    5:marche    6:arrêt
rpcidmapd       0:arrêt    1:arrêt    2:arrêt    3:marche    4:marche    5:marche    6:arrêt
rpcsvcgssd      0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
rsyslog         0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
saslauthd       0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
smartd          0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
snmpd           0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
snmptrapd       0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
sshd            0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
sssd            0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
sysstat         0:arrêt    1:marche    2:marche    3:marche    4:marche    5:marche    6:arrêt
udev-post       0:arrêt    1:marche    2:marche    3:marche    4:marche    5:marche    6:arrêt
vboxadd         0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
vboxadd-service 0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
vboxadd-x11     0:arrêt    1:arrêt    2:arrêt    3:marche    4:arrêt    5:marche    6:arrêt
wpa_supplicant  0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
ypbind          0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
```

chkconfig's command line switches are as follows:

```
[root@centos ~]# chkconfig --help
chkconfig version 1.3.47 - Copyright (C) 1997-2000 Red Hat, Inc.
Ce logiciel peut être librement distribué selon les termes de la licence publique GNU (GPL).

utilisation :   chkconfig [--list] [--type <type>] [nom]
              chkconfig --add <nom>
              chkconfig --del <nom>
         chkconfig --override <name>
         chkconfig [--level <niveaux>] [--type <type>] <nom> <on|off|reset|resetpriorities>
```

System Shutdown

The shutdown Command

The shutdown command's procedure includes :

- informing all connected users that the machine will shutdown,
- stopping all started services,
- committing all data to disk,
- unmounting all mounted filesystems.

shutdown's command line switches are as follows:

```
Usage:    shutdown [-akrhHPfnc] [-t secs] time [warning message]
                  -a:      use /etc/shutdown.allow
                  -k:      don't really shutdown, only warn.
                  -r:      reboot after shutdown.
                  -h:      halt after shutdown.
                  -P:      halt action is to turn off power.
                  -H:      halt action is to just halt.
                  -f:      do a 'fast' reboot (skip fsck).
                  -F:      Force fsck on reboot.
                  -n:      do not go through "init" but go down real fast.
                  -c:      cancel a running shutdown.
                  -t secs: delay between warning and kill signal.
                  ** the "time" argument is mandatory! (try "now") **
```

The **time** argument can take several values:

Value	Description
hh:mm	The time at which to shutdown
+m	Shutdown the system in m minutes
now	Shutdown immediately

<note important> If a shutdown is programmed for less than 5 minutes in the future any future connections are rejected, including those for root. </note>

If not using SELinux, the permission to execute shutdown can be given to a user by putting their name in the **/etc/shutdown.allow** file.

The reboot command

This command calls the **shutdown -r** command.

reboot's command line switches are as follows:

```
[root@centos ~]# reboot --help
Usage: reboot [OPTION]...
Reboot the system.

Options:
  -n, --no-sync               don't sync before reboot or halt
  -f, --force                 force reboot or halt, don't call shutdown(8)
  -p, --poweroff              switch off the power when called as halt
  -w, --wtmp-only             don't actually reboot or halt, just write wtmp
                                record
  -q, --quiet                 reduce output to errors only
  -v, --verbose               increase output to include informational messages
      --help                  display this help and exit
      --version               output version information and exit

This command is intended to instruct the kernel to reboot or halt the system;
when run without the -f option, or when in a system runlevel other than 0 or 6,
it will actually execute /sbin/shutdown.


Report bugs to <upstart-devel@lists.ubuntu.com>
```

The halt Command

This command calls the **shutdown -h** command.

halt's command line switches are as follows:

```
[root@centos ~]# halt --help
Usage: halt [OPTION]...
Halt the system.

Options:
  -n, --no-sync               don't sync before reboot or halt
  -f, --force                 force reboot or halt, don't call shutdown(8)
  -p, --poweroff              switch off the power when called as halt
  -w, --wtmp-only             don't actually reboot or halt, just write wtmp
                                record
  -q, --quiet                 reduce output to errors only
  -v, --verbose               increase output to include informational messages
      --help                  display this help and exit
      --version               output version information and exit
```

```
This command is intended to instruct the kernel to reboot or halt the system;
when run without the -f option, or when in a system runlevel other than 0 or 6,
it will actually execute /sbin/shutdown.


Report bugs to <upstart-devel@lists.ubuntu.com>
```

The poweroff Command

This command calls the **shutdown -hP** command.

halt's command line switches are as follows:

```
[root@centos ~]# poweroff --help
Usage: poweroff [OPTION]...
Power off the system.

Options:
  -n, --no-sync               don't sync before reboot or halt
  -f, --force                 force reboot or halt, don't call shutdown(8)
  -p, --poweroff              switch off the power when called as halt
  -w, --wtmp-only             don't actually reboot or halt, just write wtmp
                                record
  -q, --quiet                 reduce output to errors only
  -v, --verbose               increase output to include informational messages
      --help                  display this help and exit
      --version               output version information and exit

This command is intended to instruct the kernel to reboot or halt the system;
when run without the -f option, or when in a system runlevel other than 0 or 6,
it will actually execute /sbin/shutdown.
```

```
Report bugs to <upstart-devel@lists.ubuntu.com>
```

~~DISCUSSION:off~~
