

Laboratorio 2



Apellidos: Moreno Vera

Nombres: Felipe Adrian

Código: 20120354I

Asignatura: Administración de Redes (CC481)

2016 - I

Indice

Actividad 1	(3)
Actividad 2	(6)
Actividad 3	(13)
Actividad 4	(15)
Actividad 5	(21)
Actividad 6	(21)
Actividad 7	(23)
Actividad 8	(26)

Actividad 1

1. Realice un pequeño resumen del significado de Kickstart y PXE en GNU/Linux.

a) Kickstart es un método de instalación de sistemas red hat tal que se crea un archivo con todas las indicaciones y opciones de instalación y sus valores asociados. y define como configurar el sistema de almacenamiento. Al momento de hacer la instalación, kickstart lee ese archivo en un host remoto, se usa para automatizar la instalación de red hat en múltiples máquinas.

```
#platform=x86, AMD64, or Intel EM64T
#version=DEVEL
# Firewall configuration
firewall --enabled --service=ssh

# Install OS instead of upgrade
install

# Use CDRom installation media
cdrom
repo --name="Oracle Linux Server" --baseurl=cdrom:sr0 --cost=100

# System authorization information
auth --useshadow --passalgo=sha512

# Root password
rootpw --iscrypted SHA512_password_hash

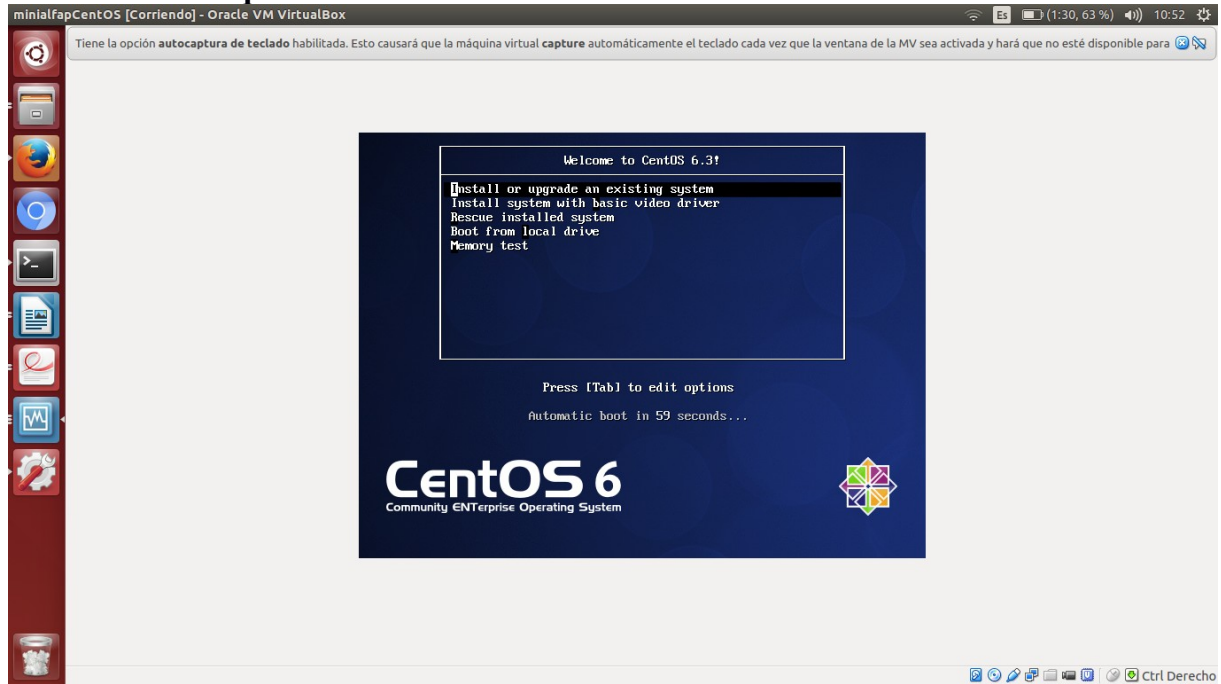
# Use graphical install
graphical
firstboot --disable
```

b) PXE (Preboot eXecution Environment), Es un entorno para arrancar e instalar el sistema operativo en ordenadores remotos a través de una red.

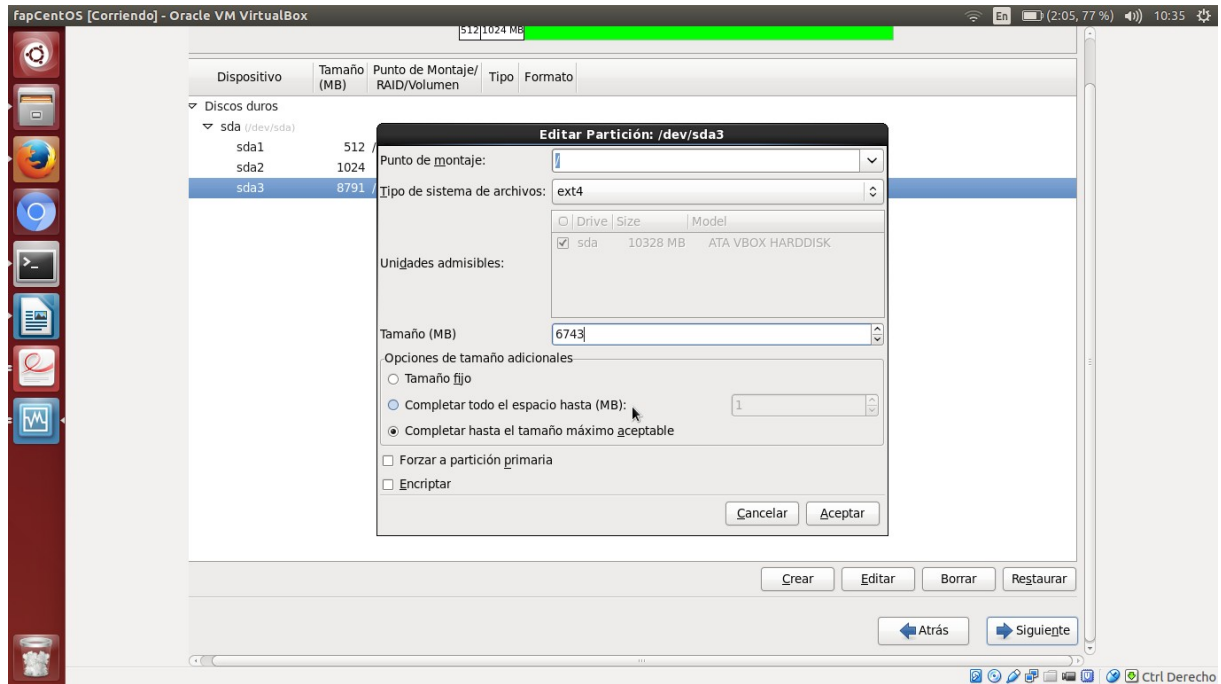
Lo hace mediante un servidor y un cliente PXE, el cliente trata de encontrar un servicio de redirección PXE en la red buscando un servidor de arranque PXE, luego al encontrar, se solicita al servidor de arranque el file path network bootstrap program (NBP), se descargará y almacenará en RAM mediante FTP y finalmente ejecutará.

Instalación de CentOS.

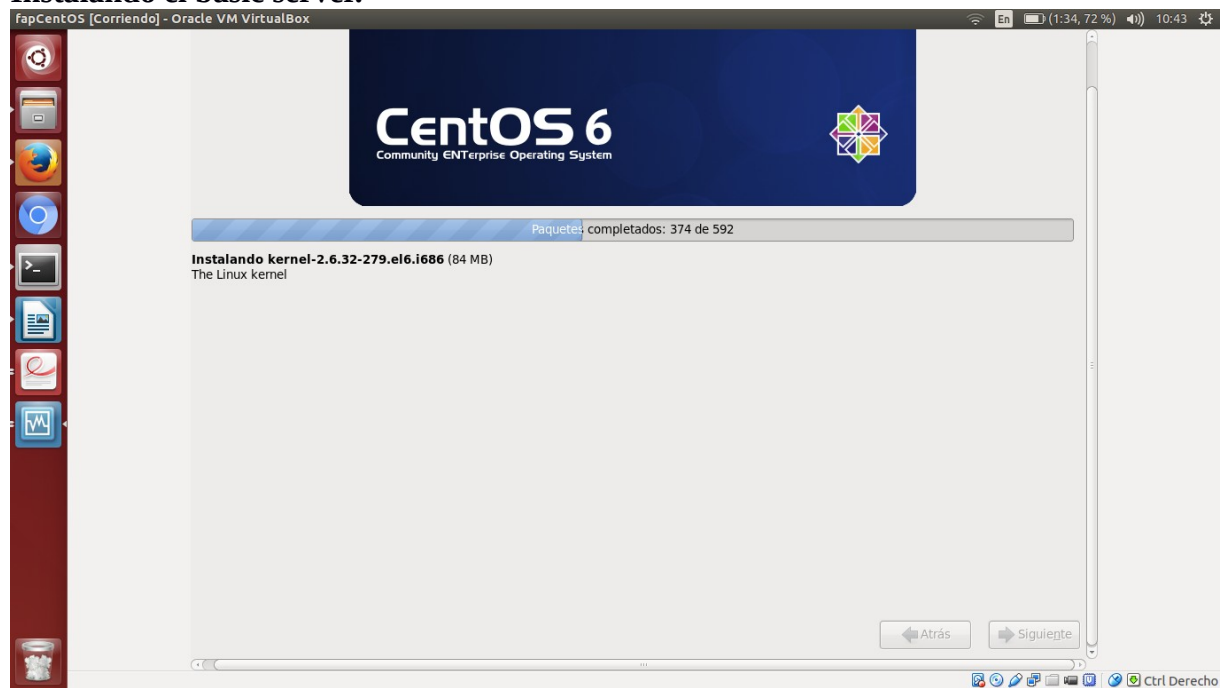
Corriendo la máquina virtual en virtualbox.



Particionando

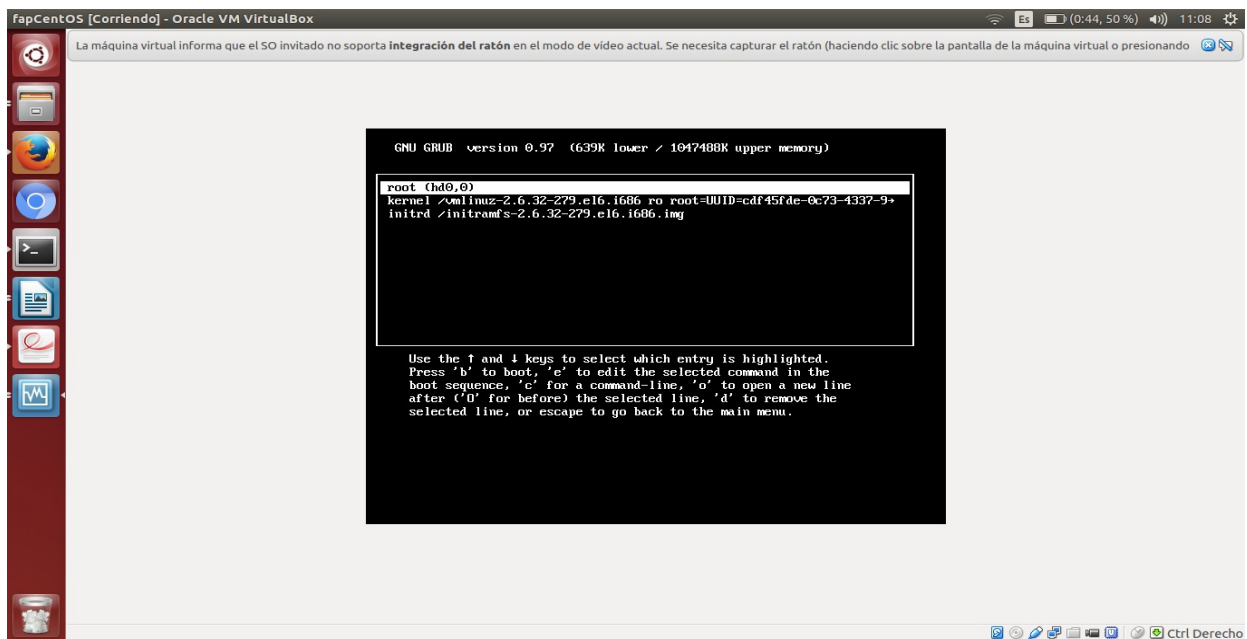


Instalando el basic server.



Actividad 2

1. Entrar en el modo edición de la carga del sistema presionando 'e', mostrará la configuración para la carga del sistema operativo. Esta entrada contiene tres líneas, determinar qué significa cada una (i.e. root (hd0,0)...) con la ayuda de la información sobre grub (puede consultarse en el propio host anfitrión con info grub, sección de configuración).

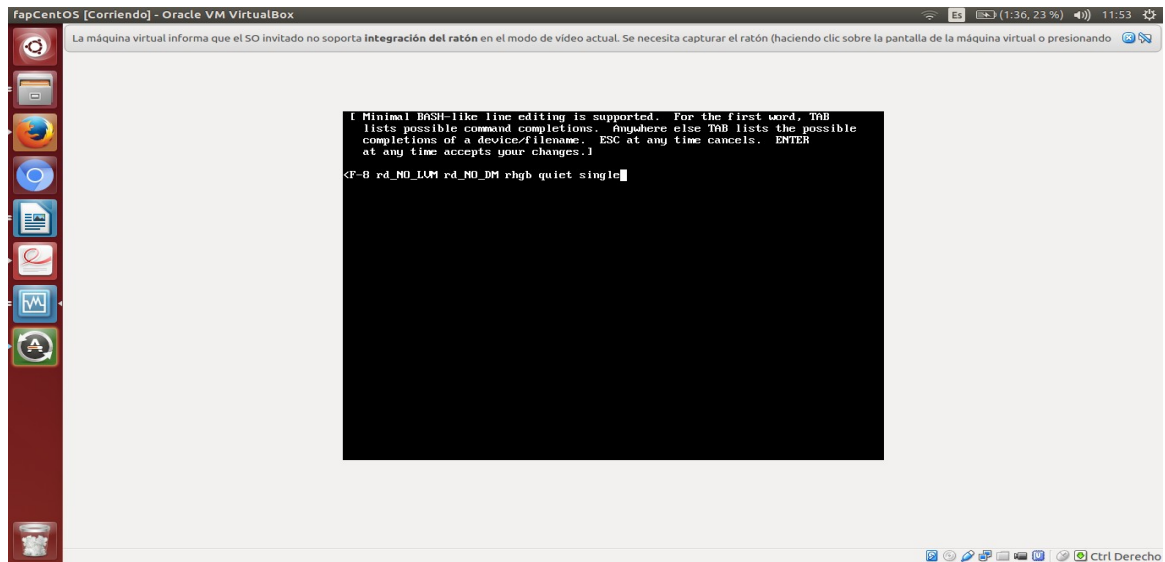


En la primera línea: root (hd0,0) indica al grub donde está el directorio raíz (la partición /boot) hd0 es primera partición y 0 es el primer drive físico.

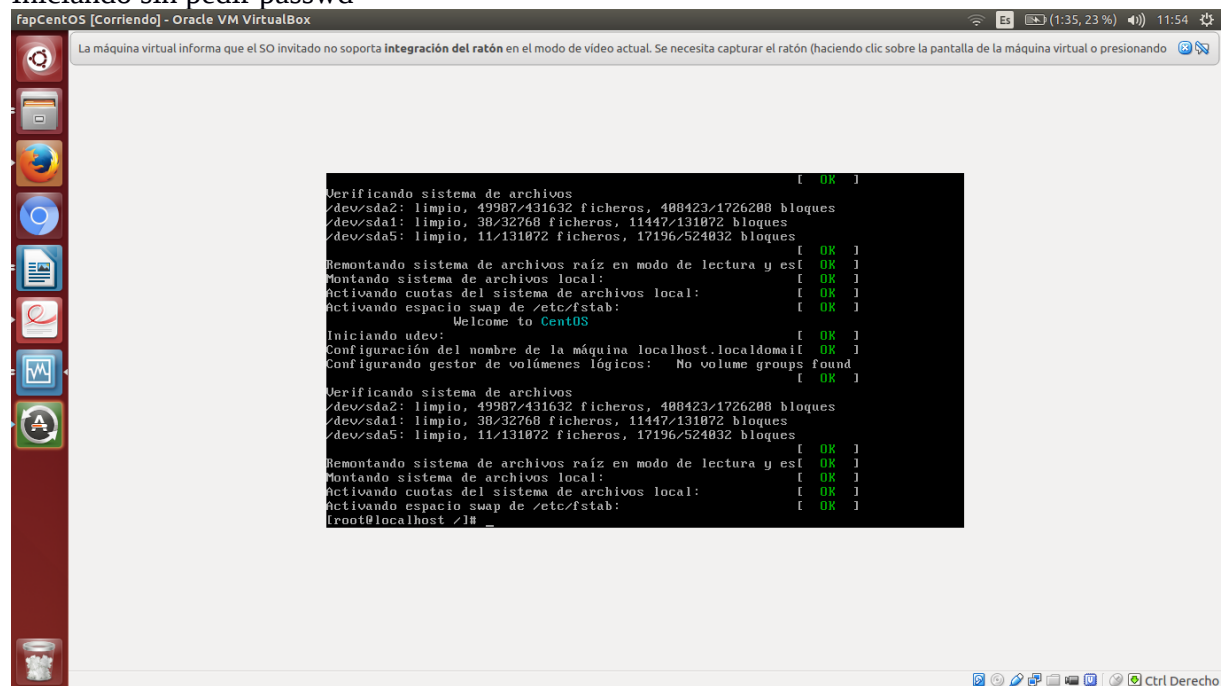
En la segunda línea: kernel /vmlinuz-2.6.32-279.el6.i686 ro root=UUID=cdf45fde-0c73-4337-9 indica a el kernel donde esta la partición root que va a cargar(puede haber mas de uno en un directorio raiz).

En la tercera línea: initrd /initramf indica el esquema para la carga de una raíz temporal del sistema de archivos en la memoria RAM. Initrd y initramf se refieren a 2 métodos diferentes de conseguir dicha carga.

2. Abrir para edición la línea del kernel, el último parámetro que se añade a esta línea sirve para indicar el nivel de arranque, que veremos más adelante. Muchas veces para reparar un sistema debe arrancar en modo mono-usuario, probar las opciones single o init=/bin/sh al final. Arrancar el sistema con 'b'. Comprobar que no se solicita contraseña y la importancia de proteger la BIOS con passwd en entornos no seguros.



Iniciando sin pedir passwd



3. Reiniciar el sistema con el comando reboot. Estudiar los contenidos del directorio /boot y relacionarlos con los del fichero /boot/grub/menu.lst.

Cambiar por ejemplo el título, el tiempo de espera por defecto y los colores del GRUB.

1. CentOS 7: Para ello modificaremos el fichero que se encuentra en /etc/default/grub y recompilaremos el grub con grub2-mkconfig -o /boot/grub2/grub.cfg.

```
[root@localhost boot]# cd /boot/
[root@localhost boot]# ls
config-2.6.32-279.el6.i686      lost+found
efi                             symvers-2.6.32-279.el6.i686.gz
grub                           System.map-2.6.32-279.el6.i686
initramfs-2.6.32-279.el6.i686.img vmlinuz-2.6.32-279.el6.i686
[root@localhost boot]# cd grub/
[root@localhost grub]# ls
device.map      grub.conf      minix_stage1_5    stage2
e2fs_stage1_5  iso9660_stage1_5  reiserfs_stage1_5 ufs2_stage1_5
fat_stage1_5   jfs_stage1_5      splash.xpm.gz     vstafs_stage1_5
ffs_stage1_5   menu.lst          stage1            xfs_stage1_5
[root@localhost grub]# Meow xd_
```

El contenido del directorio /boot son las carpetas grub efi y lost+found.
Las imágenes de sistema y el fichero de configuración.

En el fichero /boot/grub/menu.lst tenemos:

```
[root@localhost grub]# cat 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 (2.6.32-279.el6.i686)
    root (hd0,0)
    kernel /vmlinuz-2.6.32-279.el6.i686 ro root=UUID=cdf45fde-0c73-4337-9ca1
-62a88f2349b2 rd_NO_LUKS KEYBOARDTYPE=pc KEYTABLE=es rd_NO_MD SYSFONT=latacyrh
eb-sun16 crashkernel=auto LANG=es_ES.UTF-8 rd_NO_LVM rd_NO_DM rhgb quiet
    initrd /initramfs-2.6.32-279.el6.i686.img
[root@localhost grub]#
[root@localhost grub]# Meow xd_
```

vemos que tenemos la lista de imágenes boteables que nos muestra al inicio del grub, cuando

vamos a la opción “e” en la opción de CentOS.

Y 3 variables:

default = De la lista de Sistemas Operativos que se muestran en el grub, el que carga por defecto es el número o un título de menú que indica este parámetro.

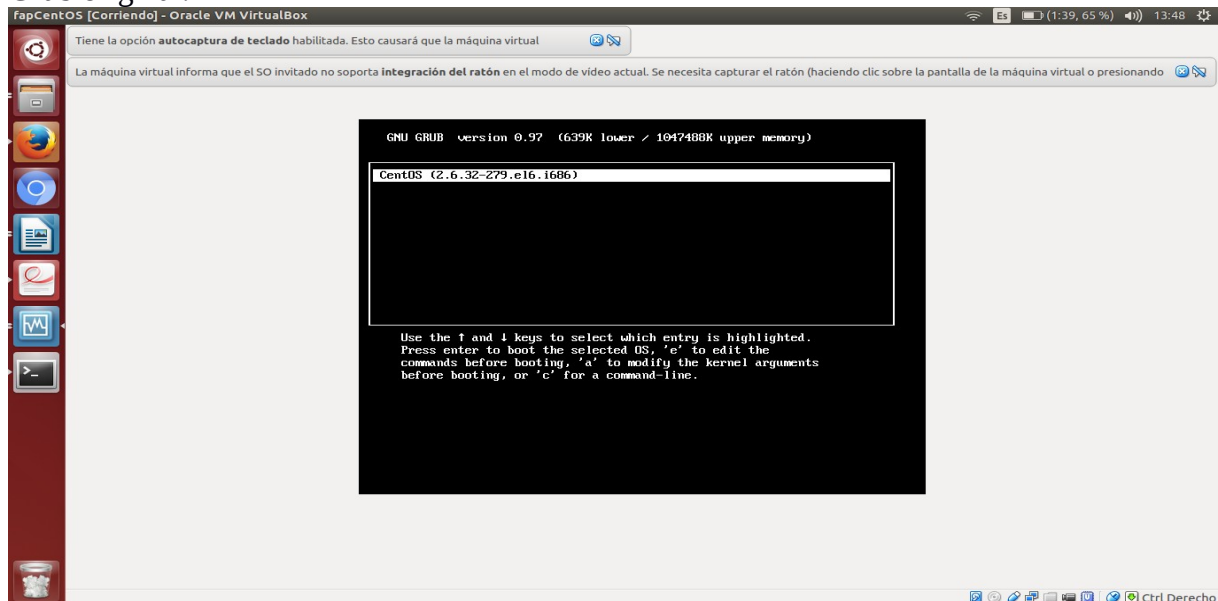
Timeout = Es el tiempo de espera en segundos que el grub espera antes de cargar el Sistema operativo por defecto.

splashimage = Imagen de fondo del grub.

hiddenmenu = es el que aparecerá cuando apretamos la tecla de “e” en el tittle que escojamos.

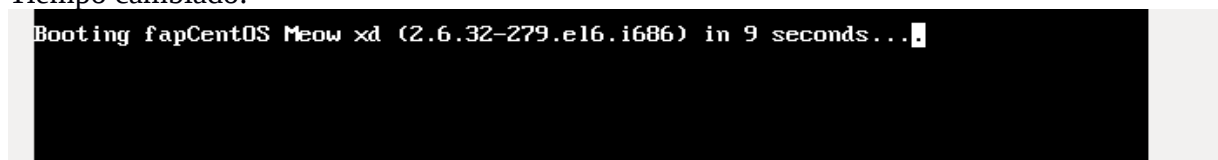
Title = Es el nombre que va a tener el contenido de entradas posteriores, indica el Sistema Operativo que va a cargar (de preferencia).

Grub original:

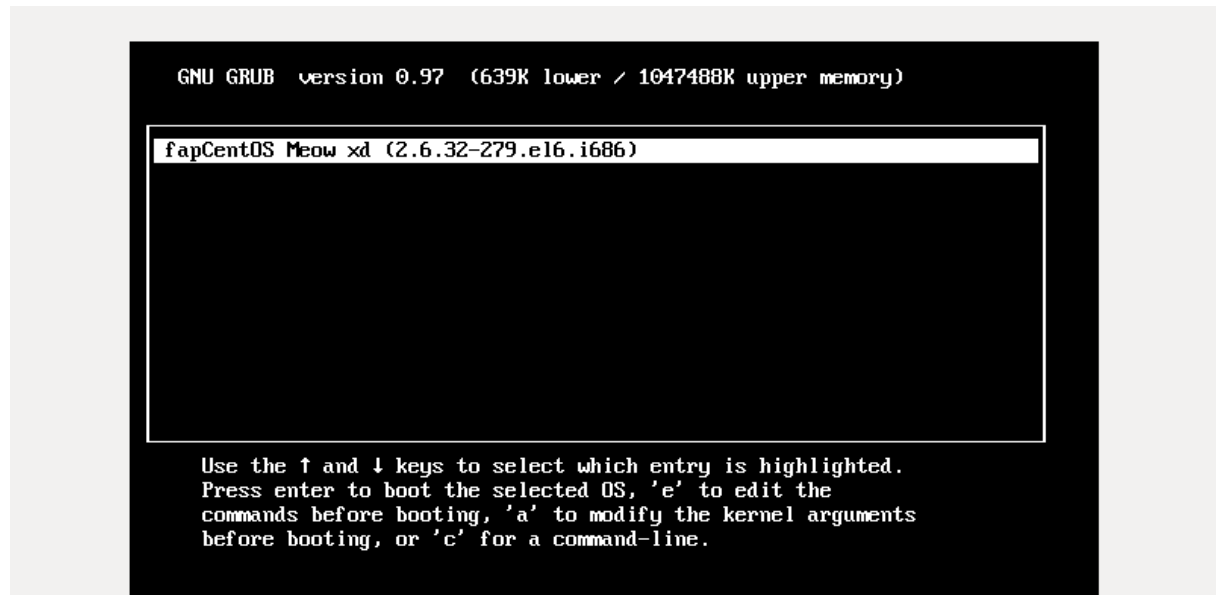


Grub cambiado:

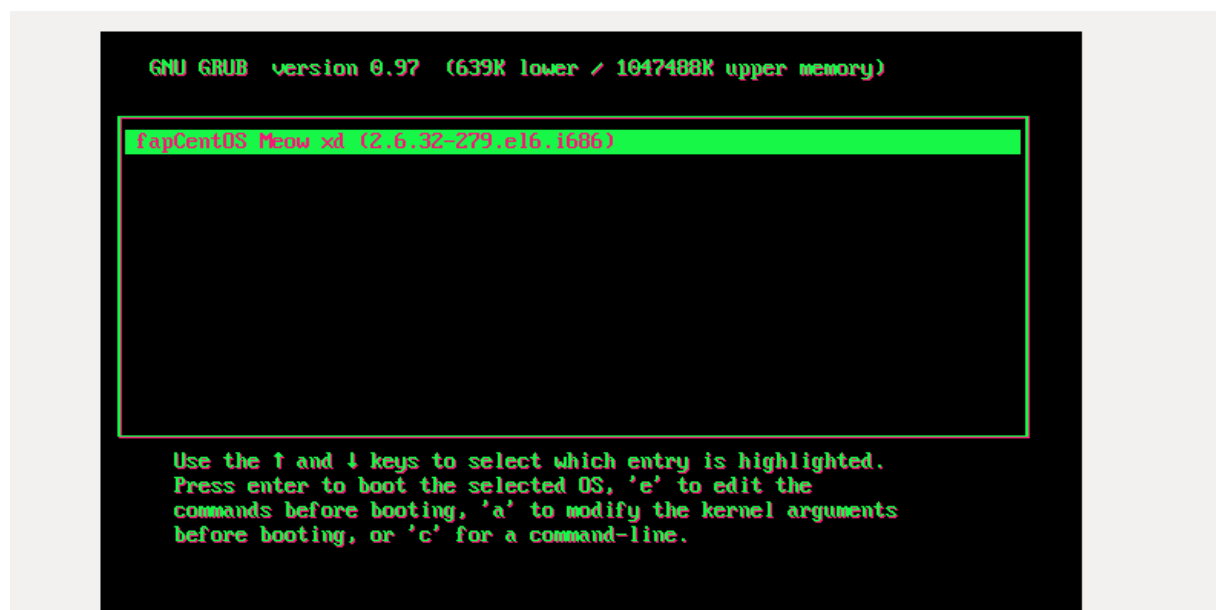
Tiempo cambiado:



nombre cambiado:



color cambiado:



4. Antes de seguir busque información de dónde se encuentra y como se llama el Kernel instalado en el sistema.

```
[root@localhost ~]# uname -a
Linux localhost.localdomain 2.6.32-279.el6.i686 #1 SMP Fri Jun 22 10:59:55 UTC 2
012 i686 i686 i386 GNU/Linux
[root@localhost ~]#
[root@localhost ~]# Meow xd_
```

El kernel se encuentra en la carpeta /boot (arranque) y la version es 2.6.32-279.el6.i686.

5. Muchas de las opciones de la línea del kernel se aplican una vez cargado el kernel se procesan por el RAM disk, mediante dracut. Consultar la información y opciones de este sistema en:

<http://www.kernel.org/pub/linux/utils/boot/dracut/dracut.html#dracutcmdline7>

Es una herramienta para crear imágenes initramfs copiando herramientas y ficheros del sistema instalado y combinandolo con el framework dracut.

Dracut se caracteriza por crear la imagen con el menor tamaño posible, el único propósito de la imagen initramfs es obtener el rootfs montado con tal que pueda transicionar al rootfs real lo antes posible.

`init=<ruta al init real>`

Especifica la ruta del binario init que se iniciara después que initramfs haya finalizado.

`root=<ruta a la partición />`

Especifica el dispositivo que se montara como root /..

`rootfstype=<sistema de ficheros de la partición root>`

`rd.cmdline=ask`

El sistema pide al usuario por parámetros al kernel adicionales.

`rd.fstab=0`

NO usar los parámetros encontrados en el fichero `/etc/fstab` en la partición root.

6. GRUB dispone además de un modo comandos ('c') estudiar los comandos básicos: **find, cat, boot, root.**

Find: es para buscar si existe o no un determinado archivo en todas las particiones y devuelve las particiones en donde se encuentra.

Cat: Imprime el contenido del Fichero.

```

lists possible command completions. Anywhere else TAB lists the possible
completions of a device/filename. ESC at any time exits.]

grub> find /grub/menu.lst
(hd0,0)

grub> cat (hd0,0)/grub/grub.conf
# 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
#foreground = 16f44a
#background = f41676
default=0
timeout=15
splashimage=(hd0,0)/grub/splash.xpm.gz
hiddenmenu
title fapCentOS Meow xd (2.6.32-279.el6.i686)
    root (hd0,0)
    kernel /vmlinuz-2.6.32-279.el6.i686 ro root=UUID=cdf45fde-0c73-4337-9ca1
-62a88f2349b2 rd_NO_LUKS KEYBOARDTYPE=pc KEYTABLE=es rd_NO_MD SYSFONT=latarcyrh
eb-sun16 crashkernel=auto LANG=es_ES.UTF-8 rd_NO_LVM rd_NO_DM rhgb quiet
    initrd /initramfs-2.6.32-279.el6.i686.img
grub> Meow xd

```

Boot: Arranca el Sistema Operativo.

Root: Configura en cual partición vamos a levantar como /.

```

GNU GRUB version 0.97 (639K lower / 1047488K upper memory)

[ Minimal BASH-like line editing is supported. For the first word, TAB
lists possible command completions. Anywhere else TAB lists the possible
completions of a device/filename. ESC at any time exits.]

grub> root
(hd0,0): Filesystem type is ext2fs, partition type 0x83

grub> root (hd,)
Error 23: Error while parsing number

grub> root (hd0,0)

grub> boot

```

Actividad 3

Hacemos una parada aquí y vamos a ver la importancia que tiene el `initrd` para un sistema operativo. Como siempre vamos a trabajar haciendo una copia al archivo para no modificar el original del sistemas. Se va a indicar el proceso, la explicación de los comandos ya sabe que con un `man`, `info`, `apropos...` puede obtener más información :-)

1. Lo primero a realizar es localizar y descomprimir el fichero `initrd`. Decir que dicho fichero se encuentra en el directorio raíz (/)

1. `mkdir /home/carpPersonal/tmp`
2. `cp /boot/initramfs*.img ~/tmp`
3. `cd ~/tmp`
4. `mv initramfs*.img initrd.gz`
5. `gunzip initrd.gz`

```
[root@localhost ~]# mkdir tmp
[root@localhost ~]# cp /boot/initramfs*.img ~/tmp
[root@localhost ~]# cd tmp
[root@localhost tmp]# ls
initramfs-2.6.32-279.el6.i686.img
[root@localhost tmp]# mv initramfs*.img initrd.gz
[root@localhost tmp]# gunzip initrd.gz
[root@localhost tmp]# ls
initrd
[root@localhost tmp]# _
```

2. Una vez que tenemos el fichero en nuestro directorio procedemos a extraerlo:

1. `mkdir tmp2`
2. `cd tmp2`
3. `cpio -id < ../initrd`
4. `gunzip initrd.gz`

```
[root@localhost tmp]# mkdir tmp2
[root@localhost tmp]# cd tmp2/
[root@localhost tmp2]# cpio -id < ../initrd
68819 blocks
[root@localhost tmp2]# gunzip initrd.gz
gzip: initrd.gz: No such file or directory
[root@localhost tmp2]# ls
bin                emergency          initqueue-finished  mount              proc              tmp
cmdline            etc                initqueue-settled   pre-pivot          sbin              usr
dev                init               initqueue-timeout   pre-trigger        sys               var
dracut-004-283.el6 initqueue          lib                  pre-udev           sysroot
```

3. Realizamos un ls y observaremos que tenemos todas las carpetas y archivos de arranque. También veremos un fichero init, este es el proceso principal padre de todos los procesos. Realice un cat a dicho archivo y vea su contenido (tranquilo no llegamos a tanto en la asignatura solamente observe y admire el script del DIOS SUPREMO Y CREADOR DE LOS PROCESOS EN UNIX).

Haciendo ls:

```
[root@localhost tmp2]# ls
bin                emergency          initqueue-finished  mount              proc              tmp
cmdline            etc                initqueue-settled   pre-pivot          sbin              usr
dev                init               initqueue-timeout   pre-trigger        sys               var
dracut-004-283.el6 initqueue          lib                  pre-udev           sysroot
```

Haciendo cat init:

```
info "Switching root"

wait_for_loginit

if [ -f /etc/capsdrop ]; then
    . /etc/capsdrop
    info "Calling $INIT with capabilities $CAPS_INIT_DROP dropped."
    exec capsh --drop="$CAPS_INIT_DROP" -- -c "exec switch_root \"$NEWROOT\" \"$
INIT\" $initargs" || {
        warn "Command:"
        warn capsh --drop=$CAPS_INIT_DROP -- -c "'exec switch_root \"$NEWROOT\" \"$
INIT\" $initargs'"
        warn "failed."
        emergency_shell
    }
else
    exec switch_root "$NEWROOT" "$INIT" $initargs || {
        warn "Something went very badly wrong in the initramfs. Please "
        warn "file a bug against dracut."
        emergency_shell
    }
fi
# vim:ts=8:sw=4:sts=4:et

[root@localhost tmp2]# _
```

Actividad 4

1. Con la ayuda del comando `dmesg` (usar una tubería a `less` y el comando de búsqueda) observar los mensajes que produce el kernel durante esta fase.

```
Initializing cgroup subsys cpuset
Initializing cgroup subsys cpu
Linux version 2.6.32-279.el6.i686 (mockbuild@fc6b9.bsys.dev.centos.org) (gcc version 4.4.6 20120305 (Red Hat 4.4.6-4) (GCC) ) #1 SMP Fri Jun 22 10:59:55 UTC 2012
KERNEL supported cpus:
  Intel GenuineIntel
  AMD AuthenticAMD
  NSC Geode by NSC
  Cyrix CyrixInstead
  Centaur CentaurHauls
  Transmeta GenuineTMx86
  Transmeta TransmetaCPU
  UMC UMC UMC UMC
BIOS-provided physical RAM map:
BIOS-e820: 0000000000000000 - 000000000009fc00 (usable)
BIOS-e820: 000000000009fc00 - 00000000000a0000 (reserved)
BIOS-e820: 00000000000f0000 - 0000000000100000 (reserved)
BIOS-e820: 0000000000100000 - 0000000003fff000 (usable)
BIOS-e820: 0000000003fff000 - 0000000004000000 (ACPI data)
BIOS-e820: 000000000fff0000 - 0000000010000000 (reserved)
```

```
Adding 1048568k swap on /dev/sda3. Priority:-1 extents:1 across:1048568k
SELinux: initialized (dev binfmt_misc, type binfmt_misc), uses genfs_contexts
readahead-disable-service: delaying service auditd
NET: Registered protocol family 10
lo: Disabled Privacy Extensions
ip6_tables: (C) 2000-2006 Netfilter Core Team
nf_conntrack version 0.5.0 (16107 buckets, 64428 max)
ip_tables: (C) 2000-2006 Netfilter Core Team
RPC: Registered named UNIX socket transport module.
RPC: Registered udp transport module.
RPC: Registered tcp transport module.
RPC: Registered tcp NFSv4.1 backchannel transport module.
SELinux: initialized (dev rpc_pipefs, type rpc_pipefs), uses genfs_contexts
SELinux: initialized (dev autofs, type autofs), uses genfs_contexts
SELinux: initialized (dev autofs, type autofs), uses genfs_contexts
SELinux: initialized (dev autofs, type autofs), uses genfs_contexts
Bridge firewalling registered
type=1305 audit(1452804652.398:33587): auid=4294967295 ses=4294967295 subj=system_u:system_r:readahead_t:s0 op="remove rule" key=(null) list=4 res=1
type=1305 audit(1452804652.398:33588): audit_enabled=0 old=1 auid=4294967295 ses=4294967295 subj=system_u:system_r:readahead_t:s0 res=1
readahead-collector: starting delayed service auditd
readahead-collector: sorting
readahead-collector: finished
[root@localhost ~]#
```

1. A continuación se pasa el control al proceso init (el primer proceso del sistema con PID=1) que a su vez pasa el control al script /etc/rc.d/rc.sysinit. Explorar dicho archivo y describir su funcionalidad.

```
#!/bin/bash
#
# /etc/rc.d/rc.sysinit - run once at boot time
#
# Taken in part from Miquel van Smoorenburg's bcheckrc.
#
HOSTNAME=$(/bin/hostname)

set -m

if [ -f /etc/sysconfig/network ]; then
    . /etc/sysconfig/network
fi
if [ -z "$HOSTNAME" -o "$HOSTNAME" = "(none)" ]; then
    HOSTNAME=localhost
fi

if [ ! -e /proc/mounts ]; then
```

```
    /usr/sbin/system-config-network-cmd --profile $arg##netprofile=$
    fi
done
fi
fi

# Now that we have all of our basic modules loaded and the kernel going,
# let's dump the syslog ring somewhere so we can find it later
[ -f /var/log/dmesg ] && mv -f /var/log/dmesg /var/log/dmesg.old
dmesg -s 131072 > /var/log/dmesg

# create the crash indicator flag to warn on crashes, offer fsck with timeout
touch /.autofsck &> /dev/null

[ "$$PROMPT" != no ] && plymouth --ignore-keystroke=li
if strstr "$cmdline" confirm ; then
    touch /var/run/confirm
fi

# Let rhgb know that we're leaving rc.sysinit
if [ -x /bin/plymouth ]; then
    /bin/plymouth --sysinit
fi

[root@localhost rc.d]# _
```

Establece las rutas de entorno, levanta la swap, verifica los sistemas de ficheros y se encarga de todos los requerimientos de inicialización del sistema.

2. Determinar la ubicación de init y consultar la información referente a init. ¿Qué tipo de archivo es? Consultar el contenido del fichero /etc/rc.d/rc.sysinit e identificar las partes importantes del programa (repasar si es necesario los conceptos básicos de programación en shell). Una vez configurados los servicios básicos por rc.sysinit, se determina el nivel de arranque y se arrancan los servicios específicos de cada nivel. El nivel de arranque por defecto se determina en el fichero /etc/inittab. En el apartado anterior hemos visto como desde GRUB se puede también seleccionar el nivel de arranque para reparar el sistema.

Estudiar este fichero.

```
[root@localhost ~]# whereis init
init: /sbin/init /etc/init.d /etc/init /usr/share/man/man8/init.8.gz /usr/share/
man/man5/init.5.gz
[root@localhost ~]# which init
/sbin/init
[root@localhost ~]# Meow xd_
```

init es una daemon que funciona hasta que el sistema se apaga. es el padre directo o indirecto de todos los procesos y automáticamente adopta a todos los procesos hijos.

Haciendo nano /etc/rc.d/rc.sysinit obtenemos:

Se observa que en el fichero /etc/rc.d/rc.sysinit primero establece el nombre de nuestro host y algunas propiedades de red local, luego monta la partición virtual de proc y sysfs, usb.

También establece las propiedades de inicio de SELINUX para luego imprimir el mensaje de “Welcome to CENTOS” que aparece primero en la pantalla.

```
GNU nano 2.8.9          Fichero: /etc/rc.d/rc.sysinit
#!/bin/bash
#
# /etc/rc.d/rc.sysinit - run once at boot time
#
# Taken in part from Miquel van Smoorenburg's bcheckrc.
#
HOSTNAME=$(/bin/hostname)
set -m
if [ -f /etc/sysconfig/network ]; then
    . /etc/sysconfig/network
fi
if [ -z "$HOSTNAME" -o "$HOSTNAME" = "(none)" ]; then
    HOSTNAME=localhost
fi
if [ ! -e /proc/mounts ]; then
    mount -n -t proc /proc /proc
^G Ver ayuda  ^O Guardar  ^R Leer Fich ^Y Pág Ant  ^K CortarTxt ^C Pos actual
^X Salir      ^J Justificar ^W Buscar    ^U Pág Sig  ^U PegarTxt  ^T Ortografía
```

Haciendo nano /etc/inittab :

```
GNU nano 2.0.9          Fichero: inittab

# 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:3:initdefault:

^G Ver ayuda  ^O Guardar   ^R Leer Fich ^Y Pág Ant   ^K CortarTxt ^C Pos actual
^X Salir      ^J Justificar^W Buscar    ^U Pág Sig   ^U PegarTxt  ^T Ortografía
```

3. Determinar los niveles de arranque disponibles en el sistema (consultar el fichero /etc/inittab). Cambiar el nivel por defecto del sistema al mono usuario y reiniciar el sistema.

Del anterior se tiene:

```
GNU nano 2.0.9          Fichero: inittab

# 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:3:initdefault:

^G Ver ayuda  ^O Guardar   ^R Leer Fich ^Y Pág Ant   ^K CortarTxt ^C Pos actual
^X Salir      ^J Justificar^W Buscar    ^U Pág Sig   ^U PegarTxt  ^T Ortografía
```

los niveles de arranque son:

- 0: apagar
- 1: single mode... para iniciar sin contraseña
- 2: Modo multiusuario, hay conexión entre todos los usuarios sin NFS.
- 3: full multiuser, al iniciar, te pide que te logees con user y password
- 4: modo suspendido (sin usar).
- 5: X11: inicia en modo con interfaz gráfica.
- 6: reboot, reinicia el sistema.

Cambiando al modo 1. (single o mono), esto se obtiene al reiniciar (no pide contraseña).

```
Informando a INIT para ir a modo monousuario.
[root@localhost ~]# Meow xd _
```

4. El nivel de ejecución se puede determinar con runlevel (comprobar que estamos en el nivel 1). Además podemos pasar de un nivel a otro con el comando init. Pasar al nivel multiusuario y observar los servicios que se arrancan. Finalmente devolver el nivel de ejecución por defecto al 3 y reiniciar, esta vez con init.

Viendo el nivel:

```
[root@localhost ~]# runlevel
1 S
[root@localhost ~]# _
```

al hacer init 3

```

Informando a INIT para ir a modo monousuario.
init: re main process (1099) killed by TERM signal
[root@localhost ~]# init 3
[root@localhost ~]# Calling the system activity data collector (sadc):
iptables: aplicando las reglas del cortafuegos: [ OK ]
iptables: aplicando reglas del cortafuegos: [ OK ]
Activación de la interfaz de loopback: [ OK ]
Iniciando auditd: [ OK ]
Iniciando portreserve: [ OK ]
Iniciando gestor de registro del sistema: [ OK ]
Iniciando irqbalance: [ OK ]
Iniciando rpcbind: [ OK ]
Starting kdump: ( failed )
Iniciando bus de mensajes del sistema: [ OK ]
Iniciando cups: [ OK ]
Montando otros sistemas de archivos: [ OK ]
Iniciación del demonio acpi: [ OK ]
Iniciando el demonio HAL: [ OK ]
Relanzar los eventos de udev fallidos [ OK ]
_

```

despues de ejecutar init 3

```
CentOS release 6.3 (Final)
Kernel 2.6.32-279.el6.i686 on an i686

localhost login: _
```

5. Además de init, reboot podemos usar el comando poweroff y shutdown para apagar el servidor. Consultar la página de manual de shutdown y reiniciar la máquina con un periodo de gracia de 1 minuto.

poweroff: apaga la maquina.

reboot: reinicia el sistema.

Shutdown: apaga el sistema segun el parámetro que le des.

```
[root@localhost ~]# poweroff _
```

```
[root@localhost ~]# reboot _
```

6. Busque información sobre los niveles de arranque del 2 al 5 y exponga brevemente cada uno.

Nivel 2 : Modo multiusuario sin servicio de red.

Nivel 3 : Modo multiusuario con soporte de red.

Nivel 4 : No tiene una propiedad en especial, se usa por el administrador para cargar algún servicio.

Nivel 5 : Es como el nivel 3 pero con interfaz gráfica.

Actividad 5

1. Busque información sobre que es el proceso swapper.

Cuando carga el sistema, el kernel se copia en la RAM y se ejecuta, este inicia el hardware y crea un proceso de sistema, el proceso 0 conocido como swapper y este proceso crea el proceso madre init (proceso 1).

Luego de bifurcarse en el proceso init, swapper se vuelve un proceso ocioso.

Actividad 6

1. Los archivos de configuración de upstart están en /etc/init. La mayor parte de los nombres son auto-explicativos. Cambiar el funcionamiento de control-alt-delete.conf para que para que cambie al nivel 1 en lugar de reiniciar el sistema. Probar el funcionamiento y volver al nivel 3.

NOTA: Para mandar la señal acceder al menú de VirtualBox.

```
GNU nano 2.0.9      Fichero: 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"
exec /sbin/init 1
```

Después de ejecutar la señal con VirtualBox, sucede lo siguiente.

```
Informando a INIT para ir a modo monousuario.
init: rc main process (973) killed by TERM signal
[root@localhost ~]# _
```

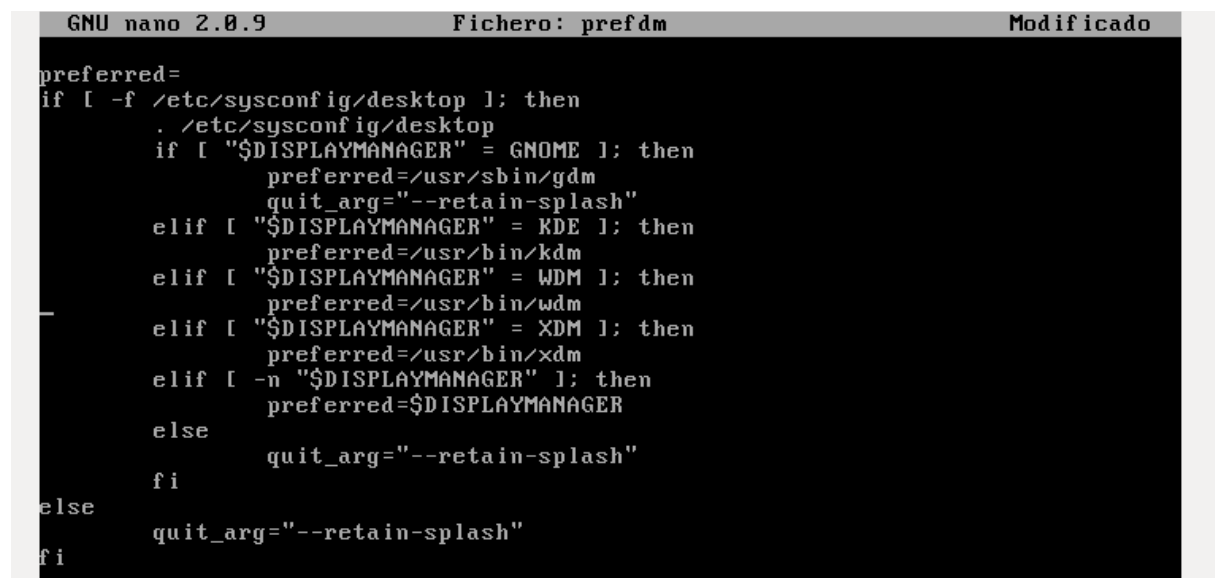
Luego de ejecutar init 2

```
CentOS release 6.3 (Final)
Kernel 2.6.32-279.el6.i686 on an i686

localhost login: _
```

2. Linux tiene terminales gráficos y serie, que se configuran en el arranque. Los archivos que controlan los terminales son `prefdm.conf` (arranca y mantiene el login gráfico), `start-ttys.conf` y `getty.conf`:

1. Consultar el contenido de `prefdm.conf`. Interpretar el contenido de `/etc/X11/prefdm`. ¿En qué nivel se arranca el sistema gráfico?

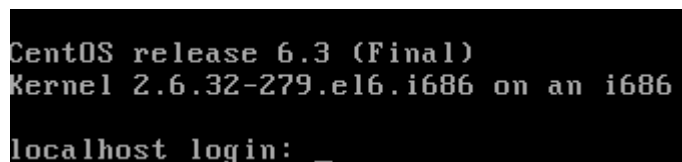


```
GNU nano 2.0.9          Fichero: prefdm          Modificado
preferred=
if [ -f /etc/sysconfig/desktop ]; then
    . /etc/sysconfig/desktop
    if [ "$DISPLAYMANAGER" = GNOME ]; then
        preferred=/usr/sbin/gdm
        quit_arg="--retain-splash"
    elif [ "$DISPLAYMANAGER" = KDE ]; then
        preferred=/usr/bin/kdm
    elif [ "$DISPLAYMANAGER" = WDM ]; then
        preferred=/usr/bin/wdm
    elif [ "$DISPLAYMANAGER" = XDM ]; then
        preferred=/usr/bin/xdm
    elif [ -n "$DISPLAYMANAGER" ]; then
        preferred=$DISPLAYMANAGER
    else
        quit_arg="--retain-splash"
    fi
else
    quit_arg="--retain-splash"
fi
```

Según el `/etc/inittab` el nivel 5 contiene el arranque X11.

El contenido del archivo, es que dependiendo de que parámetros tome, arranca una distinta interfaz gráfica, como KDM o WDM o XDM.

2. Probar los distintos terminales (CtrlDcho + F1, CtrlDcho + F2)



```
CentOS release 6.3 (Final)
Kernel 2.6.32-279.el6.i686 on an i686

localhost login: _
```

Se obtienen nuevas sesiones para hacer login.

Actividad 7

Haciendo el `ls -l` en `/etc/rc2.d`

```
lrwxrwxrwx. 1 root root 14 ene 14 10:46 K88sssd -> ../init.d/sssd
lrwxrwxrwx. 1 root root 15 ene 14 10:43 K89rdisc -> ../init.d/rdisc
lrwxrwxrwx. 1 root root 18 ene 14 10:43 K95cgconfig -> ../init.d/cgconfig
lrwxrwxrwx. 1 root root 14 ene 14 10:45 K99rngd -> ../init.d/rngd
lrwxrwxrwx. 1 root root 17 ene 14 10:45 S01sysstat -> ../init.d/sysstat
lrwxrwxrwx. 1 root root 22 ene 14 10:45 S02lvm2-monitor -> ../init.d/lvm2-monitor
lrwxrwxrwx. 1 root root 19 ene 14 10:42 S08ip6tables -> ../init.d/ip6tables
lrwxrwxrwx. 1 root root 18 ene 14 10:42 S08iptables -> ../init.d/iptables
lrwxrwxrwx. 1 root root 17 ene 14 10:43 S10network -> ../init.d/network
lrwxrwxrwx. 1 root root 16 ene 14 10:45 S11auditd -> ../init.d/auditd
lrwxrwxrwx. 1 root root 21 ene 14 10:42 S11portreserve -> ../init.d/portreserve
lrwxrwxrwx. 1 root root 17 ene 14 10:43 S12rsyslog -> ../init.d/rsyslog
lrwxrwxrwx. 1 root root 18 ene 14 10:45 S13cpuspeed -> ../init.d/cpuspeed
lrwxrwxrwx. 1 root root 17 ene 14 10:43 S13rpcbind -> ../init.d/rpcbind
lrwxrwxrwx. 1 root root 19 ene 14 10:43 S15mdmonitor -> ../init.d/mdmonitor
lrwxrwxrwx. 1 root root 20 ene 14 10:42 S22messagebus -> ../init.d/messagebus
lrwxrwxrwx. 1 root root 14 ene 14 10:44 S25cups -> ../init.d/cups
lrwxrwxrwx. 1 root root 15 ene 14 10:45 S26acpid -> ../init.d/acpid
lrwxrwxrwx. 1 root root 19 ene 14 10:43 S26udev-post -> ../init.d/udev-post
lrwxrwxrwx. 1 root root 14 ene 14 10:45 S55sshd -> ../init.d/sshd
lrwxrwxrwx. 1 root root 17 ene 14 10:43 S80postfix -> ../init.d/postfix
lrwxrwxrwx. 1 root root 15 ene 14 10:43 S90crond -> ../init.d/crond
lrwxrwxrwx. 1 root root 11 ene 14 10:43 S99local -> ../rc.local
```

1. Veamos un ejemplo que aclare el apagado de un servicio en el nivel de arranque. Para ellos siga los pasos descritos posteriormente comentando los comandos ejecutados y sus respectivas salidas.

```
[root@localhost etc]# service apache2 status
apache2: service desconocido
[root@localhost etc]# _
```

Como no tenemos el servicio apache, usaremos el servicio network para el ejercicio.

2. Veamos que el servicio apache2 está activado con S y funcionando con `service apache2 status`, comprobamos en el navegador con localhost que trabaja bien.

```
[root@localhost etc]# service apache2 status
apache2: service desconocido
[root@localhost etc]# service network status
Dispositivos configurados:
lo eth0
Dispositivos activos en el momento:
lo
[root@localhost etc]# meow xd_
```

Entonces, buscamos donde se encuentra el daemon network ...

```
[root@localhost etc]# ls rc3.d
K01numad      K75ntpdate      S08ip6tables    S15mdmonitor    S28autofs
K01smartd     K75quota_nld    S08iptables     S18rpcidmapd     S55sshd
K02oddjobd    K86cgred        S10network      S19rpcgssd       S80postfix
K10psacct     K87restorecond  S11auditd       S20kdump         S82abrt-ccpp
K10saslauthd  K88sssd         S11portreserve  S22messagebus   S82abrt-d
K50netconsole K89rdisc        S12rsyslog      S25cups          S82abrt-oops
K60nfs        K95cgconfig     S13cpuspeed     S25netfs         S90crond
K69rpcsvcgssd K99rngd         S13irqbalance   S26acpid         S95atd
K73ypbind     S01sysstat      S13rpcbind      S26haldaemon    S99certmonger
K74ntpd       S02lvm2-monitor S14nfslock      S26udev-post     S99local
[root@localhost etc]# _
```

Vemos que network esta iniciado con S10 y esta en la carpeta rc3.d indicando que esta en el nivel 3 de arranque.

3. Posteriormente vamos a desactivarlo mv S91apache2 K91apache2 y comprobamos en la carpeta que tiene la K de apagado y reiniciamos el sistema.

Siguiendo la guía ...

pero con network hacemos.

```
[root@localhost ~]# mv /etc/rc3.d/S1
S10network      S12rsyslog      S13rpcbind      S18rpcidmapd
S11auditd       S13cpuspeed     S14nfslock      S19rpcgssd
S11portreserve  S13irqbalance   S15mdmonitor
[root@localhost ~]# mv /etc/rc3.d/S10network /etc/rc3.d/K10
K10psacct      K10saslauthd
[root@localhost ~]# mv /etc/rc3.d/S10network /etc/rc3.d/K10network
[root@localhost ~]# init 6_
```

4. Una vez reiniciado comprobamos que se encuentra desactivado en rc2.d, comprobamos en el navegador que se encuentra desactivado dicho servicio y que no esta arrancado (service).

Ahora vemos el nuevo estado de network.

```
[root@localhost ~]# service network status
Dispositivos configurados:
lo eth0
Dispositivos activos en el momento:

[root@localhost ~]# _
```

Verificando con ifconfig

```
[root@localhost rc2.d]# ifconfig
[root@localhost rc2.d]# _
```


5. Para activar el servicio en el próximo arranque simplemente habrá que hacer la operación inversa.

```
K01certmonger K16abrt-oops K75quota_nld K95cgconfig S13rpcbind
K01numad K50netconsole K80kdump K99rngd S15mdmonitor
K01smartd K60nfs K85rpcgssd S01sysstat S22messagebus
K02odd.jobd K69rpcsvcgssd K85rpcidmapd S02lvm2-monitor S25cups
K05atd K72autofs K86cgred S08ip6tables S26acpid
K10network K73ypbind K86nfslock S08iptables S26udev-post
K10psacct K74haldaemon K87irqbalance S11auditd S55sshd
K10saslauthd K74ntpd K87restorecond S11portreserve S80postfix
K16abrt-ccpp K75netfs K88sssd S12rsyslog S90crond
K16abrt-d K75ntpdate K89rdisc S13cpuspeed S99local
[root@localhost rc2.d]# mv K10
K10network K10psacct K10saslauthd
[root@localhost rc2.d]# mv K10network S10network
[root@localhost rc2.d]# ls
K01certmonger K50netconsole K80kdump K99rngd S13rpcbind
K01numad K60nfs K85rpcgssd S01sysstat S15mdmonitor
K01smartd K69rpcsvcgssd K85rpcidmapd S02lvm2-monitor S22messagebus
K02odd.jobd K72autofs K86cgred S08ip6tables S25cups
K05atd K73ypbind K86nfslock S08iptables S26acpid
K10psacct K74haldaemon K87irqbalance S10network S26udev-post
K10saslauthd K74ntpd K87restorecond S11auditd S55sshd
K16abrt-ccpp K75netfs K88sssd S11portreserve S80postfix
K16abrt-d K75ntpdate K89rdisc S12rsyslog S90crond
K16abrt-oops K75quota_nld K95cgconfig S13cpuspeed S99local
[root@localhost rc2.d]# _
```

6. Te habrás dado cuenta que aunque le has activado con S el servicio apache no está funcionando (service). Explique detalladamente el por qué no se activa y cuando se activará dicho servicio.

Se activará luego de reiniciar el sistema, porque los tipos S se inician con el comando start.

7. Para no tener que reiniciar el servidor podemos activar el servicio ahora realizando un service apache2 start. Refresque el navegador en localhost y verá que ya funcionará o con service apache2 status. Como anécdota muchos troyanos, gusanos, virus... se activan desde el proceso init, es importante verificar dichos procesos en un sistema y comprobar que no hay ningún proceso sospechoso activado desde el inicio. Aquí empezamos nuestra monitorización de seguridad en servicios.

```
[root@localhost ~]# service network restart
Interrupción de la interfaz de loopback: [ OK ]
Activación de la interfaz de loopback: [ OK ]
[root@localhost ~]# service network status
Dispositivos configurados:
lo eth0
Dispositivos activos en el momento:
lo
[root@localhost ~]# _
```

Actividad 8

1. Los servicios de cada nivel de ejecución se ejecutan mediante el job upstart /etc/init/rc.conf. Consultar su contenido y establecer qué scripts se ejecutan en cada caso.

Usamos el comando less para ver un poco del contenido.

```
[root@localhost ~]# less /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
/etc/init/rc.conf (END)
```

Este script ejecuta los scripts que estan en nuestro nivel de ejecución (runlevel esta en 3, por lo que ejecuta los archivos deamon de la carpeta /etc/rc3.d).

**2. Consultar el contenido del directorio para el nivel 3 /etc/rc3.d/. ¿Qué tipos de ficheros hay? ¿Qué estructura tienen? Relacionarlos con el script /etc/rc.d/rc, especialmente las líneas:
for i in /etc/rc\$runlevel.d/K* ; do y for i in /etc/rc\$runlevel.d/S* ; do**

Los ficheros de /etc/rc3.d tiene definidos las funciones start, stop, reload y otras mas. Aparte, tiene definido su archivo de configuracion, incluyen la ruta de su binario en la carpeta /usr y además muestran la condición que si no eres root, no puedes ejecutar alguna acción sobre dichos deamons.

Estan los tipos K y S, donde K son ejecutados con Stop y S son ejecutados con start.

Los ficheros que muestra, son enlaces a scripts en el directorio /etc/init.d

```
[root@localhost init.d]# ls
abrt-ccpp      cpuspeed      killall        ntpd           rngd           sshd
abrt-d         crond         lvm2-lvmetad  ntpdate        rpcbind        sssd
abrt-oops      cups          lvm2-monitor  numad          rpcgssd        sysstat
acpid          functions     mdmonitor     oddjobd        rpcidmapd      udev-post
atd            haldaemon    messagebus    portreserve    rpcsvcgssd     yppbind
auditd         halt         netconsole    postfix        rsyslog
autofs         iptables     netfs         psacct         sandbox
certmonger     ip6tables    network       quota_nld      saslauthd
cgconfig       irqbalance   nfs           rdisc          single
cgred          kdump        nfslock       restorecond    smartd

[root@localhost init.d]# ls ../rc3.d
K01numad      K75ntpdate    S08ip6tables  S15mdmonitor   S28autofs
K01smartd     K75quota_nld S08iptables   S18rpcidmapd   S55sshd
K02oddjobd    K86cgred      S10network    S19rpcgssd     S80postfix
K10psacct     K87restorecond S11auditd     S20kdump       S82abrt-ccpp
K10saslauthd  K88sssd       S11portreserve S22messagebus  S82abrt-d
K50netconsole K89rdisc      S12rsyslog    S25cups        S82abrt-oops
K60nfs        K95cgconfig   S13cpuspeed   S25netfs       S90crond
K69rpcsvcgssd K99rngd       S13irqbalance S26acpid       S95atd
K73yppbind    S01sysstat    S13rpcbind    S26haldaemon   S99certmonger
K74ntpd       S02lvm2-monitor S14nfslock    S26udev-post   S99local

[root@localhost init.d]# _
```

Se observa en el fichero rc, que la línea **for i in /etc/rc\$runlevel.d/K*** en el fichero /etc/rc.d/rc detiene los servicios de cuyos enlaces comiencen con la letra K

```
GNU nano 2.0.9          Fichero: rc          Modificado

for i in /etc/rc$runlevel.d/K* ; do

    # Check if the subsystem is already up.
    subsys=${i#/etc/rc$runlevel.d/K??}
    [ -f /var/lock/subsys/$subsys -o -f /var/lock/subsys/$subsys.init ] || $
    check_runlevel "$i" || continue

    # Bring the subsystem down.
    [ -n "$UPSTART" ] && initctl emit --quiet stopping JOB=$subsys
    $i stop
    [ -n "$UPSTART" ] && initctl emit --quiet stopped JOB=$subsys

done
```

y la línea de **for i in /etc/rc\$runlevel.d/S*** inicia los scripts o evalúa si están trabajando y si no, los activa reiniciando.

```
# Now run the START scripts.
for i in /etc/rc$runlevel.d/S* ; do

    # Check if the subsystem is already up.
    subsys=${i#/etc/rc$runlevel.d/S??}
    [ -f /var/lock/subsys/$subsys ] && continue
    [ -f /var/lock/subsys/$subsys.init ] && continue
    check_runlevel "$i" ;; continue

    # If we're in confirmation mode, get user confirmation
    if [ "$do_confirm" = "yes" ]; then
        confirm $subsys
        rc=$?
        if [ "$rc" = "1" ]; then
            continue
        elif [ "$rc" = "2" ]; then
            do_confirm="no"
        fi
    fi

    update_boot_stage "$subsys"
    # Bring the subsystem up.
    [ -n "$UPSTART" ] && initctl emit --quiet starting JOB=$subsys
    if [ "$subsys" = "halt" -o "$subsys" = "reboot" ]; then
        export LC_ALL=C
        exec $i start
    fi
done
```

3. Estudiar la estructura de un script de arranque, por ejemplo sshd (/etc/init.d/sshd). Especialmente la parte (case "\$1" in). Cada servicio se puede arrancar, parar o reiniciar usando estos scripts. Detener el demonio ssh y volver a arrancarlo usando el script /etc/init.d/sshd.

Observamos que como un script típico, inicia seteando las variables locales de entorno y las funciones que utilizará.

Este script inicia el daemon de SSH, Viendo su estructura. ...

```

case "$1" in
    start)
        rh_status_q && exit 0
        start
        ;;
    stop)
        if ! rh_status_q; then
            rm -f $lockfile
            exit 0
        fi
        stop
        ;;
    restart)
        restart
        ;;
    reload)
        rh_status_q || exit 7
        reload
        ;;
    force-reload)
        ;;
    *)
        _

```

Se observa en este trozo del script que recibe parametros denominados start, stop, restart, reload, etc. tal que para cada uno realiza una operacion.

```

[root@localhost ~]# service sshd stop
Parando sshd: [ OK ]
[root@localhost ~]# /etc/init.d/sshd start
Iniciando sshd: [ OK ]
[root@localhost ~]# _

```

Se detiene el servicio ssh de la manera convencional y se vuelve a iniciar.

4. Alternativamente se puede usar el comando service (man service) para controlar los servicios (comandos: start, stop, restart, status... todas las opciones del case). Consultar además el estado de todos los servicios del nivel con la opción --status-all.

Haciendo service --status-all nos muestra todo hasta el final, entonces para ver las primeras 5 lineas hacemos. Service --status-all | head.

```

[root@localhost ~]# service --status-all | head
Se está ejecutando auditd (pid 766)...
Se está ejecutando crond (pid 902)...
Tabla: filter
Chain INPUT (policy ACCEPT)
num target      prot opt source                destination            state RELATED,
ESTABLISHED
2    ACCEPT      icmpv6  ::/0                  ::/0
3    ACCEPT      all     ::/0                  ::/0
4    ACCEPT      tcp     ::/0                  ::/0                  state NEW tcp
dpt:22
5    REJECT      all     ::/0                  ::/0                  reject-with ic
mp6-adm-prohibited
[root@localhost ~]# _

```

En esta parte de la salida se ve que los servicios cron, audit e iptables6 están iniciados.

Y al final nos muestra un estado de los respectivos scripts, si se estan ejecutando o estan detenidos.

```

nfsd está parado
rpc.rquotad está parado
Se está ejecutando rpc.statd (pid 1067)...
nptd está parado
numad está parado
oddjobd está parado
portreserve está parado
Se está ejecutando master (pid 1335)...
Cuenta de procesos desactivada.
quota_nld está parado
rdisc está parado
restorecond está parado
rngd está parado
Se está ejecutando rpcbind (pid 1049)...
rpc.gssd está parado
Se está ejecutando rpc.idmapd (pid 1099)...
rpc.svcgssd está parado
Se está ejecutando rsyslogd (pid 1007)...
sandbox is stopped
saslauthd está parado
smartd está parado
Se está ejecutando openssh-daemon (pid 1259)...
sssd está parado
ypbind está parado
[root@localhost init.d]# _

```

5. Los servicios que se arrancan en cada nivel se controlan con la orden chkconfig (man chkconfig)

1. Determinar los servicios y niveles ejecutando el comando chkconfig.

Haciendo chkconfig en la consola:

```

auditd      0:desactivado  1:desactivado  2:activo      3:activo
4:activo    5:activo      6:desactivado
crond       0:desactivado  1:desactivado  2:activo      3:activo
4:activo    5:activo      6:desactivado
ip6tables   0:desactivado  1:desactivado  2:activo      3:activo
4:activo    5:activo      6:desactivado
iptables    0:desactivado  1:desactivado  2:activo      3:activo
4:activo    5:activo      6:desactivado
netconsole  0:desactivado  1:desactivado  2:desactivado 3:desactivado
4:desactivado 5:desactivado 6:desactivado
netfs       0:desactivado  1:desactivado  2:desactivado 3:activo
4:activo    5:activo      6:desactivado
network     0:desactivado  1:desactivado  2:activo      3:activo
4:activo    5:activo      6:desactivado
postfix     0:desactivado  1:desactivado  2:activo      3:activo
4:activo    5:activo      6:desactivado
rdisc       0:desactivado  1:desactivado  2:desactivado 3:desactivado
4:desactivado 5:desactivado 6:desactivado
restorecond 0:desactivado  1:desactivado  2:desactivado 3:desactivado
4:desactivado 5:desactivado 6:desactivado
rsyslog     0:desactivado  1:desactivado  2:activo      3:activo
4:activo    5:activo      6:desactivado
ssldhd      0:desactivado  1:desactivado  2:desactivado 3:desactivado
4:desactivado 5:desactivado 6:desactivado
:_

```

Nos muestra todos los estados de los servicios y en que nivel se encuentra cada uno.

2. Determinar en qué niveles se ejecuta el servicio sshd (--list).

```

[root@localhost init.d]# chkconfig --list sshd
sshd      0:desactivado  1:desactivado  2:activo      3:activo      4:
:activo 5:activo      6:desactivado
[root@localhost init.d]# _

```

Vemos que esta activo en los niveles 2, 3, 4 y 5.

3. Desactivar el servicio sshd para el nivel 3 (--level 3). Cambiar al nivel 1, volver al 3 y comprobar que no está en ejecución (service).

```

[root@localhost ~]# chkconfig --level 3 sshd off
[root@localhost ~]# chkconfig --list sshd
sshd      0:desactivado  1:desactivado  2:activo      3:desactivado 4:
:activo 5:activo      6:desactivado
[root@localhost ~]# _

```

Hacemos init 1

```

Informando a INIT para ir a modo monousuario.
init: rc main process (1424) killed by TERM signal
[root@localhost ~]# _

```

Vemos que esta detenido en el nivel 3, porque lo detuvimos anteriormente.

```
[root@localhost ~]# runlevel
$ 3
[root@localhost ~]# service sshd status
openssh-daemon está parado
[root@localhost ~]# _
```

6. (Ejercicio obligatorio) Escribir un script de arranque para un demonio propio (/etc/init.d/mi_demonio.sh):

1. Usar la plantilla que se muestra a continuación.

```
#!/bin/sh
#
#chkconfig: <niveles de ejecución por defecto><orden arranquet> <orden
parada>
#description:
case "$1" in
    start)
        <ESCRIBIR EL FICHERO AQUI>
        exit 0
        ;;
    stop)
        <BORRAR EL FICHERO AQUI>
        exit 0
        ;;
    *)
        echo $"Usage: $0 {start|stop|}"
        exit 2
esac
```

2. El script debe escribir en el fichero /tmp/banner la fecha actual (comando date) y la cadena “Esto es un script de arranque” al arrancar (start).

3. Al parar debe borrar ese fichero (stop).

Le dare nivel de ejecucion, orden de arranque y orden parada como 345 95 06


```
#!/bin/sh
#chkconfig: 345 95 06
#description: script de ejemplo

case $1 in
    start)
        date > /tmp/banner
        echo "Esto es un script de arranque xd" >> /tmp/banner
        exit 0
        ;;
    stop)
        rm /tmp/banner
        exit 0
        ;;
    *)
        echo $"Debe usar: $0 {start|stop!}"
        exit 2
esac

[ 18 líneas escritas ]

[root@localhost ~]#
```

4. Comprobar que funciona correctamente ejecutándolo directamente. El archivo debe tener permisos de ejecución (chmod +x /etc/init.d/mi_demonio.sh).

Dandole permisos de ejecución

```
[root@localhost ~]# ls /etc/init.d/
abrt-ccpp  cpuspeed  killall    nfslock    restorecond  smartd
abrttd     crond      lvm2-lvmetad  ntpd       rngd          sshd
abrt-oops  cups       lvm2-monitor  ntpdate    rpcbind       sssd
acpid      functions  mdmonitor    numad      rpcgssd       sysstat
atd        haldaemon  messagebus   oddjobd    rpcidmapd     udev-post
auditd     halt       mi_demonio.sh portreserve rpcsvcgssd    ypbind
autofs     ip6tables  netconsole   postfix    rsyslog
certmonger iptables   netfs        psacct     sandbox
cgconfig   irqbalance network      quota_nld  saslauthd
cgred      kdump      nfs          rdisc      single

[root@localhost ~]# chmod +x /etc/init.d/mi_demonio.sh
[root@localhost ~]# ls /etc/init.d/
abrt-ccpp  cpuspeed  killall    nfslock    restorecond  smartd
abrttd     crond      lvm2-lvmetad  ntpd       rngd          sshd
abrt-oops  cups       lvm2-monitor  ntpdate    rpcbind       sssd
acpid      functions  mdmonitor    numad      rpcgssd       sysstat
atd        haldaemon  messagebus   oddjobd    rpcidmapd     udev-post
auditd     halt       mi_demonio.sh portreserve rpcsvcgssd    ypbind
autofs     ip6tables  netconsole   postfix    rsyslog
certmonger iptables   netfs        psacct     sandbox
cgconfig   irqbalance network      quota_nld  saslauthd
cgred      kdump      nfs          rdisc      single

[root@localhost ~]#
```

Verificando la funcionalidad

```
[root@localhost ~]# /etc/init.d/mi_demonio.sh start
[root@localhost ~]# cat /tmp/banner
vie ene 15 12:28:05 PET 2016
Esto es un script de arranque xd
[root@localhost ~]# /etc/init.d/mi_demonio.sh stop
[root@localhost ~]# cat /tmp/banner
cat: /tmp/banner: No existe el fichero o el directorio
[root@localhost ~]# ya no existe el fichero xd_
```

5. Añadir el nuevo servicio con chkconfig --add

6. Comprobar el nuevo servicio con chkconfig y consultando el contenido de /etc/rc3.d.

```
fapCentOS [Corriendo] - Oracle VM VirtualBox
[root@localhost ~]# chkconfig --add /etc/init.d/mi_demonio.sh
[root@localhost ~]# chkconfig --list mi_demonio.sh
mi_demonio.sh 0:desactivado 1:desactivado 2:desactivado 3:activo 4:activo 5:activo 6:desactivado
[root@localhost ~]# ls /etc/rc3.d/
K01numad      K75quota_nld      S10network      S20kdump      S82abrt-d
K01smartd     K86cgred          S11auditd       S22messagebus S82abrt-oops
K02oddjobd    K87restorecond   S11portreserve  S25cups       S90crond
K10psacct     K88sssd           S12rsyslog      S25netfs      S95atd
K10saslauthd  K89rdisc          S13cpuspeed     S26acpid      S95mi_demonio.sh
K50netconsole K95cgconfig       S13irqbalance  S26haldaemon  S99certmonger
K60nfs        K99rngd           S13rpcbind      S26udev-post  S99local
K69rpcsvcgssd S01sysstat        S14nfslock      S28autofs
K73ypbind     S02lvm2-monitor  S15mdmonitor    S55sshd
K74ntpd       S08ip6tables     S18rpcidmapd    S80postfix
K75ntpd       S08iptables      S19rpcgssd      S82abrt-ccpp
[root@localhost ~]# _
```

Se observa que tiene niveles de ejecución 3,4 y 5, y además su orden de arranque es S95.

7. Cambiar de nivel para ver que se crea y borra el fichero correctamente.

Ejecutando en nivel 1 con init 1.

```
Informando a INIT para ir a modo monousuario.
init: rc main process (2043) killed by TERM signal
[root@localhost ~]# cat /tmp/banner
cat: /tmp/banner: No existe el fichero o el directorio
[root@localhost ~]# _
```

Ejecutando en nivel 3 con init 3.

```
fapCentOS [Corriendo] - Oracle VM VirtualBox
CentOS release 6.3 (Final)
Kernel 2.6.32-279.el6.i686 on an i686

localhost login: root
Password:
Last login: Fri Jan 15 12:19:36 on tty1
[root@localhost ~]# cat /tmp/banner
vie ene 15 12:38:49 PET 2016
Esto es un script de arranque xd
[root@localhost ~]# _
```

7. Usar la herramienta ntsysv para ajustar los servicios en el nivel de ejecución.

La herramienta ntsysv permite modificar a qué niveles de ejecución van a arrancar los servicios como chkconfig.

La utilidad ntsysv proporciona una interfaz sencilla para activar o desactivar servicios. Usted puede usar ntsysv para activar un servicio xinetd gestionando el encendido o apagado.

También puede usar ntsysv para configurar los niveles de ejecución. De forma predeterminada, sólo el nivel de ejecución actual se configura. Para configurar un nivel de ejecución diferente, especifique uno o más niveles de ejecución con la opción --level. Por ejemplo, el comando ntsysv --level 345 configura los niveles de ejecución 3, 4 y 5.

Para ingresar, basta escribir en consola ntsysv:



Buscamos en la lista nuestro demonio

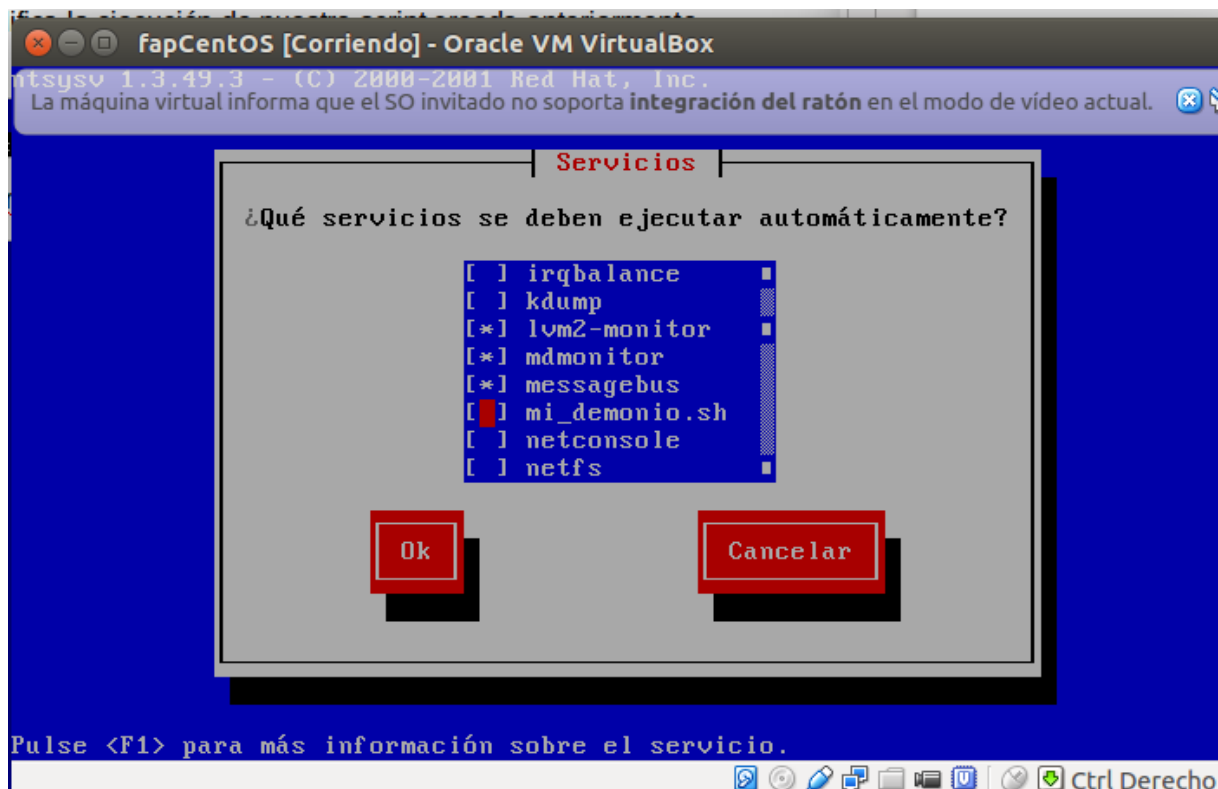
```
[ ] iptables  
[ ] mi_demonio.sh
```

y dame en la opción que se ejecute automáticamente (OK)

Ahora vamos a otro nivel usando:

```
[root@localhost ~]# ntsysv --level 2
```

Vemos que nuestro demonio en el nivel 2 está desactivado, (porque solo se arranca en nivel 3,4 y 5).



Si pulsamos el botón ok (con la tecla tab llegamos al botón), modificaremos la ejecución del servicio en el nivel 2.

En resumen, se ve que también se puede modificar bajo varios niveles a la vez (con --level 234 por ejemplo), pero procura no hacerlo a no ser que sepa lo que hace porque los cambios se sobrescriban bajo todos los niveles escritos.

Referencias:

https://dracut.wiki.kernel.org/index.php/Main_Page
<http://www.cyberciti.biz/faq/howto-display-all-installed-linux-kernel-version/>
<https://bbs.archlinux.org/viewtopic.php?id=81946>
https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-services-ntsysv.html
<http://ibiblio.org/pub/Linux/docs/LuCaS/Manuales-LuCAS/LIPP2/lipp-2.0-beta-html/node414.html>
https://es.wikipedia.org/wiki/Preboot_Execution_Environment
https://docs.oracle.com/cd/E37670_01/E41137/html/ch03s01s01.html
https://en.wikipedia.org/wiki/Parent_process
https://www.centos.org/docs/5/html/Deployment_Guide-en-US/s1-services-ntsysv.html