

Niveau : Admin Junior	Numéro de la Leçon	Dernière Modification
2/4	<progreCSS 10/12 style=inline />	2020/01/30 03:28

Gestion du Démarrage et de l'Arrêt du Système

Détail du démarrage

BIOS

Au démarrage de la machine, le premier programme exécuté est le BIOS. Le BIOS a pour fonction de :

- Tester les composants et les circuits,
- Faire appel au BIOS de la carte graphique pour initialiser le système d'affichage,
- Détecter les périphériques de stockage,
- Lancer le **gestionnaire d'amorçage** du système d'exploitation.

Gestionnaire d'amorçage

Le gestionnaire d'amorçage est en règle générale placé dans le MBR du disque. Le format du MBR est le suivant :

- 446 octets pour le gestionnaire d'amorçage,
- 64 octets pour la table de partitions, soit 16 octets par partition décrite,
- 2 octets ayant une valeur fixe en hexadécimale de **AA55**.

Il est à noter que le gestionnaire d'amorçage peut également être installé dans un PBR.

Des gestionnaires d'amorçage sous Linux, un se distingue comme étant le plus utilisé :

- GRUB (Grand Unified Boot Loader)
-

Cependant il en existe d'autres :

- LILO (Linux LOader)
- SysLinux
- LoadLin
- ...

Grub 2

Debian utilise le gestionnaire d'amorçage **Grub2**. Grub2 est le successeur de Grub mais ne garde que le nom de ce dernier. En effet, Grub2 est une réécriture complète de Grub.

Dans le cas où le gestionnaire d'amorçage **Grub2** n'est pas installé, il convient de saisir la commande suivante :

```
# grub-install /dev/périphérique [Entrée]
```

où **périphérique** est le nom du périphérique où grub doit s'installer dans le MBR.

<note important> Dans le cas où le gestionnaire d'amorçage est **Grub** il est possible de passer à Grub2 en utilisant la commande suivante :

```
# upgrade-from-grub-legacy [Entrée]
```

</note>

Les fichiers de modules de Grub2, lui permettant de gérer le démarrage sur un nombre important de système de fichiers, se trouvent dans le répertoire **/boot/grub** :

```
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_ptthru.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 lit ses entrées de menus à partir du fichier **/boot/grub/grub.cfg**. Pour visualiser ce fichier, il convient de saisir la commande suivante :

```
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 a42alddd-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 a42alddd-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 a42alddd-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 ###
```

<note important> Notez que ce fichier ne doit pas être modifié manuellement. En effet, il est généré par la commande **update-grub**. </note>

Lors de l'exécution de la commande **update-grub**, plusieurs fichiers sont lus :

Le fichier **/boot/grub/device.map**

```
root@debian:~# cat /boot/grub/device.map  
(hd0)    /dev/disk/by-id/ata-VBOX_HARDDISK_VB5bb9d489-1757bae6
```

<note important> Notez que le fichier **/boot/grub/device.map** peut être configuré manuellement ou automatiquement par la commande **grub-mkdevicemap**. </note>

Le fichier **/etc/default/grub**

Ce fichier contient la configuration par défaut des paramètres de Grub2 :

```
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> Notez que toute modification de ce fichier nécessite l'exécution de la commande **update-grub** pour que les modifications soient prises en compte. </note>

Dans ce fichier les directives sont :

Directive	Description
GRUB_DEFAULT	Entrée du menu sélectionner par défaut
GRUB_TIMEOUT	Durée de l'affichage du menu avant le démarrage en utilisant la valeur de GRUB_DEFAULT
GRUB_DISTRIBUTOR	Ligne de commande qui génère le texte de l'entrée
GRUB_CMDLINE_LINUX_DEFAULT	Paramètres passés au noyau lors d'un démarrage normal (Hors donc le mode secours)
GRUB_CMDLINE_LINUX	Paramètres passés au noyau peu importe le type de démarrage
GRUB_TERMINAL	Si non commentée, cette directive désactive le démarrage graphique

Directive	Description
GRUB_GFXMODE	Indique la résolution utilisée lors d'un démarrage graphique
GRUB_DISABLE_LINUX_UUID	Si true , cette directive empêche l'utilisation de l'UUID de la partition
GRUB_DISABLE_LINUX_RECOVERY	Si true , cette directive empêche la génération des entrées en mode recovery
GRUB_INIT_TUNE	Permet d'obtenir un beep au démarrage de Grub2
GRUB_BADRAM	Permet de spécifier de la mémoire défectueuse

Les fichiers du répertoire /etc/grub.d

Les fichiers de ce répertoire sont exécutés dans l'ordre alphanumérique et servent à construire les menus de Grub2 :

```
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
```

Le fichier /etc/grub.d/10_Linux

Le fichier **10_Linux** contient des boucles pour rechercher des noyaux Linux :

```
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 '[:upper:]' '[:lower:]' | 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
    fi
}
```

```
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

Ce fichier recherche des éventuels systèmes d'exploitation autre que Linux :

```
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`"
    LB00T="`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 [ "${LROOT}" != "${LB00T}" ]; 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 ${LB00T} | sed -e "s/^/\t/")"
    fi
    printf '%s\n' "${prepare_boot_cache}"
    cat << EOF
linux ${LKERNEL} ${LPARAMS}
EOF
    if [ -n "${LINITRD}" ] ; then
        cat << EOF
        initrd ${LINITRD}
    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
```

Les fichiers `/etc/grub.d/40_custom` et `/etc/grub.d/41_custom`

Ces deux fichiers sont fournis en tant que modèles à personnaliser :

```
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> La personnalisation d'un fichier Grub2 est couverte dans la section **Initramfs sous Debian** de cet unité. </note>

Initramfs sous Debian

Initrd *INITial Ram File System* est une image d'un système minimal initialisé au démarrage du système.

Ce fichier utilise le système de fichier **cramFS** qui est un système de fichier compressé au format gzip et archivé via cpio.

L'image est chargée en mémoire vive et permet ainsi d'avoir un système minimal pouvant ensuite charger le système de fichier principal.

Examiner l'image existante

Pour examiner l'image existante, il convient d'abord de la copier vers /tmp et de la décompresser :

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

Ensuite il convient d'extraire l'image grâce à la commande **cpio** :

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

Installez maintenant le paquet **tree** :

```
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) ...
```

Utilisez la commande **tree** pour examiner le contenu de l'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--
```

Comme vous pouvez le constater, l'image contient une arborescence Linux minimaliste :

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

<note> Utilisez le manuel de la commande **cpio** pour comprendre les options utilisées. </note>

Le script **init**

Le script **init** est lancé lors du chargement de l'image :

```
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%)
```

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

La commande **mkinitramfs**

La commande **mkinitramfs** permet de créer facilement une image initramfs.

Afin d'inclure des modules supplémentaires dans un nouveau fichier initramfs, il convient d'éditer le fichier **/etc/initramfs-tools/modules** :

```
# 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
```

Exécutez ensuite la commande suivante :

```
# mkinitramfs -o usbinitramfs-`uname -r`.img [Entrée]
```

Notez la présence de votre nouvelle image **/tmp/initrd/usbinitramfs-2.6.32-5-686.img**.

Déplacez votre fichier **usbinitramfs-2.6.32-5-686.img** au répertoire /boot :

```
root@debian:/tmp/initrd# mv usbinitramfs-2.6.32-5-686.img /boot
root@debian:/tmp/initrd# cd /boot
root@debian:/boot# ls
config-2.6.32-5-686  grub  initrd.img-2.6.32-5-686  System.map-2.6.32-5-686  usbinitramfs-2.6.32-5-686.img
vmlinuz-2.6.32-5-686
```

Créez maintenant le fichier **/etc/grub.d/09_usbdebian** :

```
#!/bin/sh -e
cat << EOF
menuentry "Debian GNU/Linux, avec usbinitramfs-2.6.32-5-686" {
set root=(hd0,msdos1)
search --no-floppy --fs-uuid --set a42alddd-14bc-4dde-a537-e6c1b984a782
echo    'Chargement de Linux 2.6.32-5-686 ...'
linux   /boot/vmlinuz-2.6.32-5-686 root=UUID=a42alddd-14bc-4dde-a537-e6c1b984a782 ro selinux=1 quiet
echo    'Chargement du disque mémoire initial ...'
initrd  /boot/usbinitramfs-2.6.32-5-686.img
}
EOF
```

Rendez ce fichier exécutable :

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

Mettez à jour grub afin que celui-ci prend en compte le nouveau fichier :

```
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
```

Contrôlez le fichier /boot/grub/grub.cfg :

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

### BEGIN /etc/grub.d/09_usbdebian ###
menuentry "Debian GNU/Linux, avec usbinitramfs-2.6.32-5-686" {
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-2.6.32-5-686.img
}
### END /etc/grub.d/09_usbdebian ###

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

<note> **Re-démarrez votre machine pour tester votre configuration.** </note>

Processus de Démarrage de Linux

Le processus de démarrage de Linux peut être divisé en 6 étapes :

Etape	Description
Chargement, configuration et exécution du chargeur du noyau	Le fichier bootsect.s est chargé en mémoire par le BIOS. Une fois configuré celui-ci charge le reste du noyau en mémoire
Configuration des paramètres et bascule vers le mode 32 bits	Le fichier boot.s met en place un IDT (<i>Interrupt Descriptor Table</i>) temporaire et GDT (<i>Global Descriptor Table</i>) temporaire et gère le basculement vers le mode 32 bits
Décompression du Noyau	Le fichier head.s décompresse le noyau
Initialisation du noyau et de la mémoire	Le fichier head.s crée un GDT et IDT définitif
Configuration du noyau	Le fichier main.c met en place les contraintes de mémoire et configure la mémoire virtuelle
Création du processus Init	Le fichier main.c crée le processus init

La fonction **init_post()** essaie ensuite d'exécuter un des processus suivant dans l'ordre :

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

Dans le cas d'un échec à ce stade le message **Kernel Panic** sera afficher.

Processus Init

Le premier processus lancé par le noyau est **Init**. L'exécutable lancé est **/sbin/init**. Son rôle est de d'initialiser le système et de lancer certains autres services. Les tâches accomplies par init sont :

- le montage de /proc et de /sys
- configuration des paramètres du noyau présents dans **/etc/sysctl.conf**,
- l'activation de SELinux,
- la mise à l'heure du système,

- la définition des consoles textes,
- la définition du nom de la machine,
- la détection des périphériques USB,
- la mise en place du support RAID et LVM,
- l'activation des quotas de disque,
- le montages des systèmes de fichiers,
- le re-montage du système de fichiers racine en lecture/écriture,
- l'activation du swap,
- le lancement de syslog,
- le chargement des modules du noyau,
- le nettoyage des fichiers temporaires,
- la définition des variables d'environnement tels PATH et RUNLEVEL

Niveaux d'exécution

Il existe 8 niveaux d'exécution ou **RUNLEVELS** sous Linux. Quatre des 8 sont réservés :

RUNLEVEL	Description
0	Arrêt de la machine
1	Mode mono-utilisateur pour la maintenance
6	Redémarrage de la machine
S ou s	Mode mono-utilisateur avec seul la partition racine montée

Les autres quatre RUNLEVELS sont définis par chaque distribution. Par exemple, sous Debian, ils sont :

RUNLEVEL	Description
2	Mode multi-utilisateur
3	Mode multi-utilisateur - Non-utilisé
4	Mode multi-utilisateur - Non-utilisé
5	Mode multi-utilisateur - Non-utilisé

Il existe aussi 3 pseudo-niveaux d'exécution **a**, **b** et **c**. Ces pseudo-niveaux permettent à init de faire quelque chose sans changer de niveau

d'exécution.

Pour connaître le niveau d'exécution actuel de la machine, saisissez la commande suivante :

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

La lettre N indique que le système n'a pas changé de niveau d'exécution depuis son démarrage.

Pour modifier le niveau d'exécution courant, il convient d'utiliser la commande **init** ou **telinit** suivie du numéro du nouveau niveau d'exécution. Ces commandes peuvent prendre plusieurs options :

Option	Description
Q ou q	Demande à Init de relire le fichier /etc/inittab
-t	Permet de modifier le temps accordé par Init aux processus entre l'envoi du signal SIGTERM et l'envoi du signal SIGKILL

Inittab

Le fichier /etc/inittab permet de définir les services à démarrer en fonction du RUNLEVEL :

```
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
```

Dans l'exemple ci-dessus, chaque ligne non-commentée est composée de quatre champs, séparés par le caractère :

Champ	Nom	Description
1	Identifiant	Identifiant unique de la ligne composé de 1 à 4 caractères
2	RUNLEVELS	Liste des niveaux d'exécution concernés par la ligne
3	Action	Méthode utilisé pour lancer la commande se trouvant dans le champ 4
4	Commande	Commande à lancer

Le champ **action** prend une des directives suivantes :

Directive	Description
respawn	Le processus est relancé en cas d'arrêt de celui-ci

Directive	Description
mingetty	Assure la gestion du terminal texte
once	Le processus n'est exécuté qu'une fois
wait	Le processus n'est exécuté qu'une fois. Init attend la fin du processus avant de passer à la ligne suivante
boot	Le processus est exécuté au démarrage de la machine. Le champ RUNLEVELS est sans importance
bootwait	Le processus est exécuté au démarrage de la machine. Le champ RUNLEVELS est sans importance. Init attend la fin du processus avant de passer à la ligne suivante
off	Revient à commenter la ligne
ondemand	La même chose que respawn mais la commande est exécuté dans un des 3 pseudo-niveaux d'exécution
initdefault	Définit le niveau d'exécution par défaut
sysinit	La commande est exécutée au démarrage de la machine avant les lignes boot et bootwait
powerfail	La commande est exécutée quand init reçoit un signal SIGPWR d'un onduleur
powerwait	La commande est exécutée quand init reçoit un signal SIGPWR d'un onduleur. Init attend la fin du processus avant de passer à la ligne suivante
powerokwait	La commande est exécutée si Init reçoit un signal de rétablissement du courant
powerfailnow	La commande est exécutée quand Init reçoit un signal comme quoi la batterie de l'onduleur est presque vide
ctrlaltdel	La commande est exécutée quand Init reçoit un signal SIGINT. Ce signal est envoyé par la combinaison de touches [CTRL] [ALT] [SUPPR]
kbrequest	La commande est exécutée suivant des séquences de touches saisies au clavier

L'analyse de notre fichier d'exemple indique donc :

Ligne	Description
id:2:initdefault:	Le niveau d'exécution par défaut est 2
si::sysinit:/etc/init.d/rcS	Le script /etc/init.d/rcS est lancé au démarrage de la machine
~~:S:wait:/sbin/sulogin	La commande /sbin/sulogin est lancée au démarrage de la machine en niveau d'exécution S
l0:0:wait:/etc/init.d/rc 0	Le script /etc/rc.d/rc est lancé pour le niveau d'exécution 0
l1:1:wait:/etc/init.d/rc 1	Le script /etc/rc.d/rc est lancé pour le niveau d'exécution 1
l2:2:wait:/etc/init.d/rc 2	Le script /etc/rc.d/rc est lancé pour le niveau d'exécution 2
l3:3:wait:/etc/init.d/rc 3	Le script /etc/rc.d/rc est lancé pour le niveau d'exécution 3
l4:4:wait:/etc/init.d/rc 4	Le script /etc/rc.d/rc est lancé pour le niveau d'exécution 4
l5:5:wait:/etc/init.d/rc 5	Le script /etc/rc.d/rc est lancé pour le niveau d'exécution 5
l6:6:wait:/etc/init.d/rc 6	Le script /etc/rc.d/rc est lancé pour le niveau d'exécution 6

Ligne	Description
z6:6:respawn:/sbin/sulogin	La commande /sbin/sulogin est lancée au redémarrage de la machine si nécessaire
ca:12345:ctrlaltdel:/sbin/shutdown -t1 -a -r now	La commande /sbin/shutdown -t1 -a -r now est lancée si les touches [CTRL] [ALT] [SUPPR] sont appuyées simultanément
pf::powerwait:/etc/init.d/powerfail start	Le script /etc/init.d/powerfail start est lancée quand Init reçoit le signal SIGPWR
pn::powerfailnow:/etc/init.d/powerfail now	Le script /etc/init.d/powerfail now est lancée quand Init reçoit un signal comme quoi la batterie de l'onduleur est presque vide
po::powerokwait:/etc/init.d/powerfail stop	Le script /etc/init.d/powerfail stop est lancée quand Init reçoit un signal de rétablissement du courant
1:2345:respawn:/sbin/getty 38400 tty1	Le terminal tty1 est initialisé par la commande /sbin/mingetty. Le terminal est disponible en appuyant sur les touches [CTRL] [ALT] [F1]
2:23:respawn:/sbin/getty 38400 tty2	Le terminal tty2 est initialisé par la commande /sbin/mingetty. Le terminal est disponible en appuyant sur les touches [CTRL] [ALT] [F2]
3:23:respawn:/sbin/getty 38400 tty3	Le terminal tty3 est initialisé par la commande /sbin/mingetty. Le terminal est disponible en appuyant sur les touches [CTRL] [ALT] [F3]
4:23:respawn:/sbin/getty 38400 tty4	Le terminal tty4 est initialisé par la commande /sbin/mingetty. Le terminal est disponible en appuyant sur les touches [CTRL] [ALT] [F4]
5:23:respawn:/sbin/getty 38400 tty5	Le terminal tty5 est initialisé par la commande /sbin/mingetty. Le terminal est disponible en appuyant sur les touches [CTRL] [ALT] [F5]
6:23:respawn:/sbin/getty 38400 tty6	Le terminal tty6 est initialisé par la commande /sbin/mingetty. Le terminal est disponible en appuyant sur les touches [CTRL] [ALT] [F6]

Répertoire init.d

Le répertoire **/etc/init.d** contient les scripts permettant de lancer les services du système :

```
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

D'après l'étude du fichier **inittab**, nous savons que le script `/etc/init.d/rcS` est exécuté en premier. Ce script appelle tous les scripts dans `/etc/init.d` qui sont référencés par les liens dans `/etc/rcS.d/` et ceci dans un ordre numérique/alphabétique :

```
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
```

Répertoires rcx.d

Les répertoires **/etc/rc0.d** à **/etc/rc6.d** contiennent des liens vers les scripts du répertoire **init.d**.

Pour mieux comprendre, saisissez les commandes suivantes :

```
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> Notez que chaque répertoire correspondant à un niveau d'exécution contient des liens pointant vers un script dans le répertoire **/etc/init.d**. La lettre **S** indique au script **rc** que le script dans **/etc/init.d** doit être exécuté avec l'option **start**. De cette façon les processus sont lancés dans le niveau d'exécution spécifié. Le numéro qui suit la lettre **S** indique l'ordre de lancement par le script **rc**. Si deux scripts dans un répertoire **/etc/rcX.d** ont le même numéro, l'ordre alphabétique prime. Notez aussi la présence du lien **S99rc.local** qui lance le script **rc.local** en dernier. Ce script peut contenir des commandes spécifiées par root à lancer lors du démarrage de la machine. </note>

Rappelez la commande précédente et modifiez la lettre S en **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
```

Ici le principe est le même sauf que la lettre **K** indique au script **rc** que le script dans `/etc/init.d` doit être lancé avec l'option **stop**.

rc.local

Le script `rc.local` est lancé dans les niveaux d'exécution **2, 3, 4 et 5**. C'est dans ce script que **root** peut ajouter des commandes.

La commande update-rc.d

La commande **update-rc.d** est utilisée pour gérer les liens dans les répertoires `rc[x].d`. Cette commande permet d'insérer les liens vers un script dans `/etc/init.d` et prend la forme suivante :

```
update-rc.d <service> start <priorité de démarrage> <runlevels de démarrage> . stop <priorité d'arrêt> <runlevels d'arrêt> .
```

Par exemple la commande suivante crée les liens **S** dans `rc2.d` à `rc5.d` avec une priorité de 20 et les liens **K** dans `rc0.d`, `rc1.d` et `rc6.d` avec une priorité

de 20 pour le script /etc/init.d/ssh :

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

<note important> Il existe aussi un mot clef : **default**. Ce mot clef indique à **update-rc.d** d'utiliser les valeurs spécifiées dans la command précédente. </note>

Pour supprimer les liens il convient d'utiliser la commande suivante :

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

<note important> Cette commande laisse le script ssh dans le répertoire /etc/init.d. </note>

Options de la commande

Les options de cette commande sont :

```
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.

La commande chkconfig

Pour avoir une vue globale des services lancés par niveau d'exécution nous pouvons utiliser la commande ***chkconfig**. Sous Debian la commande **chkconfig** n'est pas installée par défaut. Il convient donc de l'installer en utilisant la commande **apt-get** :

```
root@debian:/etc/init.d# apt-get install chkconfig
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les NOUVEAUX paquets suivants seront installés :
  chkconfig
0 mis à jour, 1 nouvellement installés, 0 à enlever et 219 non mis à jour.
Il est nécessaire de prendre 9 182 o dans les archives.
Après cette opération, 69,6 ko d'espace disque supplémentaires seront utilisés.
Réception de : 1 http://ftp.fr.debian.org/debian/ squeeze/main chkconfig all 11.0-79.1-2 [9 182 B]
9 182 o réceptionnés en 50s (183 o/s)
(Lecture de la base de données... 130287 fichiers et répertoires déjà installés.)
Dépaquetage de chkconfig (à partir de .../chkconfig_11.0-79.1-2_all.deb) ...
Traitement des actions différées (« triggers ») pour « man-db »...
Paramétrage de chkconfig (11.0-79.1-2) ...
```

Saisissez maintenant la commande suivante :

```
root@debian:/tmp# 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

Options de la commande

Les options de la commande **chkconfig** sont :

```
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
```

Les options les plus importantes sont :

Option	Description
- -add (nom)	Permet d'ajouter un service aux runlevels définis par le service lui-même
- -del (nom)	Permet de supprimer un service aux runlevels définis par le service lui-même
- -level [niveau] (nom) [on off reset]	Permet d'activer, de désactiver ou de réinitialiser un service inscrit

Arrêt du Système

La commande shutdown

Lors de l'arrêt de la machine, Linux procède, entre autre, aux tâches suivantes :

- Il prévient les utilisateurs,
- Il arrête tous les services,
- Il inscrit toutes les données sur disque,
- Il démonte les systèmes de fichiers.

La commande utilisée pour arrêter le système est la commande **shutdown** :

```
shutdown [-t sec] [-akrhHPfnc] heure [message]
```

Options de la commande

Les options de cette commande sont :

```
root@debian:/etc/init.d# shutdown --help
shutdown: invalid option -- '-'
Usage:      shutdown [-akrhPHfFc] [-t sec] 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") **
```

Parmi les options les plus importantes, on note :

Option	Description
-h	Arrêter le système
-r	Re-démarrer le système
-c	Annuler l'opération shutdown en cours
-f	Re-démarrer rapidement sans vérifier les systèmes de fichiers
-F	Forcer la vérification des systèmes de fichiers lors du prochain démarrage

L'option **time** peut prendre plusieurs valeurs :

Valeur	Description
hh:mm	L'heure à laquelle l'opération aura lieu

Valeur	Description
+m	Nombre de minutes avant que l'opération aura lieu
now	L'opération est immédiate

<note important> Si l'opération est programmée pour dans moins de 5 minutes, les connexions supplémentaires sont interdites, y comprises les tentatives de connexion de root. </note>

L'utilisation de la commande **shutdown** peut être accordée à d'autres utilisateurs de root en utilisant le fichier **/etc/shutdown.allow**

La commande reboot

Cette commande redémarre le système. Quand le système fonctionne normalement, l'exécution de reboot appelle la commande **shutdown -r**.

Options de la commande

Les options de cette commande sont :

```
root@debian:/etc/init.d# reboot --help
usage: reboot [-n] [-w] [-d] [-f] [-h] [-i]
  -n: don't sync before halting the system
  -w: only write a wtmp reboot record and exit.
  -d: don't write a wtmp record.
  -f: force halt/reboot, don't call shutdown.
  -h: put harddisks in standby mode.
  -i: shut down all network interfaces.
```

La commande halt

Cette commande arrête le système. Quand le système fonctionne normalement, l'exécution de halt appelle la commande **shutdown -h**.

Options de la commande

Les options de cette commande sont :

```
root@debian:/etc/init.d# halt --help
usage: halt [-n] [-w] [-d] [-f] [-h] [-i] [-p]
  -n: don't sync before halting the system
  -w: only write a wtmp reboot record and exit.
  -d: don't write a wtmp record.
  -f: force halt/reboot, don't call shutdown.
  -h: put harddisks in standby mode.
  -i: shut down all network interfaces.
  -p: power down the system (if possible, otherwise halt).
```

La commande poweroff

Cette commande arrête le système et coupe l'alimentation électrique. Elle est l'équivalente de la commande **halt -p**. Quand le système fonctionne normalement, l'exécution de **poweroff** appelle la commande **shutdown -hP**.

Options de la commande

Les options de cette commande sont :

```
root@debian:/etc/init.d# poweroff --help
usage: poweroff [-n] [-w] [-d] [-f] [-h] [-i]
  -n: don't sync before halting the system
  -w: only write a wtmp reboot record and exit.
  -d: don't write a wtmp record.
  -f: force halt/reboot, don't call shutdown.
  -h: put harddisks in standby mode.
```

```
-i: shut down all network interfaces.
```

~~DISCUSSION:off~~

Donner votre Avis

```
{(rater>id=debian_6_l116|name=cette page|type=rate|trace=user|tracedetails=1)}
```