

Práctica GNU/Linux

Colegio Sant Josep Obrer
Implantación de Sistemas Operativos
ASIX 1
Piqueras Sastre Alejandro
Daniel Rosselló Sánchez

Índice de contenido

Contenido.....	3
Pregunta 1.....	3
Pregunta 2.....	4
1.-Iniciando la iso de Archlinux.....	4
2.-Distribución del teclado.....	5
3.-Particionar el disco.....	5
4.-Formatear las particiones.....	6
5.-Montar particiones.....	6
6.-Seleccionar el servidor de réplica.....	7
7.-Instalar los paquetes del sistema base.....	7
8.-Configurar el sistema.....	8
9.-Instalar un gestor de arranque.....	10
10.-Reiniciar.....	12
Pregunta 3.....	13
1.-Disponibilidad del usuario root tras la instalación del s.o.....	13
2.-Repaso de las órdenes vistas en clase para revisar funcionalidades en su momento descartadas al precisar de privilegios de usuario root para su realización. P.e: edición de la fecha y hora del sistema, orden chown,.....	13
3.-Documentar la orden shutdown.....	15
Pregunta 4.....	17
1.-Documentar el proceso de configuración del sistema de arranque Linux.....	17
2.-Documentar el proceso de configuración del sistema de arranque Microsoft.....	19
Pregunta 5.....	21
1.-Configurar el X Window System (Entorno gráfico básico).....	21
2.-Instalar Openbox.....	22
3.-Configuración de openbox.....	24
4.-Comparación con Windows.....	30
Pregunta 6.....	33
1.-Localización y descripción de vmlinuz, init, inittab y estructura de directorio y scripts relacionados con runlevels.....	33
2.-Orden runlevel.....	34
Pregunta 7.....	35
Pregunta 8.....	38
1.-Gestión de usuarios.....	38
2.-Gestión de grupos.....	39
3.-Desde entorno gráfico.....	40
Pregunta 9.....	42
1.-/etc/passwd.....	42
2.-/etc/group.....	43
3.-/etc/shadow.....	44
4.-/etc/gpasswd.....	44
Pregunta 10.....	46
1.-Usuario iso2015.....	46
2.-Grupo isogroup2015.....	47
Pregunta 11.....	48

1.-Comprobar existencia de /etc/skel.....	48
2.-Comprobación de su funcionamiento.....	48
Pregunta 12.....	49
1.-su.....	49
2.-sudo.....	49
Pregunta 13.....	52
1.-wall.....	52
2.-Message of the day.....	53
Pregunta 14.....	54
Pregunta 15.....	56
1.-mount.....	56
2.-umount.....	56
3.-mkfs.....	57
4.-df.....	58
5.-fsck.....	58
6.-e2fsck.....	60
7.-dumpe2fs.....	60
Pregunta 16.....	62
1.-ext.....	62
2.-ext2.....	62
3.-ext3.....	62
4.-ext4.....	63
Problemas encontrados.....	64
Opinión personal.....	65
Alejandro Piqueras Sastre.....	65
Daniel Rosselló Sanchez.....	65
Puntos de discusión.....	65
Bibliografía.....	66

Contenido

Pregunta 1

Elegir una distribución Linux y realizar una instalación de Linux de forma que convivan en el mismo ordenador dos sistemas operativos (1.-Microsoft Windows y 2.-Linux, los números indican el orden de instalación)

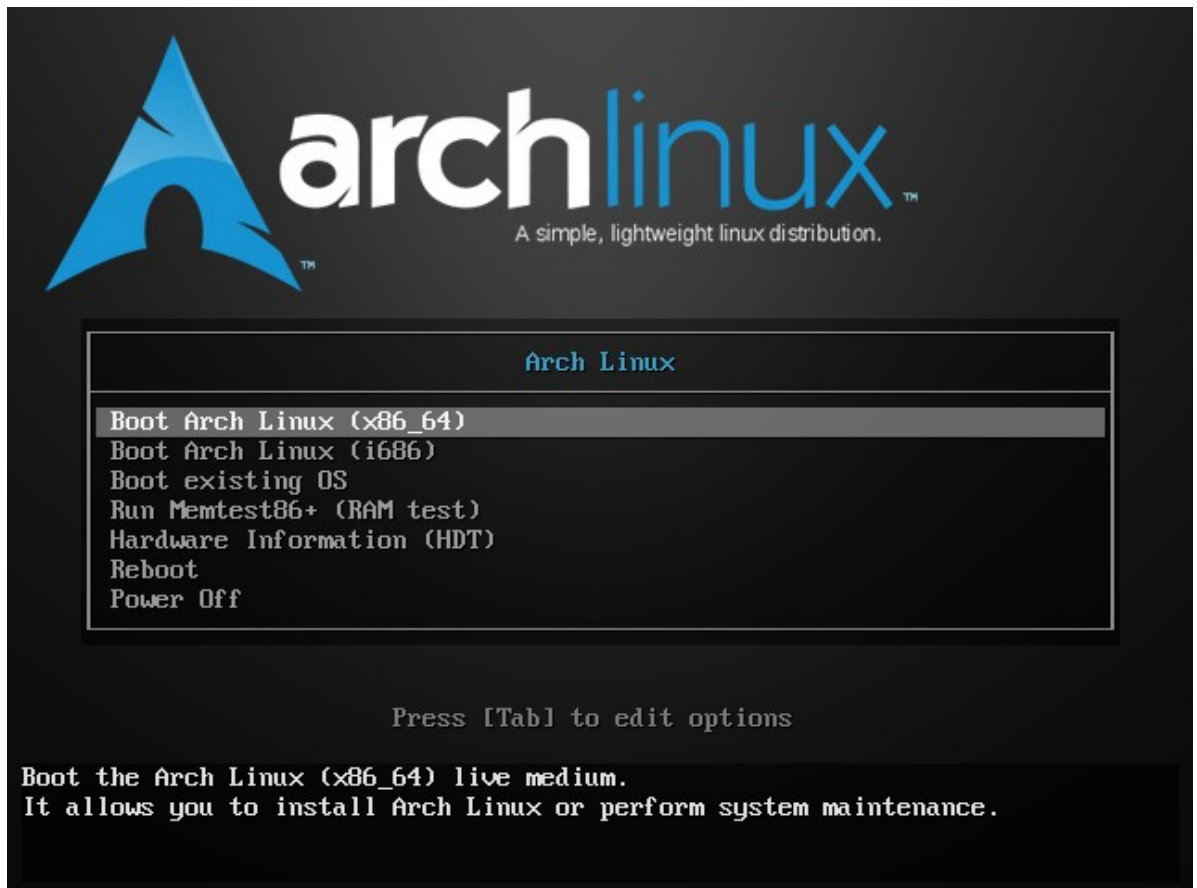
La distribución Linux elegida ha sido Archlinux, una distribución GNU/Linux independiente, de propósito general, lo suficientemente versátil como para adaptarse a cualquier función. Su desarrollo se centra en la simplicidad, el minimalismo y la elegancia de código. Arch se instala como un sistema de base mínimo, configurado por el usuario, sobre el que se monta su propio entorno ideal, mediante la instalación de sólo lo que se requiere o se desea para sus propósitos particulares. Las GUI de las utilidades de configuración no son proporcionadas oficialmente y la mayor parte de la configuración del sistema se realiza desde la shell, mediante la simple modificación de archivos de textos. Basado en un modelo rolling-release, Arch se esfuerza por mantenerse al día y, por lo general, ofrece las últimas versiones estables de la mayoría del software.

Pregunta 2

Documentar el proceso de preparación e instalación de la distribución de Linux elegida, comparándolo con el descrito en clase.

1.- Iniciando la iso de Archlinux

En una máquina virtual con un windows y algo de espacio disponible, seleccionamos la iso de instalación de Archlinux y reiniciamos la máquina para entrar en el menú de arranque de la distro linux. Aquí arrancaremos Archlinux en la arquitectura más adecuada para nuestra máquina.



2.- Distribución del teclado

La distribución del teclado con la que arranca la iso, por defecto probablemente no sea la adecuada para nuestro teclado. Si el teclado es el estándar español, usaremos el siguiente comando:

```
# loadkeys es
```

```
Arch Linux 4.0.1-1-ARCH (tty1)
archiso login: root (automatic login)
root@archiso ~ # loadkeys es
root@archiso ~ # _
```

3.- Particionar el disco

Usando la utilidad cfdisk, particionaremos el espacio libre del disco para el nuevo Sistema Operativo. Crearemos una partición para la raíz (4GiB en mi caso), otra para home (5,5GiB) y una última como swap (0,5GiB). En esta última cambiaremos el Tipo de partición y seleccionaremos Linux swap. Cabe destacar, que puesto que es un disco MBR, solo caben 4 particiones primarias, así que ha habido que crear una extendida para poder alojar dentro las particiones lógicas de home y swap. Notese que aún no hemos formateado nada, solo particionado. Usaremos Write para escribir los datos y Quit para volver al prompt.

```

Disk: /dev/sda
Size: 20 GiB, 21474836480 bytes, 41943040 sectors
Label: dos, identifier: 0xe599bdf7

Device      Boot      Start        End    Sectors   Size Id Type
/dev/sda1   *          2048        206847    204800    100M  7 HPFS/NTFS/exFAT
/dev/sda2             206848    20969471    20762624    9.9G  7 HPFS/NTFS/exFAT
/dev/sda3             20969472    29358079    8388608     4G  83 Linux
/dev/sda4             29358080    41943039    12584960     6G   5 Extended
└─/dev/sda5             29360128    40869887    11509760    5.5G  83 Linux
>> └─/dev/sda6             40871936    41943039    1071104    523M  82 Linux swap / Solaris

[Bootable] [ Delete ] [ Quit ] [ Type ] [ Help ] [ Write ]
[ Dump ]

Write partition table to disk (this might destroy data)
```

4.- Formatear las particiones

Una vez particionado, toca formatear. El comando `lsblk` nos mostrará las unidades y sus particiones, y nos ayudará a elegir las particiones a formatear correctamente.

Formatearemos `/dev/sda3` y `/dev/sda5` como `ext4`, y les pondremos los labels `root` y `home` respectivamente con el parámetro `-L` para identificar cada partición. `/dev/sda6` lo formatearemos como `swap` con el comando `mkswap`, y le pondremos con el mismo parámetro el label `swap`.

```
root@archiso ~ # lsblk
NAME                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda                   8:0    0   20G  0 disk
├─sda1                 8:1    0  100M  0 part
├─sda2                 8:2    0   9.9G  0 part
├─sda3                 8:3    0    4G  0 part
├─sda4                 8:4    0    1K  0 part
├─sda5                 8:5    0   5.5G  0 part
└─sda6                 8:6    0  522M  0 part
sr0                   11:0    1  626M  0 rom  /run/archiso/bootmnt
loop0                 7:0    0  272M  1 loop /run/archiso/sfs/airootfs
loop1                 7:1    0   32G  1 loop
└─arch_airootfs       254:0    0   32G  0 dm  /
loop2                 7:2    0  256M  0 loop
└─arch_airootfs       254:0    0   32G  0 dm  /
root@archiso ~ # mkfs.ext4 /dev/sda3 -L root && mkfs.ext4 /dev/sda5 -L home && m
kswap /dev/sda6 -L swap_
```

5.- Montar particiones

Una vez formateadas las particiones, toca montarlas.

Montaremos todo el árbol de directorios como quisiéramos que quedara en la instalación final bajo el directorio `/mnt`. Montaremos pues primero `sda3` en `/mnt`, después crearemos el directorio `/mnt/home`, y montaremos `sda5` en el. Por último activaremos la partición que hemos creado como `swap`. Es importante establecer este árbol correctamente, ya que esta configuración se copiará al `/etc/fstab` de nuestra instalación final.

```
root@archiso ~ # mount /dev/sda3 /mnt && mkdir /mnt/home && mount /dev/sda5 /mnt
/home && swapon /dev/sda6_
```

6.- Seleccionar el servidor de réplica

Archlinux descarga los paquetes desde un servidor. Normalmente, los servidores regionales funcionarán mejor. Para elegir el servidor, modificaremos el archivo `/etc/pacman.d/mirrorlist` y copiaremos la línea correspondiente al servidor que consideremos más oportuno al principio del documento. Puesto que estamos en España, he considerado que el servidor `rediris` es el más apropiado. Vale la pena configurar este archivo correctamente, ya que una copia de este irá a parar a nuestra instalación final.

```
##
## Arch Linux repository mirrorlist
## Sorted by mirror score from mirror status page
## Generated on 2015-05-01
##

Server = http://sunsite.rediris.es/mirror/archlinux/$repo/os/$arch
## Score: 0.3, France
Server = http://archlinux.polymorf.fr/$repo/os/$arch
## Score: 0.3, Denmark
Server = http://mirror.one.com/archlinux/$repo/os/$arch
## Score: 0.4, Netherlands
Server = http://ftp.nluug.nl/os/Linux/distr/archlinux/$repo/os/$arch
## Score: 0.4, Germany
Server = http://archlinux.my-universe.com/$repo/os/$arch
## Score: 0.4, France
Server = http://arch.tamcore.eu/$repo/os/$arch
## Score: 0.4, Germany
Server = http://mirror.js-webcoding.de/pub/archlinux/$repo/os/$arch
## Score: 0.5, Norway
Server = http://os-sharing.org/archlinux/$repo/os/$arch
## Score: 0.7, Ukraine
Server = http://archlinux.bln-ua.net/$repo/os/$arch
```

7.- Instalar los paquetes del sistema base

Usaremos el script `pacstrap` incluido en la iso para instalar el grupo de paquetes base.

```
root@archiso ~ # pacstrap /mnt base
==> Creating install root at /mnt
==> Installing packages to /mnt
:: Synchronizing package databases...
core               122.3 KiB   35.8K/s   00:03 [#####] 100%
extra              1723.8 KiB  699K/s   00:02 [#####] 100%
community          2.6 MiB    614K/s   00:04 [#####] 100%
```


8.- Configurar el sistema

Primero generaremos el archivo `fstab` a partir de el árbol que hemos montado antes con el comando `"# genfstab -p /mnt >> /mnt/etc/fstab"`. Después entraremos en el sistema recién instalado, usando el comando `"# arch-chroot /mnt"`.

```
root@archiso ~ # genfstab -p /mnt >> /mnt/etc/fstab
root@archiso ~ # arch-chroot /mnt
sh-4.3# _
```

Ahora, una vez dentro de nuestro sistema, configuraremos el nombre del equipo usando `"# echo nombre_equipo > /etc/hostname"`. También cambiaremos la zona horaria creando un link simbólico llamado `localtime` dentro de `/etc` apuntando al archivo correspondiente a nuestra zona horaria. En nuestro caso, usaremos el comando `"# ln -sf /usr/share/zoneinfo/Europe/Madrid /etc/localtime"`.

```
sh-4.3# echo archvm >> /etc/hostname
sh-4.3# ln -sf /usr/share/zoneinfo/Europe/Madrid /etc/localtime
sh-4.3# vi /etc/locale.gen _
```

Configuraremos el idioma del sistema modificando el archivo `/etc/locale.gen` y descomentando las líneas correspondientes a nuestro idioma.

```
#en_ZA ISO-8859-1
#en_ZM UTF-8
#en_ZW.UTF-8 UTF-8
#en_ZW ISO-8859-1
#es_AR.UTF-8 UTF-8
#es_AR ISO-8859-1
#es_BO.UTF-8 UTF-8
#es_BO ISO-8859-1
#es_CL.UTF-8 UTF-8
#es_CL ISO-8859-1
#es_CO.UTF-8 UTF-8
#es_CO ISO-8859-1
#es_CR.UTF-8 UTF-8
#es_CR ISO-8859-1
#es_CU UTF-8
#es_DO.UTF-8 UTF-8
#es_DO ISO-8859-1
#es_EC.UTF-8 UTF-8
#es_EC ISO-8859-1
es_ES.UTF-8 UTF-8
es_ES ISO-8859-1
#es_ES@euro ISO-8859-15
#es_GT.UTF-8 UTF-8
#es_GT ISO-8859-1
:
```

Una vez guardado el archivo, ejecutaremos el comando `"# locale-gen"` para generar los archivos descomentados. Ahora queda insertar el idioma en el archivo `/etc/locale.conf` con

el comando `"# echo LANG=nuestro_locale > /etc/locale.conf"`.

También ejecutaremos `"# echo KEYMAP=es > /etc/vconsole.conf"` para establecer la distribución del teclado de nuestra nueva instalación.

Ahora hay que crear una imagen RAM inicial nueva con la orden `"# mkinitcpio -p linux"`.

```
sh-4.3# mkinitcpio -p linux
==> Building image from preset: /etc/mkinitcpio.d/linux.preset: 'default'
-> -k /boot/vmlinuz-linux -c /etc/mkinitcpio.conf -g /boot/initramfs-linux.img
==> Starting build: 4.0.4-2-ARCH
-> Running build hook: [base]
-> Running build hook: [udev]
-> Running build hook: [autodetect]
-> Running build hook: [modconf]
-> Running build hook: [block]
-> Running build hook: [filesystems]
-> Running build hook: [keyboard]
-> Running build hook: [fsck]
==> Generating module dependencies
==> Creating gzip-compressed initcpio image: /boot/initramfs-linux.img
==> Image generation successful
==> Building image from preset: /etc/mkinitcpio.d/linux.preset: 'fallback'
-> -k /boot/vmlinuz-linux -c /etc/mkinitcpio.conf -g /boot/initramfs-linux-fal
lback.img -S autodetect
==> Starting build: 4.0.4-2-ARCH
-> Running build hook: [base]
-> Running build hook: [udev]
-> Running build hook: [modconf]
-> Running build hook: [block]
```

Para acabar con la configuración del sistema, estableceremos la contraseña para el usuario root con el comando `"# passwd"`.

```
sh-4.3# passwd
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
sh-4.3# _
```

9.- Instalar un gestor de arranque

Ya tenemos el sistema instalado, pero este no puede iniciar a menos que instalemos un gestor de arranque. En este ejemplo usaremos Syslinux, aunque podríamos usar otros gestores como GRUB, LILO u otros muchos. El comando `"# pacman -S syslinux"` instalará el paquete, así como sus dependencias si las tuviera.

```
sh-4.3# pacman -S syslinux
resolving dependencies...
looking for conflicting packages...

Packages (1) syslinux-6.03-3

Total Download Size:  1.47 MiB
Total Installed Size: 4.62 MiB

:: Proceed with installation? [Y/n] _
```

Una vez instalado el paquete, queda instalar el gestor en la máquina. Usaremos el script `"# syslinux-install_update -i -a -m"` para hacerlo.

```
sh-4.3# syslinux-install_update -i -a -m
Syslinux BIOS install successful
Boot Flag Set - /dev/sda3
Installed MBR (/usr/lib/syslinux/bios/mbr.bin) to /dev/sda
sh-4.3# _
```

El parámetro `-i` indica que instale los archivos, `-a` que marque la partición como activa, y `-m` que instale el MBC en el MBR.

Por defecto, el menú nos mostrará únicamente las opciones relativas al linux que hemos instalado, así que para acceder a nuestra instalación anterior de windows, tendremos que hacer unos cambios en el archivo `/boot/syslinux/syslinux.cfg`.

```

LABEL arch
    MENU LABEL Arch Linux
    LINUX ../vmlinuz-linux
    APPEND root=/dev/sda3 rw
    INITRD ../initramfs-linux.img

LABEL archfallback
    MENU LABEL Arch Linux Fallback
    LINUX ../vmlinuz-linux
    APPEND root=/dev/sda3 rw
    INITRD ../initramfs-linux-fallback.img

LABEL windows
    MENU LABEL Windows
    COM32 chain.c32
    APPEND hd0 1

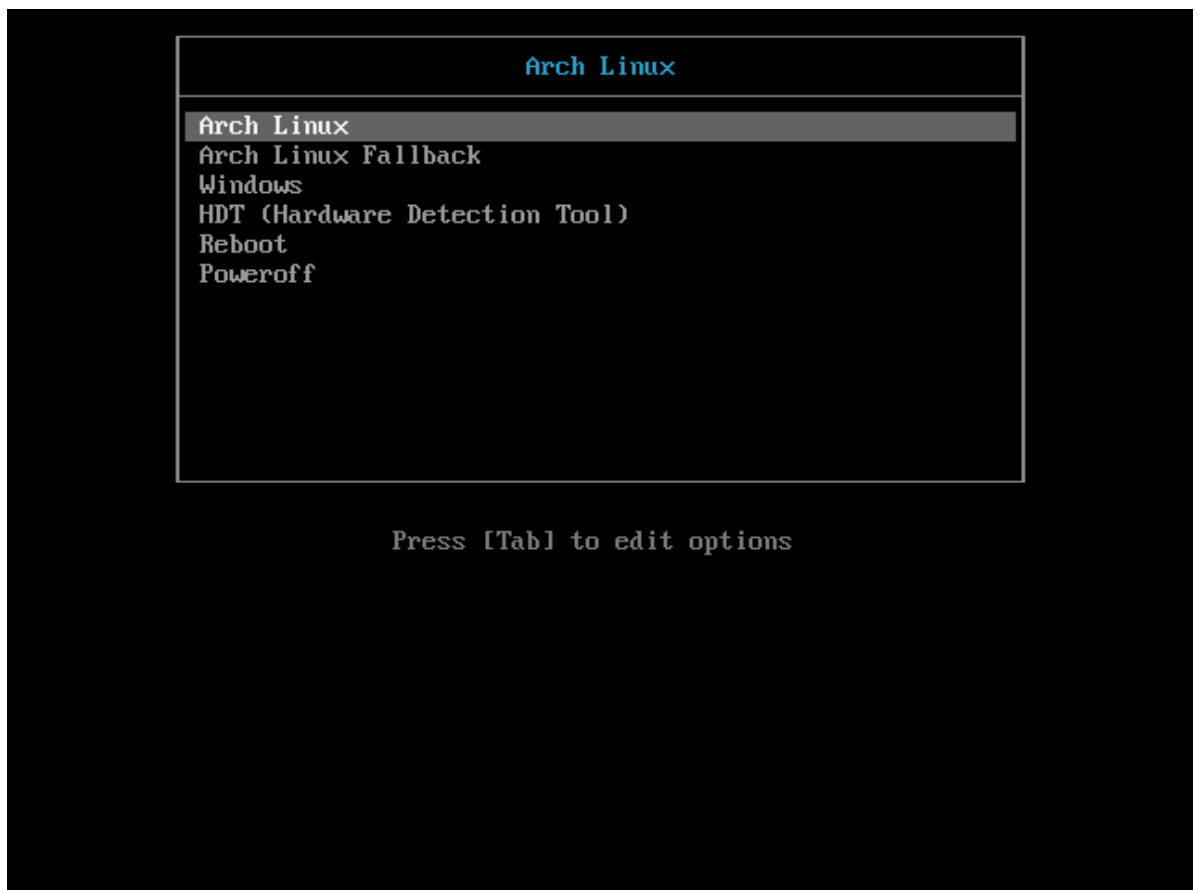
LABEL hdt
    MENU LABEL HDT (Hardware Detection Tool)
    COM32 hdt.c32

LABEL reboot
    MENU LABEL Reboot
    COM32 reboot.c32
```

En este archivo, buscaremos la entrada windows, y la descomentaremos entera. En nuestro caso, así es suficiente, ya que el PBC de windows está en la partición 1 del disco duro, pero en caso de no ser así, habría que cambiar la línea que pone APPEND. Hd0 1 indica que es el disco duro 0, partición 1. Es importante saber que los discos duros se enumeran empezando por el 0, mientras que las particiones empiezan por el 1.

10.- Reiniciar

Llegados a este punto, solo queda salir del chroot con `"# exit"`, desmontar todo el árbol de directorios con `"# umount -R /mnt"`, y usar el comando `"# reboot"` para reiniciar el sistema. Si todo ha ido bien, nos encontraremos en el menú de arranque de Syslinux.



Pregunta 3

Usuario root

1.- Disponibilidad del usuario root tras la instalación del s.o.

Tras la instalación del sistema operativo, el único usuario disponible es root, con la contraseña que le hemos proporcionado al ejecutar el comando `passwd`. Si queremos más usuarios, tendremos que crearlos desde el usuario root.

Cabe destacar que el comando `sudo` no está disponible por defecto. Habrá que instalar el respectivo paquete para poder usarlo.

2.- Repaso de las órdenes vistas en clase para revisar funcionalidades en su momento descartadas al precisar de privilegios de usuario root para su realización. P.e: edición de la fecha y hora del sistema, orden `chown`,...

En clase hemos visto una serie de ordenes que no se han podido probar por necesitar privilegios de root. También hay algunas que se podían utilizar sin los privilegios, pero estaban limitadas. Estas ordenes son:

- **date:** Esta orden puede usarse sin ser root para ver la fecha, pero si se tienen privilegios de root también permite modificarla.
- **chown:** Permite cambiar el usuario propietario de un archivo o carpeta.
- **useradd:** Permite añadir usuarios al sistema.
- **usermod:** Permite modificar los usuarios del sistema.
- **userdel:** Permite eliminar usuarios.
- **chfn:** Permite cambiar la descripción de un usuario.
- **adduser:** Utilidad que permite crear usuarios fácilmente.
- **groupadd:** Permite crear grupos.
- **groupmod:** Permite modificar los grupos ya existentes.

- **groupdel:** Permite eliminar grupos.
- **pwck:** Solo para root. Busca en `/etc/passwd` posibles errores de formato, así como posibles inconsistencias: usuarios duplicados, usuarios sin directorio home...
- **grpck:** Solo para root. Busca en `/etc/group` posibles errores de formato, así como posibles inconsistencias.
- **passwd:** Sin ser root permite modificar tu contraseña, siendo root permite cambiar la contraseña de cualquier usuario, borrar contraseñas, bloquear usuarios, ver estado de las contraseñas...
- **su:** Permite conmutar el id de presentación temporalmente. Permite conmutar a cualquier usuario, por defecto es el root.
- **sudo:** Se utiliza para ejecutar una orden ejecutar una orden con privilegios de otro usuario, incluido el administrador. El fichero `/etc/sudoers` permite configurar su funcionamiento.
- **wall:** Utilidad de administrador que envía simultáneamente e inmediatamente un mensaje a todos los usuarios que estén en ese momento conectados al sistema.
- **mount:** Sirve para montar sistemas de archivos en el árbol de directorios.
- **umount:** Sirve para desmontar sistemas de archivos montados sobre el arbol de directorios.
- **mkfs:** Formatea un sistema de archivos.
- **fsck:** Hace un chequeo del sistema de archivos.
- **init:** ArchLinux ya no hace uso del anticuado sistema SysVinit, sino que ha migrado a systemd. Como consecuencia, algunos conceptos como los runlevels han cambiado. Systemd tiene el concepto de *targets* que sirve a un propósito similar al de niveles de ejecución (runlevels) pero actúa ligeramente diferente. Cada *target* es nombrado en vez de numerado y está diseñado para servir un propósito específico. Sin embargo, para mantener cierto nivel de

compatibilidad, a estos targets se les asigna un número equivalente a su runlevel. Así pues, el runlevel 0 equivaldría en systemd a runlevel0.target (modo compatibilidad) o más adecuadamente, poweroff.target. El comando init existe para mantener compatibilidad, actuando de igual manera que en sysVinit, aunque por detrás es un simple traductor de comandos al respectivo comando en systemd.

3.- Documentar la orden shutdown

```
$ shutdown [OPCIONES...] [TIEMPO] [WALL...]
```

La orden shutdown se usa para apagar, reiniciar o poner en estado halt la máquina.

El primer argumento se refiere al tiempo, que usualmente es "now". Opcionalmente puede ir seguido de un mensaje wall, que será enviado a todos los usuario logueados antes de cerrar su sesión.

El formato de la cadena de tiempo puede ser "hh:mm" en formato 24h especificando la hora a la que será ejecutado el shutdown. Alternativamente puede usarse la sintaxis +m refiriéndose a los minutos m a partir del momento de ejecución del comando. "now" es un alias para "+0" para ejecutar un shutdown inmediato. Si no hay argumento especificado, se usa por defecto "+1".

Cuando se usa el argumento de tiempo, 5 minutos antes de que el sistema se apague, se crea el archivo /run/nologin para asegurarse que no se permiten mas logins en el sistema.

OPCIONES

- **--help** Imprime un mensaje de ayuda y sale.
- **-H, --halt** Pone en estado de halt la máquina.
- **-P, --poweroff** Apaga la máquina (por defecto).
- **-r, --reboot** Reinicia la máquina.
- **-h** Equivale a --poweroff, a menos que --halt esté especificado.
- **-k** No enviar a halt, apagar o reiniciar, solo escribir

el mensaje wall.

- **--no-wall** No enviar el mensaje wall antes del halt, power-off o reboot.
- **-c** Cancelar un shutdown pendiente. Puede ser usado para cancelar el efecto de una invocación shutdown con un argumento que no sea "+0" o "now".

Pregunta 4

Sistema de arranque

1.- Documentar el proceso de configuración del sistema de arranque Linux.

El gestor de arranque usado en esta instalación ha sido syslinux. La configuración de este se hace mediante la edición manual del archivo `/boot/syslinux/syslinux.cfg`.

A diferencia de otros gestores de arranque, la configuración de este es considerablemente sencilla. Podemos dividir el archivo en 3 partes. La primera de configuración general, como el `TIMEOUT` o la opción por defecto `DEFAULT`, la segunda con configuración visual del menú, que suelen ser líneas precedidas de la palabra `MENU` donde especificamos tamaños, colores o incluso un background, y la tercera y mas importante parte, las entradas de los sistemas operativos, que empiezan por la palabra `LABEL`.

```

DEFAULT arch
PROMPT 0          # Set to 1 if you always want to display the boot: prompt
TIMEOUT 50
UI vesamenu.c32

MENU TITLE Arch Linux
MENU BACKGROUND splash.png
MENU COLOR border      30;44  #40ffffff #a0000000 std
MENU COLOR title       1;36;44 #9033ccff #a0000000 std
MENU COLOR sel         7;37;40 #e0ffffff #20ffffff all
MENU COLOR unsel       37;44  #50ffffff #a0000000 std
MENU COLOR help        37;40  #c0ffffff #a0000000 std
MENU COLOR timeout_msg 37;40  #80ffffff #00000000 std
MENU COLOR timeout     1;37;40 #c0ffffff #00000000 std
MENU COLOR msg07       37;40  #90ffffff #a0000000 std
MENU COLOR tabmsg      31;40  #30ffffff #00000000 std

LABEL arch
  MENU LABEL Arch Linux
  LINUX ../vmlinuz-linux
  APPEND root=/dev/sda3 rw
  INITRD ../initramfs-linux.img

LABEL windows
:
```

En la imagen podemos ver las tres partes diferenciadas. `DEFAULT` indica el label por defecto que será ejecutado una vez pasado el `TIMEOUT`, que está en décimas de segundo (10 es 1 segundo). `PROMPT` se usa para indicar si debe aparecer

el mensaje de boot una vez seleccionada una entrada, siendo un valor binario (0 o 1). UI es usado para importar la librería gráfica que construirá el entorno.

En el siguiente bloque, vemos varias entradas MENU, que indican el título de la pantalla, un background, y colores de varios elementos.

```
MENU COLOR help      37;40    #c0ffffff #a0000000 std
MENU COLOR timeout_msg 37;40    #80ffffff #00000000 std
MENU COLOR timeout    1;37;40  #c0ffffff #00000000 std
MENU COLOR msg07      37;40    #90ffffff #a0000000 std
MENU COLOR tabmsg     31;40    #30ffffff #00000000 std

LABEL arch
  MENU LABEL Arch Linux
  LINUX ../vmlinuz-linux
  APPEND root=/dev/sda3 rw
  INITRD ../initramfs-linux.img

LABEL windows
  MENU LABEL Windows
  COM32 chain.c32
  APPEND hd0 1

LABEL reboot
  MENU LABEL Reboot
  COM32 reboot.c32

LABEL poweroff
  MENU LABEL Poweroff
  COM32 poweroff.c32
```

Los elementos LABEL, están marcados con un nombre único (arch, windows, reboot...) que no tiene mayor importancia que la de seleccionar el DEFAULT. Dentro de cada LABEL, habrá un MENU LABEL indicando el nombre a mostrar, y dependiendo de lo que haga esta entrada, unas cosas u otras.

Para linux, habrá una entrada LINUX donde se pondrá la ruta del kernel. Un APPEND que indicará cual es el disco duro a arrancar y los permisos con los que montarlo, y un INITRD donde se indicara el disco RAM inicial.

Para windows, el bootloader hará lo que se llama un chainload, que consiste en cargar el sector de arranque de la partición que le indiquemos, en este caso la de windows, y así cargará su PBC y después el bootloader nativo de windows. Para hacer el chainload, se carga la librería chain.c32 en con la orden COM32, y después con la orden APPEND hacemos el chainload al disco 0 partición 1 (hd0 1). Notar que los discos empiezan a numerarse por el 0 y las particiones por el 1.

Las entradas poweroff y reboot simplemente cargan una librería que hace lo propio, mediante la orden COM32.

2.- Documentar el proceso de configuración del sistema de arranque Microsoft

Puesto que el gestor de arranque de Microsoft requiere que la partición donde reside esté activa, modificaremos el gestor de arranque de linux para que pueda arrancar sin ser así. Ejecutaremos el siguiente comando en linux:

```
# printf '\x3' | cat /usr/lib/syslinux/bios/altmbr.bin - |  
dd bs=440 count=1 iflag=fullblock of=/dev/sda
```

Donde x3 es nuestra partición /dev/sda3 donde reside syslinux.

Después montaremos la partición de 100MB de windows y crearemos una copia del MBR (actualmente de linux) allí.

```
# mount /dev/sda1 /mnt
```

```
# dd if=/dev/sda of=/mnt/archlinux.bin bs=512 count=1
```

Y con el comando "cfdisk" cambiaremos la partición bootable de la /dev/sda3 a la /dev/sda1.

Una vez hecho esto, toca iniciar el disco de instalación de windows, iniciar una reparación, abrir una consola y ejecutar el comando:

```
> bootrec /fixmbr
```

Así recuperaremos el MBR de windows, y comprobamos que efectivamente, al reiniciar, se inicia windows con normalidad.

Una vez en windows, tendremos que editar el BCD para cargar la copia del que era el MBR de linux. Para hacer esto crearemos una nueva entrada en el BCD:

```
> bcdedit /create /d "ArchLinux" /application bootsector
```

Este comando nos devolverá una id similar a {d7294d4e-9837-11de-99ac-f3f3a79e3e93}. En los siguientes comandos sustituiremos {ID} por la que nos diera este comando.

Establecemos el disco en el que buscará el arrancable:

```
> bcdedit /set {ID} device  
partition=\Device\HarddiskVolume1
```

Establecemos el fichero que tiene que arrancar:

```
> bcdedit /set {ID} path \archlinux.bin
```

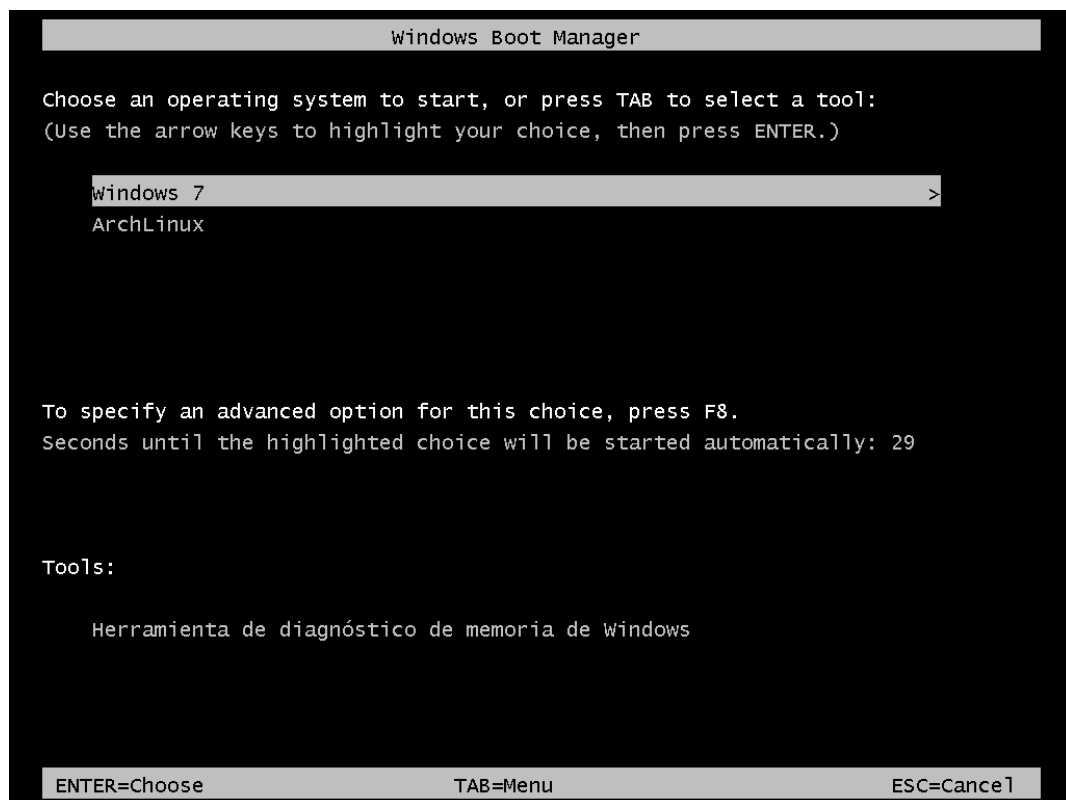
Añadimos la entrada a la lista:

```
> bcdedit /displayorder {ID} /addlast
```

Y insertamos un timeout:

```
> bcdedit /timeout 30
```

Al reiniciar la máquina, podemos ver como aparece un menú dandonos a elegir entre Windows 7 o ArchLinux.



Pregunta 5

Configurar de un entorno gráfico. (Comparación con el entorno Windows, herramientas de administración del s.o., aplicaciones de usuario incluidas, configuración del entorno gráfico, ...)

1.- Configurar el X Window System (Entorno gráfico básico)

El primer paso para instalar un entorno gráfico, es instalar un entorno gráfico básico. El más extendido es el **X Window System**, también conocido como **X11** o simplemente **X**.

Primero actualizaremos el sistema:

```
$ sudo pacman -Syu
```

Instalemos primero los paquetes base **Xorg** (la implementación *open source* del **X Window System**):

```
$ sudo pacman -S xorg-server xorg-xinit xorg-utils xorg-server-utils
```

Instalaremos también mesa para el soporte 3D:

```
$ sudo pacman -S mesa mesa-demos
```

Instalaremos los drivers de vídeo. Puesto que usamos una máquina virtual en virtualbox, solo tendremos que instalar los guest additions e activar los modules.

```
$ sudo pacman -S virtualbox-guest-utils
```

```
$ sudo echo "vboxguest
```

```
> vboxsf
```

```
> vboxvideo
```

```
> vboxservice" > /etc/modules-load.d/virtualbox.conf
```

Notar los saltos de línea en el echo.

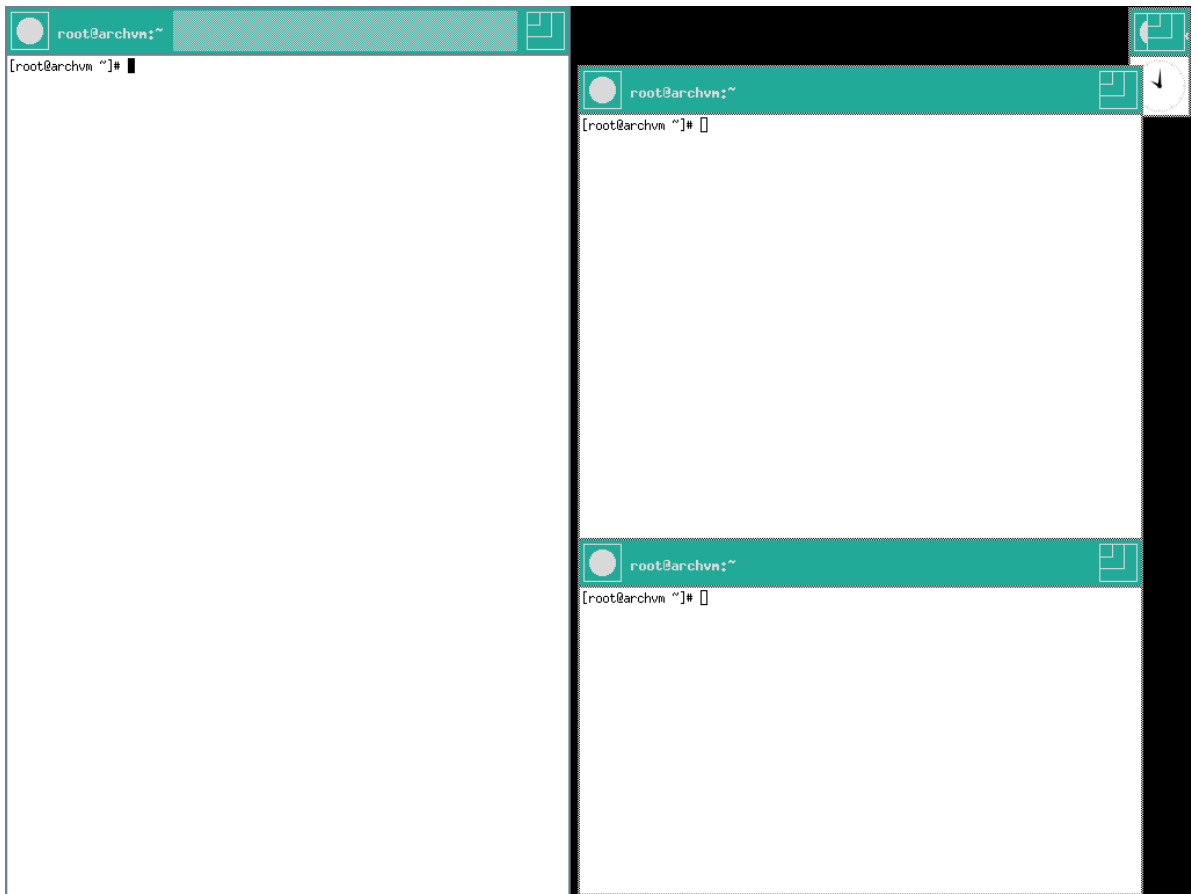
Ejecutaremos lo siguiente para ejecutar los modulos en esta sesión:

```
$ sudo modprobe -a vboxguest vboxsf vboxvideo vboxservice
```

Podemos instalar un entorno muy básico para comprobar que todo vaya bien.

```
$ sudo pacman -S xorg-twm xorg-xclock xterm
```

Ahora ejecutamos "startx" y si todo va bien, tendremos un entorno como el siguiente:



Podemos salir de este escritorio con el comando "sudo kill x"

2.- Instalar Openbox

Openbox ha sido el entorno elegido por su ligereza, rapidez y posibilidades de configuración.

La instalación se realiza con el comando:

```
$ sudo pacman -S openbox
```

```

[isouser2015@archvm root]$ sudo pacman -S openbox
resolviendo dependencias...
buscando conflictos entre paquetes....

Paquetes (1) openbox-3.6-1

Tamaño total de la instalación: 1,19 MiB

:: ¿Continuar con la instalación? [S/n]
(1/1) comprobando las claves del depósito [#####] 100%
(1/1) verificando la integridad de los paquetes [#####] 100%
(1/1) cargando los archivos de los paquetes [#####] 100%
(1/1) comprobando conflictos entre archivos [#####] 100%
(1/1) comprobando el espacio disponible en disco [#####] 100%
(1/1) instalando openbox [#####] 100%
Dependencias opcionales para openbox
  kdatabase-workspace: for the KDE/Openbox xsession
  python2-xdg: for the openbox-xdg-autostart script
[isouser2015@archvm root]$ _

```

Después de su instalación, copiaremos los archivos de configuración en una carpeta llamada openbox que crearemos en el directorio .config de nuestra carpeta personal (crearemos también .config si no existe).

```

[isouser2015@archvm root]$ mkdir -p ~/.config/openbox
[isouser2015@archvm root]$ cp /etc/xdg/openbox/{rc.xml,menu.xml,autostart} ~/.config/openbox
[isouser2015@archvm root]$ _

```

Después crearemos un archivo ~/.xinitrc con el siguiente contenido:

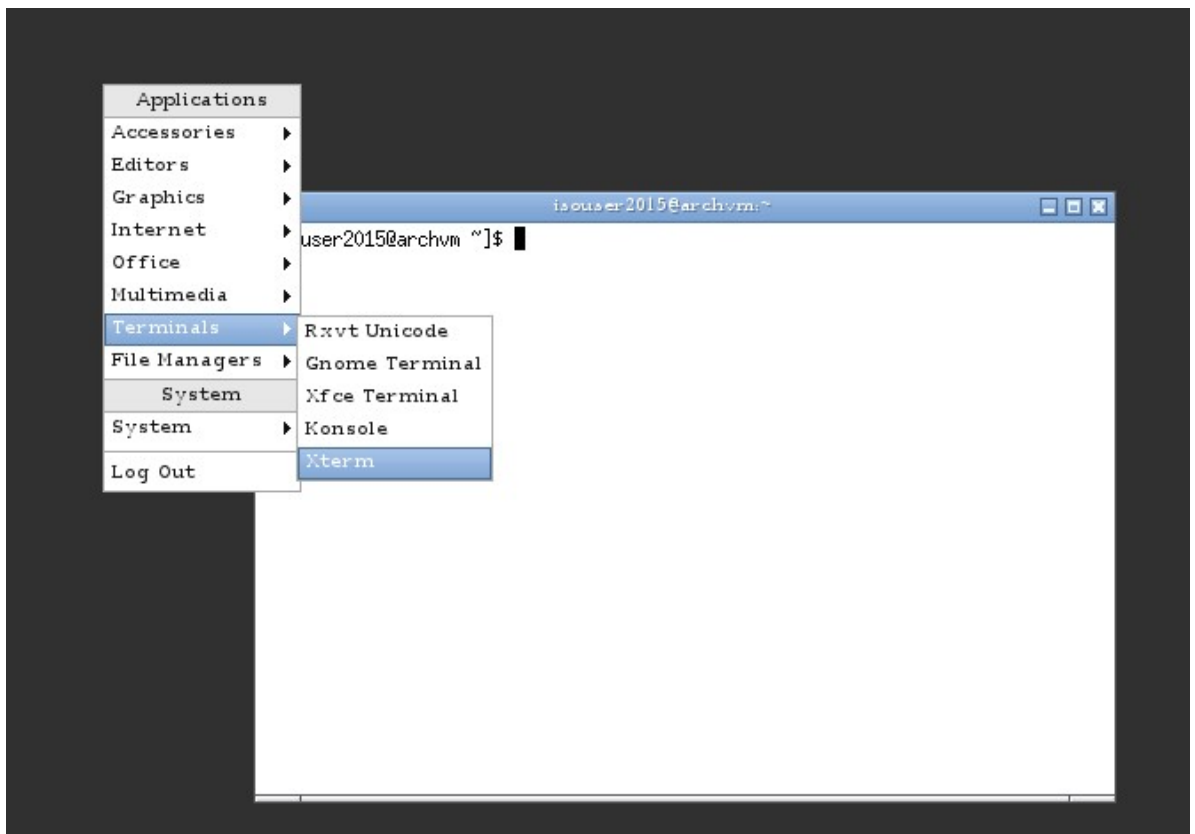
```

#!/bin/sh

exec openbox-session

```

Y ejecutando el comando "startx" tendremos un entorno openbox básico y sin configurar.



3.- Configuración de openbox

Primero instalaremos unas fuentes. No olvidemos que hay que actualizarlas después de instalarlas.

```
$ sudo pacman -S ttf-bitstream-vera ttf-dejavu ttf-droid  
ttf-freefont
```

```
$ fc-cache -vf
```

Archlinux por defecto no nos crea las típicas carpetas Imágenes, Documentos, Música... Lo cual es genial, ya que permite personalizarlas a nuestro gusto. Instalaremos el siguiente paquete:

```
$ sudo pacman -S xdg-user-dirs
```

Y lo configuraremos editando el archivo `/etc/xdg/user-dirs.defaults`:

```
DESKTOP=Desktop  
DOWNLOAD=Descargas  
DOCUMENTS=Documentos  
MUSIC=Música
```

PICTURES=Imágenes

VIDEOS=Vídeos

Y actualizamos los directorios con el siguiente comando:

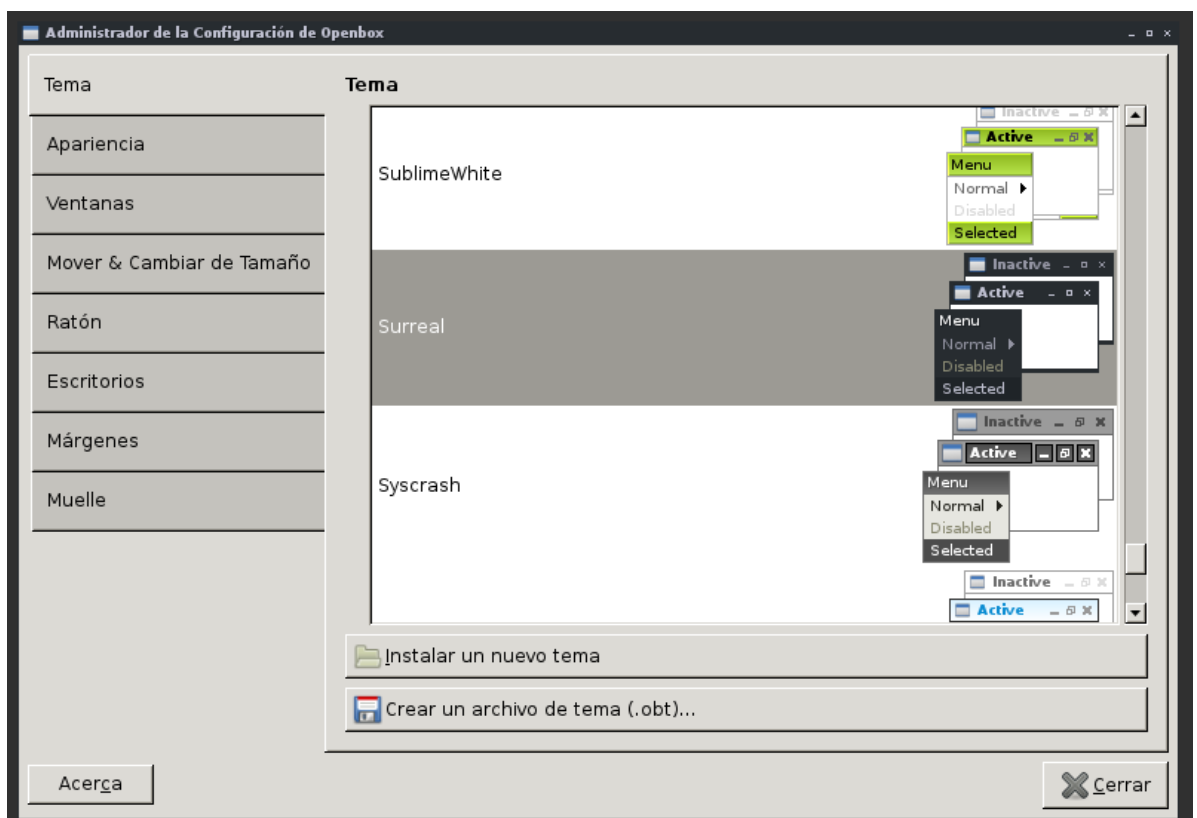
```
$ xdg-user-dirs-update
```

Para ajustar la apariencia de nuestro entorno gráfico, se recomienda la instalación de **obconf**, un administrador de configuración para **Openbox**:

Para instalar más temas, ejecuta:

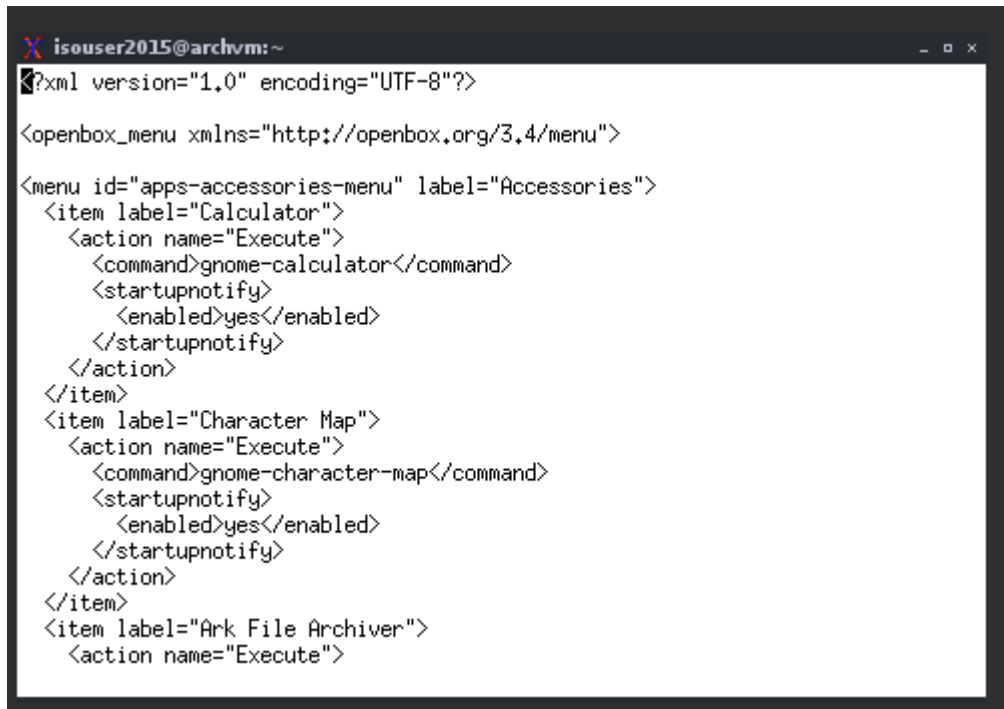
```
$ sudo pacman -S openbox-themes
```

Después, podemos acceder a obconf ejecutando un comando con el mismo nombre o haciendo click derecho sobre el escritorio, y entrando en System → Openbox Configuration Manager.



El menú principal en **Openbox** se invoca con el botón secundario del mouse, y si eres observador notarás que hace referencia a aplicaciones que muy probablemente no tengas instaladas. ¿Cómo editar este menú? Hay varias formas de hacerlo.

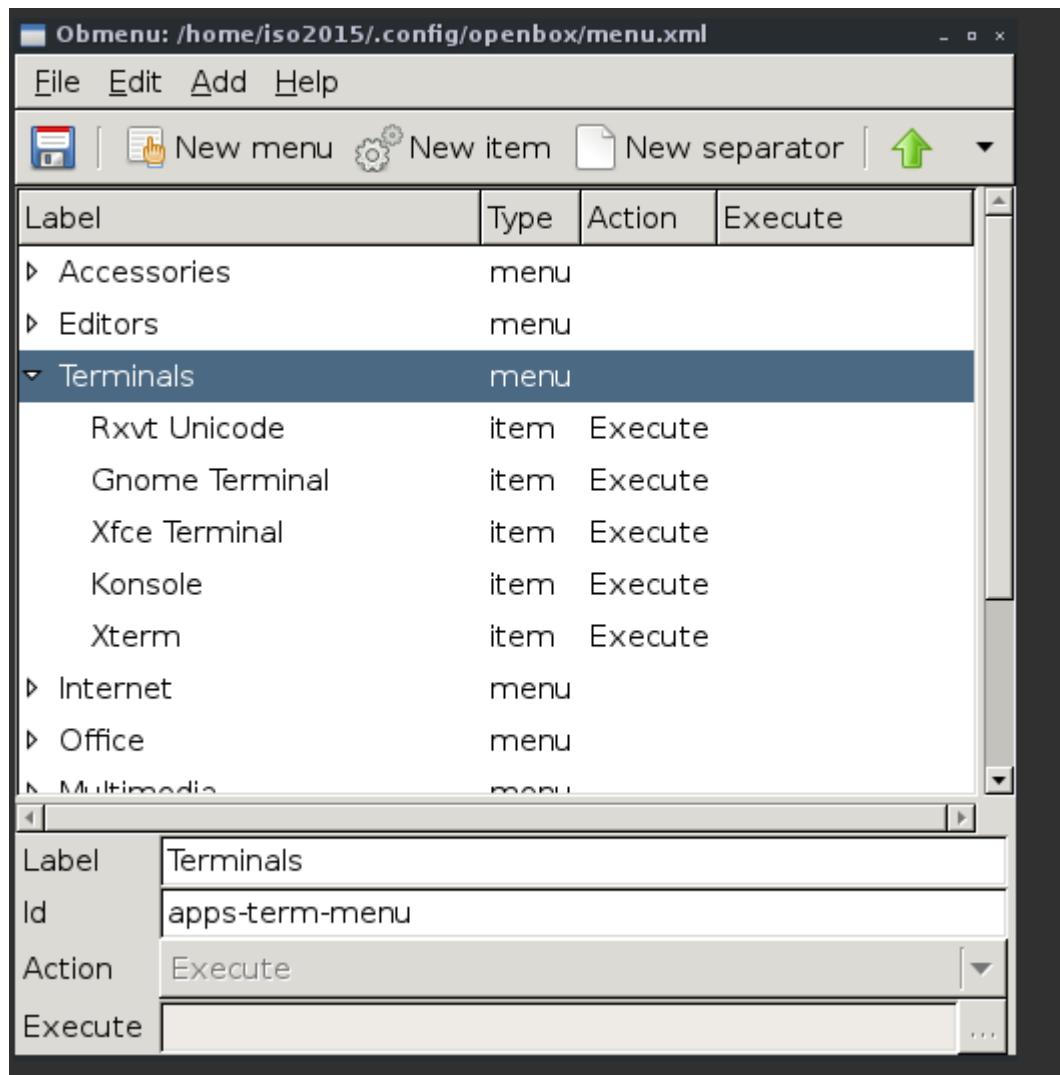
La forma manual: Editando el archivo
~/.config/openbox/menu.xml



```
isouser2015@archvm:~  
?xml version="1.0" encoding="UTF-8"?  
  
<openbox_menu xmlns="http://openbox.org/3.4/menu">  
  
<menu id="apps-accessories-menu" label="Accessories">  
  <item label="Calculator">  
    <action name="Execute">  
      <command>gnome-calculator</command>  
      <startupnotify>  
        <enabled>yes</enabled>  
      </startupnotify>  
    </action>  
  </item>  
  <item label="Character Map">  
    <action name="Execute">  
      <command>gnome-character-map</command>  
      <startupnotify>  
        <enabled>yes</enabled>  
      </startupnotify>  
    </action>  
  </item>  
  <item label="Ark File Archiver">  
    <action name="Execute">
```

La forma gráfica es la utileria obmenu.

\$ pacman -S obmenu



Con ella podemos modificar todas las entradas que queramos, asignando un comando a cada una de ellas. ¡Las posibilidades son infinitas!

Ahora toca un lanzador de aplicaciones. Existen muchos lanzadores de aplicaciones para **Linux**, pero como estamos apegándonos a la filosofía minimalista de **Openbox**, vamos a instalar uno extremadamente liviano, ¡y muy eficiente! Se llama **dmenu** y lo instalan con:

```
$ sudo pacman -S dmenu
```

Para vincularlo a un atajo de teclado, toca modificar otro archivo. `~/.config/openbox/rc.xml`

```

term| koi8rxterm setterm uxterm xterm
X isouser2015@archvm:~
<keybind key="W-e">
  <action name="Execute">
    <startupnotify>
      <enabled>true</enabled>
      <name>Konqueror</name>
    </startupnotify>
    <command>kfmclient openProfile filemanagement</command>
  </action>
</keybind>
<keybind key="A-F2">
  <action name="Execute">
    <command>dmenu_run</command>
  </action>
</keybind>
</keyboard>
<mouse>
  <dragThreshold>1</dragThreshold>
  <!-- number of pixels the mouse must move before a drag begins -->
  <doubleClickTime>500</doubleClickTime>
  <!-- in milliseconds (1000 = 1 second) -->
  <screenEdgeWarpTime>400</screenEdgeWarpTime>
  <!-- Time before changing desktops when the pointer touches the edge of the
  screen while moving a window, in milliseconds (1000 = 1 second).
".config/openbox/rc.xml" 789L, 25772C escritos          332,14      41%

```

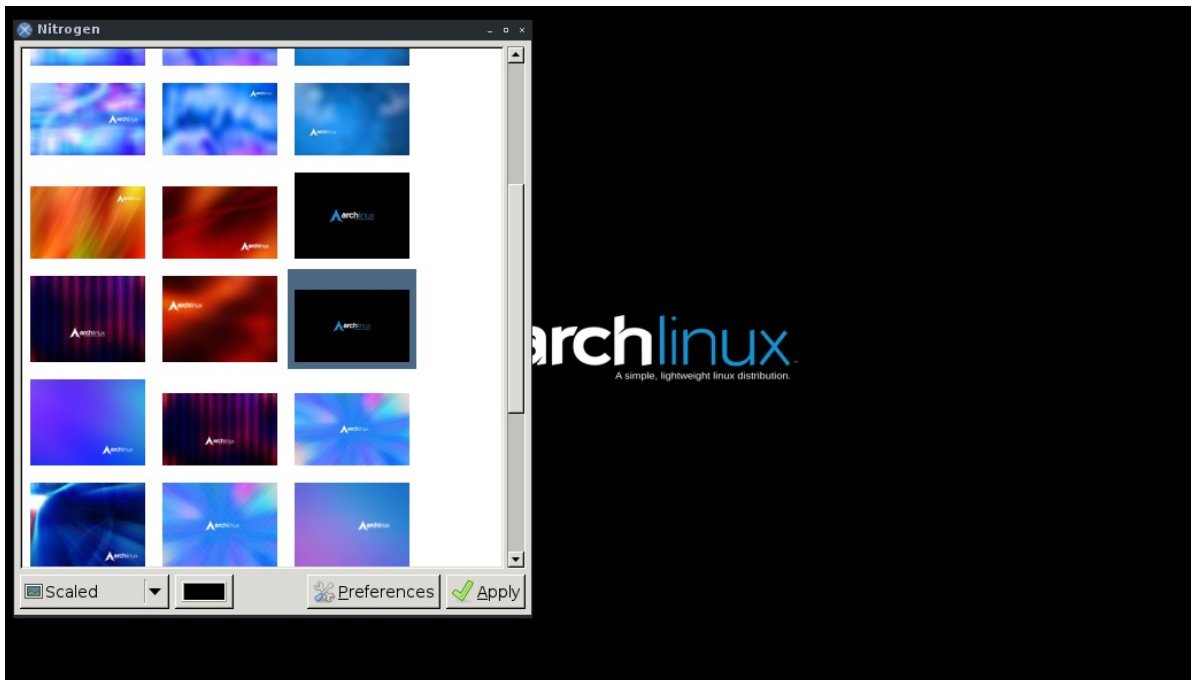
En el ejemplo lo hemos asignado a las teclas Alt-F2, y podemos comprobar que funciona.

Openbox no ofrece establecer de manera nativa el *wallpaper*, pero existen varias utilerías muy livianas que lo hacen. Aquí usaremos **nitrogen** por su simplicidad y ligereza. La instalamos como de costumbre:

```
$ pacman -S nitrogen
```

Ejecutando la orden **nitrogen** (o añadiéndola al menú) podremos cambiar el wallpaper. Para que los cambios permanezcan al reiniciar sesión, añadiremos "nitrogen --restore &" al final del archivo

~/.config/openbox/autostart.



Para acabar sólo nos faltaría un panel para cambiar entre aplicaciones. Uno muy liviano y simple es pypanel. Lo instalamos:

```
$ pacman -S pypanel
```

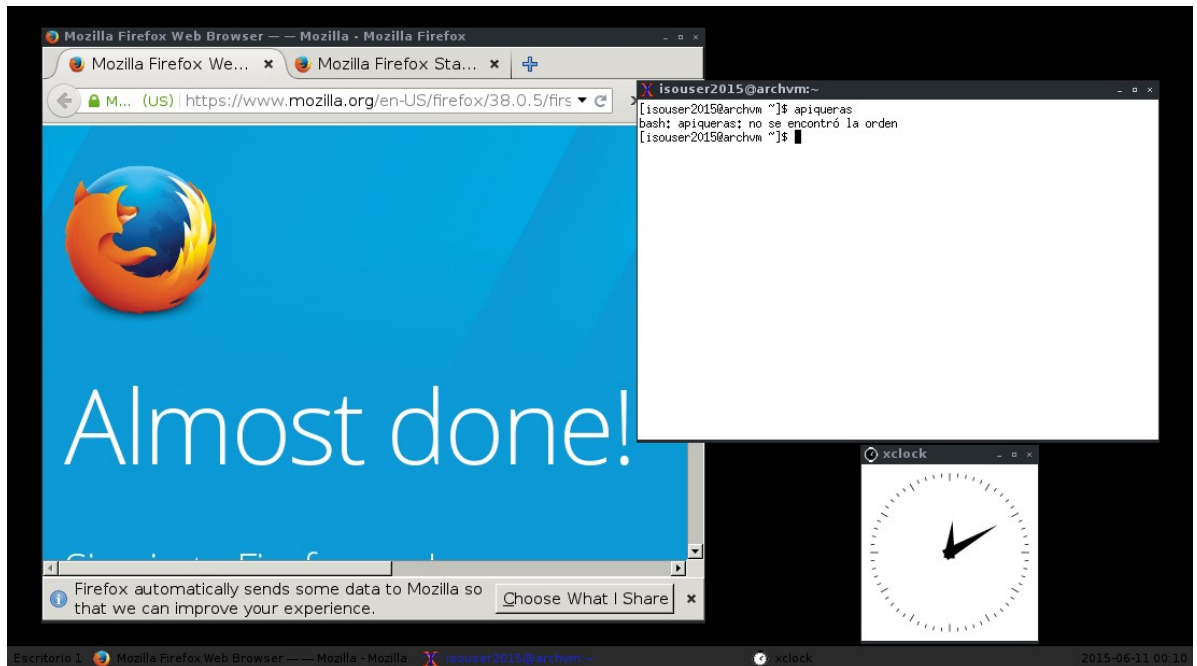
Y añadimos "pypanel &" al final del archivo

~/.config/openbox/autostart

Para verlo sin tener que reiniciar, basta ejecutar con dmenu (Alt-F2) pypanel, y nos aparecerá en la parte inferior de la pantalla.

Este panel también es configurable mediante el fichero **~/.pypanelrc**, pudiendo configurar colores, posición, tamaños, contenido y varias cosas más.

Con esto ya tendríamos un entorno de escritorio usable, y plenamente configurable. Solo quedaría instalar las aplicaciones que fuéramos a usar y nuestro sistema estaría completo.



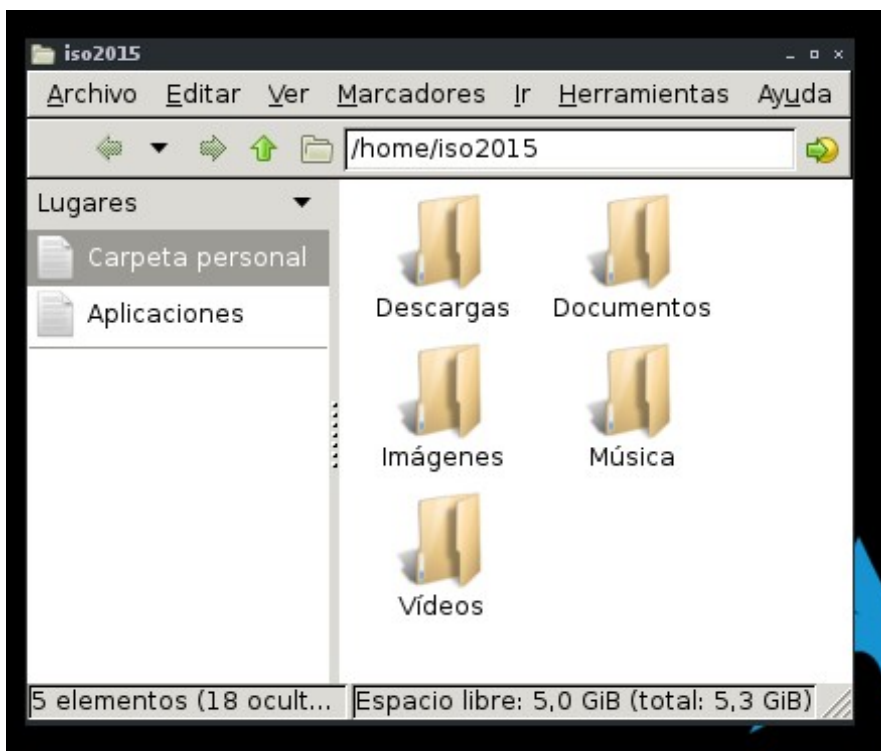
4.- Comparación con Windows

Empezamos por el modo en que se loguea un usuario en el sistema. En windows tenemos la login screen en todas sus versiones, donde podemos de manera visual seleccionar un usuario y introducir su contraseña si la tuviera. En esta instalación de linux hemos optado por no instalar ningún login manager. Los hay que aportan una interfaz de características similares a la de windows, pero sin embargo para mantener el sistema al mínimo hemos decidido que el login se haga por consola, y una vez dentro, con "startx" se inicie el entorno gráfico.

Una vez en el escritorio, lo primero que notamos es la falta de un start button como en windows (al menos hasta la llegada de windows 8). En windows todas las aplicaciones, así como accesos a los lugares más importantes del sistema están listados en el menú de inicio. En openbox, tenemos un menú similar haciendo click derecho sobre cualquier parte del escritorio. Este menú es similar a un menú contextual, donde usualmente mediante la configuración de este, cada opción es una utilidad, o una categoría que abre otro apartado de este menú. Este menú, al ser configurado por el usuario es muy flexible, ya que para cualquier cosa que sea posible hacer con un comando (practicamente todo en linux) se le puede hacer una entrada en este menú.



En las ventanas, podemos ver que salvo en el tema, son idénticas. Ambas tienen 3 botones, uno para cerrar, otro para maximizar y otro para minimizar en orden de derecha a izquierda.



En lo que el sistema difiere en gran medida es en todo el apartado de configuración. En esta distribución, configurada tal y como está, todas las configuraciones salvo la apariencia de las ventanas, el menú de aplicaciones y el fondo de pantalla se modifican mediante la consola de comandos. No tiene ninguna herramienta de

administración, ni GUI de configuración. Todo esto se debe a que la instalación ha sido realizada siguiendo la filosofía de Arch, llamada the Arch Way, o el principio KISS, cuya máxima es mantener la simplicidad (que no sencillez) al máximo, evitando GUIs y frontends que tapen lo que realmente la máquina está haciendo. Una explicación más extensa está en la pregunta 8 apartado 3, en la página 41.

Pregunta 6

Comprobar y documentar la estructura de archivos y directorios que sustenta el arranque de vuestra distribución Linux.

- 1.- Localización y descripción de `vmlinuz`, `init`, `inittab` y estructura de directorio y scripts relacionados con runlevels.

vmlinux es un archivo ejecutable enlazado estáticamente y que contiene el núcleo Linux en uno de los formatos ejecutables soportados por Linux. El archivo `vmlinux` puede utilizarse en una depuración del núcleo, para generar la tabla de símbolos u otras operaciones, o como núcleo del sistema operativo. El archivo **vmlinuz** es simplemente un archivo `vmlinux` comprimido generalmente en `zlib`.

init es un ejecutable usado en sistemas `sysVinit` como proceso padre del sistema con `pid 1`, siendo el primero en ejecutarse. En sistemas basados en `systemd`, se mantiene este ejecutable por retrocompatibilidad, pero el que hará sus funciones será otro ejecutable llamado `systemd`.

systemd es un demonio de administración de sistema diseñado exclusivamente para la API del núcleo Linux. `systemd` fue desarrollado para reemplazar el sistema de inicio (`init`) heredado de los sistemas operativos `System V` y `Berkeley Software Distribution (BSD)`. En el proceso de arranque en Linux, es el primer proceso que se ejecuta en el espacio de usuario, por lo tanto, también es el proceso padre de todos los procesos hijos en el espacio de usuario.

inittab es el fichero encargado de establecer los runlevels disponibles, para que pueda ser leído por `init`. Está localizado en la carpeta `/etc`. Este archivo solo existe si el sistema usa un sistema de arranque estilo `System V`, que no viene a ser el caso de nuestra instalación.

Directorios. En `systemd` la estructura de directorios cambia notablemente. En vez de runlevels hay targets, que estan nombrados en vez de numerados. Dentro del directorio `/usr/lib/systemd/system` están todos los equivalentes a los scripts de `init`. Dentro del directorio `/etc/systemd/system`

están los `.wants`, que son directorios con el mismo nombre que un `target` de `/usr/lib/systemd/system` que contienen vínculos simbólicos a los scripts que ejecutará el `target` con su mismo nombre.

2.- Orden runlevel.

Los niveles de ejecución («*runlevels*») son asignados a un fin específico de la instalación vanilla de Fedora; 0, 1, 3, 5, y 6; tienen una correlación de 1:1 con un específico `target` de `systemd`.

Runlevel de SysV	Target de systemd	Notas
0	<code>runlevel0.target</code> , <code>poweroff.target</code>	Detiene el sistema.
1, s, single	<code>runlevel1.target</code> , <code>rescue.target</code>	Modalidad de usuario único.
2, 4	<code>runlevel2.target</code> , <code>runlevel4.target</code> , <code>multi-user.target</code>	Definidos por el usuario. Preconfigurados a 3.
3	<code>runlevel3.target</code> , <code>multi-user.target</code>	Multiusuario, no gráfica. Los usuarios, por lo general, pueden acceder a través de múltiples consolas o a través de la red.
5	<code>runlevel5.target</code> , <code>graphical.target</code>	Multiusuario, gráfica. Por lo general, tiene todos los servicios del nivel de ejecución 3, además de un inicio de sesión gráfica.
6	<code>runlevel6.target</code> , <code>reboot.target</code>	Reinicia el sistema.
emergency	<code>emergency.target</code>	Consola de emergencia.

Pregunta 7

Comprobar y documentar la estructura de directorios del S.O. (utilizar el libro y completar)

- **/:** En los sistemas UNIX, únicamente hay una raíz, señalada con `/`. En la raíz están contenidos el resto de directorios del sistema.
- **/bin:** Es un vínculo que apunta al directorio `/usr/bin`, donde están contenidos los binarios. Se usa para retrocompatibilidad.
- **/boot:** Este directorio contiene todos los ficheros y directorios necesarios para iniciar el sistema, como el gestor de arranque con su configuración, el kernel del sistema y otros.
- **/dev:** Este directorio contiene los archivos de dispositivo utilizados para la interacción entre el SO y los periféricos, que son representados por un elemento de E/S (entrada y salida).
- **/etc:** Este directorio contiene los ficheros de configuración utilizados en todo el sistema.
- **/home:** Este directorio contiene los directorios de los usuarios del sistema.
- **/lib:** Es un vínculo que apunta al directorio `/usr/lib`, donde están contenidos las librerías del sistema. Se usa para retrocompatibilidad.
- **/lib64:** Es un vínculo que apunta al directorio `/usr/lib`, donde están contenidos las librerías del sistema. Se usa para retrocompatibilidad.
- **/mnt:** Este directorio contiene los sistemas de ficheros montados temporalmente.
- **/opt:** Contiene los llamados paquetes problemáticos, que son aquellos que no encajan claramente en el esquema de directorios GNU.
- **/proc:** Es un directorio especial. No contiene ficheros

reales, sino información del sistema (memoria del sistema, dispositivos montados, configuración del hardware, etc). Por esa razón el kernel lo usa como centro de control e información. De hecho, muchas utilidades del sistema son simplemente llamadas a archivos de este directorio. Por ejemplo, "lsmod" es lo mismo que "cat /proc/modules".

- **/root:** Directorio home del administrador del sistema. Puede sonar confuso una carpeta root dentro de root (raíz), pero históricamente, la raíz era la propia carpeta home del administrador (de ahí el nombre del usuario administrador). Para mantener las cosas ordenadas, el usuario root obtuvo su propia carpeta home, pero no se situó dentro de /home como el resto por el hecho de que suele montarse en otra partición o incluso en otro sistema, en el que root podría no tener acceso cuando solo "/" estuviera montado.
- **/run:** El punto de montaje /run se usa para montar tmpfs durante el proceso de carga del sistema operativo, accesible y modificable para todas las herramientas en cualquier momento de este. Substituye a /var/run en funciones, ya que var podría estar montada en otra partición, siendo ahora /var/run un vínculo que apunta a /run.
- **/sbin:** Es un vínculo que apunta al directorio /usr/bin, donde están contenidos los binarios. Se usa para retrocompatibilidad.
- **/srv:** El propósito principal de esta carpeta es que los usuarios puedan encontrar la localización de archivos de datos para un servicio particular.
- **/sys:** Sistema de archivos virtual proveído por el kernel para exportar información sobre subkernels, dispositivos de hardware, drivers de dispositivos...
- **/tmp:** Este directorio contiene los ficheros temporales del sistema.
- **/usr:** Contiene principalmente los binarios, documentación, librerías, archivos de cabecera, etc. Se comparte entre los usuarios del sistema, y es sólo

lectura (excepto para el gestor de paquetes). Históricamente este era el actual directorio home, que contenía tanto los binarios como los directorios de los usuarios, pero ante el crecimiento de ambos, se separaron, dejando **usr** (**U**ser **S**ystem **R**esources) como contenedor para los binarios y **home** para los datos del usuario.

- **/usr/bin**: Este directorio contiene los binarios ejecutables. Antes solo contenía los de mayor tamaño, aunque ahora se han juntado los directorios **/bin** **/sbin** **/usr/bin** y **/usr/sbin** en uno solo, unificando todos los binarios.
- **/usr/include**: Contiene archivos de cabecera (header files) para compilar código fuente.
- **/usr/lib**: Este directorio contiene las bibliotecas utilizadas por los compiladores de lenguajes de programación.
- **/usr/sbin**: Es un vínculo que apunta al directorio **/usr/bin**, donde están contenidos los binarios. Se usa para retrocompatibilidad.
- **/usr/share**: Este directorio contiene archivos que son independientes de la arquitectura del sistema (documentos, imágenes, fuentes, etc).
- **/usr/share/man**: Contiene los manuales del sistema. En el libro se indica que estos manuales están en el directorio **/usr/man**.
- **/usr/src**: Contiene código fuente del kernel.
- **/usr/local**: En este directorio (vacío por defecto en Archlinux) quiere imitar la estructura de **/usr** para paquetes de terceros. Así pues, el gestor de paquetes de ArchLinux **pacman** instalará los paquetes en **/usr**, mientras que los compilados manualmente se instalarán en **/usr/local** para evitar conflictos con los primeros.
- **/var**: Los archivos variables como logs, e-mails temporales y otros son almacenados aquí. En Arch, el árbol ABS y el caché de pacman también residen aquí.

Pregunta 8

Comprobar y documentar el funcionamiento de las distintas órdenes y utilidades para la administración de usuarios. Distinguir cuáles están disponibles desde el entorno texto y cuáles desde el entorno gráfico.

1.- Gestión de usuarios

```
# useradd -m -g [grupo_principal] -G [grupos_adicionales]  
-s [shell_de_ingreso] [nombre_de_usuario]
```

- `-m` crea el directorio home del usuario con la ruta `/home/[nombre de usuario]`, dentro de su directorio de usuario, de modo que un usuario normal puede escribir archivos, borrarlos, instalar programas, etc.
- `-g` define el nombre o el número del grupo principal del inicio de sesión del usuario; el nombre del grupo debe existir; si un número de grupo se proporciona, se debe hacer referencia a un grupo ya existente; si no se especifica, el comportamiento de `useradd` dependerá de la variable del entorno `USERGROUPS_ENAB` definida en `/etc/login.defs`.
- `-G` introduce una lista de grupos suplementarios de los que el usuario será miembro: cada grupo estará separado del siguiente por una coma, sin espacios en blanco; el valor predeterminado es que el usuario solo pertenece al grupo principal.
- `-s` define la ruta y el nombre de la shell predeterminada de inicio de sesión del usuario; después de que el proceso de arranque se ha completado, la shell de inicio de sesión predeterminada será lanzada; compruebe que el paquete de la shell elegida, si es distinta de Bash, se ha instalado.

Después de crear el usuario, usaremos el siguiente comando para crear la contraseña.

```
# passwd [nombredeusuario]
```

Para añadir más grupos adicionales al usuario usaremos el

siguiente comando.

```
# usermod -aG [grupos_adicionales] [nombredeusuario]
```

Para escribir la información adicional del usuario, se usa el comando:

```
# chfn [nombredeusuario]
```

Podemos eliminar un usuario con el siguiente comando.

```
# userdel -r [nombredeusuario]
```

Podemos ver la base de datos de usuarios con el comando:

```
$ cat /etc/passwd
```

Habrà una línea por usuario con el formato

```
account:password:UID:GID:GECOS:directory:shell
```

- **account** es el nombre del usuario.
- **password** es la contraseña del usuario. Si tiene una x indica que esta contraseña está cifrada en el archivo /etc/shadow.
- **UID** es el ID numérico del usuario.
- **GID** es el ID numérico del grupo principal del usuario.
- **GECOS** es un campo opcional que contiene información adicional del usuario; por lo general, contiene el nombre completo del usuario.
- **directory** es la carpeta \$HOME del usuario.
- **shell** es el intérprete de órdenes utilizado por el usuario (por defecto es /bin/sh).

2.- Gestión de grupos

La orden groups muestra la pertenencia al grupos.

```
$ groups [usuario]
```

Omitiendo el usuario, se muestran los grupos del usuario actual.

Se pueden listar los grupos del sistema con el comando:

```
$ cat /etc/group
```

La orden groupadd crea un nuevo grupo.


```
# groupadd [grupo]
```

Con `gpasswd` se puede agregar un usuario a un grupo (igual que con `usermod`).

```
# gpasswd -a [usuario] [grupo]
```

Para eliminar grupos usamos la orden:

```
# groupdel [grupo]
```

Puede eliminar un usuario de un grupo con el comando:

```
# gpasswd -d [usuario] [grupo]
```

Hay que reiniciar sesión para que los cambios en los grupos surtan efecto.

3.- Desde entorno gráfico

Los siguientes cinco principios constituyen lo que se conoce comúnmente como «*Arch Way*» o la Filosofía de Arch, mejor resumido por el acrónimo KISS cuyas siglas hacen referencia a «*Keep It Simple, Stupid*» («mantenlo simple, estúpido»).

- 1. Simplicidad.** Arch Linux define simplicidad como una ligera estructura de base UNIX sin agregados innecesarios, modificaciones o complicaciones, que permite a un usuario individual modelar el sistema de acuerdo a sus propias necesidades. En síntesis, una aproximación elegante y minimalista.
- 2. Precisión de código por encima de la comodidad.** La simplicidad de la *implementación*, la elegancia de código y el minimalismo deberán permanecer siempre en la máxima prioridad del desarrollo de Arch.
- 3. Centrado en el usuario.** Arch Linux tiene por objetivo ser cómoda para los usuarios GNU/Linux, dándoles un completo control y responsabilidad sobre el sistema. Los desarrolladores de Arch no dilapidan energía reinventando GUI de las herramientas del sistema; Arch se basa en un diseño sensible y una excelente documentación.
- 4. Abierto.** Arch Linux utiliza herramientas simples, que son seleccionadas o construidas con filosofía de código abierto.

5. Libre. Al mantener el sistema sencillo, Arch Linux proporciona la libertad de tomar cualquier decisión sobre el sistema.

Al haber instalado el sistema siguiendo esta filosofía, no tenemos ninguna herramienta gráfica para la gestión de usuarios, sino que esta se hace directamente desde consola.

Pregunta 9

Comprobar la existencia y el contenido de los archivos `/etc/passwd`, `/etc/group`, `/etc/shadow` y `/etc/gshadow` tras la ejecución de distintas operaciones de administración de usuarios y grupos realizadas con las órdenes suministradas a tal efecto.

1.- `/etc/passwd`

Este es el contenido de `/etc/passwd` antes de crear ningún usuario.

```
[root@archvm ~]# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/usr/bin/nologin
daemon:x:2:2:daemon:/:/usr/bin/nologin
mail:x:8:12:mail:/var/spool/mail:/usr/bin/nologin
ftp:x:14:11:ftp:/srv/ftp:/usr/bin/nologin
http:x:33:33:http:/srv/http:/usr/bin/nologin
uidd:x:68:68:uidd:/:/usr/bin/nologin
dbus:x:81:81:dbus:/:/usr/bin/nologin
nobody:x:99:99:nobody:/:/usr/bin/nologin
systemd-journal-gateway:x:191:191:systemd-journal-gateway:/:/usr/bin/nologin
systemd-timesync:x:192:192:systemd-timesync:/:/usr/bin/nologin
systemd-network:x:193:193:systemd-network:/:/usr/bin/nologin
systemd-bus-proxy:x:194:194:systemd-bus-proxy:/:/usr/bin/nologin
systemd-resolve:x:195:195:systemd-resolve:/:/usr/bin/nologin
systemd-journal-remote:x:998:998:systemd Journal Remote:/:/sbin/nologin
systemd-journal-upload:x:999:999:systemd Journal Upload:/:/sbin/nologin
[root@archvm ~]# _
```

Crearemos un usuario iso2015 perteneciente al grupo users, y vemos que el contenido de **/etc/passwd** lo refleja.

```
[root@archvm ~]# useradd -m -g users -s /bin/bash iso2015
[root@archvm ~]# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/usr/bin/nologin
daemon:x:2:2:daemon:/:/usr/bin/nologin
mail:x:8:12:mail:/var/spool/mail:/usr/bin/nologin
ftp:x:14:11:ftp:/srv/ftp:/usr/bin/nologin
http:x:33:33:http:/srv/http:/usr/bin/nologin
uidd:x:68:68:uidd:/:/usr/bin/nologin
dbus:x:81:81:dbus:/:/usr/bin/nologin
nobody:x:99:99:nobody:/:/usr/bin/nologin
systemd-journal-gateway:x:191:191:systemd-journal-gateway:/:/usr/bin/nologin
systemd-timesync:x:192:192:systemd-timesync:/:/usr/bin/nologin
systemd-network:x:193:193:systemd-network:/:/usr/bin/nologin
systemd-bus-proxy:x:194:194:systemd-bus-proxy:/:/usr/bin/nologin
systemd-resolve:x:195:195:systemd-resolve:/:/usr/bin/nologin
systemd-journal-remote:x:998:998:systemd Journal Remote:/:/sbin/nologin
systemd-journal-upload:x:999:999:systemd Journal Upload:/:/sbin/nologin
iso2015:x:1000:100:/:/home/iso2015:/bin/bash
[root@archvm ~]# _
```

2.- /etc/group

Este archivo es bastante más extenso que el anterior, ya que por defecto hay una gran cantidad de grupos creados en el sistema.

```
http:x:33:
games:x:50:
lock:x:54:
uidd:x:68:
dbus:x:81:
network:x:90:
video:x:91:
audio:x:92:
optical:x:93:
floppy:x:94:
storage:x:95:
scanner:x:96:
input:x:97:
power:x:98:
nobody:x:99:
users:x:100:
systemd-journal:x:190:
systemd-journal-gateway:x:191:
systemd-timesync:x:192:
systemd-network:x:193:
systemd-bus-proxy:x:194:
systemd-resolve:x:195:
systemd-journal-remote:x:998:
systemd-journal-upload:x:999:
[root@archvm ~]# _
```

Creamos un grupo nuevo llamado isogroup2015 y comprobamos de nuevo su contenido.

```

games:x:50:
lock:x:54:
uidd:x:68:
dbus:x:81:
network:x:90:
video:x:91:
audio:x:92:
optical:x:93:
floppy:x:94:
storage:x:95:
scanner:x:96:
input:x:97:
power:x:98:
nobody:x:99:
users:x:100:
systemd-journal:x:190:
systemd-journal-gateway:x:191:
systemd-timesync:x:192:
systemd-network:x:193:
systemd-bus-proxy:x:194:
systemd-resolve:x:195:
systemd-journal-remote:x:998:
systemd-journal-upload:x:999:
isogroup2015:x:1000:
[root@archvm ~]# _

```

Vemos que efectivamente, el grupo se ha creado con el gid 1000.

3.- /etc/shadow

```

[root@archvm ~]# cat /etc/shadow
root:$6$QMuSjfgi$rapYa4Z0GP7axu03dAxzQrMDRbDI3MS7Gbd8d9KpEzEq00PCUZpkeNd442X4Y4Q
nkjG2Nl0fuIAv5LjyQuemt1:16583::::::
bin:x:14871::::::
daemon:x:14871::::::
mail:x:14871::::::
ftp:x:14871::::::
http:x:14871::::::
uidd:x:14871::::::
dbus:x:14871::::::
nobody:x:14871::::::
systemd-journal-gateway:x:14871::::::
systemd-timesync:x:14871::::::
systemd-network:x:14871::::::
systemd-bus-proxy:x:14871::::::
systemd-resolve:x:14871::::::
systemd-journal-remote:!!:16583::::::
systemd-journal-upload:!!:16583::::::
iso2015:$6$inKgdgR.$SdB/ftf7wcj25UKk108JODLT3T3fUeALBdCeAUJLqkYPbTc0b8QF1RXq4ZQd
ajpBiVH0NkPRCS1qpmU5KP/Pz1:16596:0:99999:7:::
[root@archvm ~]# _

```

Comprobamos el contenido de dicho archivo y comprobamos que contiene el usuario creado anteriormente con una contraseña encriptada.

4.- /etc/gpasswd

Primero crearemos una contraseña para el grupo isogroup2015

con el comando **gpasswd**.

```
[root@archvm ~]# gpasswd isogroup2015
Cambiando la contraseña para el grupo isogroup2015
Nueva contraseña:
Vuelva a introducir la nueva contraseña:
[root@archvm ~]# _
```

Ahora comprobamos si en gshadow se ha añadido un campo contraseña.

```
lock:::
uid:::
x:::
dbus:x:::
network:x:::
video:x:::
audio:::
optical:::
floppy:x:::
storage:x:::
scanner:x:::
input:x:::
power:x:::
nobody:::
users:::
systemd-journal:::
systemd-journal-gateway:::
systemd-timesync:::
systemd-network:::
systemd-bus-proxy:::
systemd-resolve:::
systemd-journal-remote:!!:::
systemd-journal-upload:!!:::
isogroup2015:$6$QhW7Q//S$pu83EmY.BUoMeikpZuAWcFNPWddbNCTymqCc5ZCKF1J5DHPRNGd0IUy
5HTwgFvACqAhWgA9gX.5NB/TMPKxde.:
[root@archvm ~]# _
```

Vemos que en isogroup2015 hay después de los dos puntos una cadena de caracteres muy larga, que corresponde a la contraseña encriptada.

Pregunta 10

Prestando mucha atención realiza alguna operación de administración de usuarios y grupos editando directamente los archivos `/etc/passwd` y `/etc/group`. Comprueba, por ejemplo desde YaST, que realmente estos cambios se han producido.

Debido a que no tenemos ninguna utilidad en el entorno gráfico que nos permita hacer esto, no podemos realizar este ejercicio tal y como se pide. Sin embargo, podemos realizar la comprobación editando en el archivo `passwd` el nombre de usuario, y iniciando sesión en él. Para `group` la comprobación será cambiar el nombre a un grupo y comprobar si estos cambios se reflejan haciendo un `id` en un usuario que estuviera en este grupo.

1.- Usuario iso2015

Cambiamos el nombre de usuario de los archivos `/etc/passwd` y `/etc/shadow` de `iso2015` a `isouser2015`, y con un `su` comprobar que podemos iniciar sesión.

```
bin:x:14871::::::
daemon:x:14871::::::
mail:x:14871::::::
ftp:x:14871::::::
http:x:14871::::::
uidd:x:14871::::::
dbus:x:14871::::::
nobody:x:14871::::::
systemd-journal-gateway:x:14871::::::
systemd-timesync:x:14871::::::
systemd-network:x:14871::::::
systemd-bus-proxy:x:14871::::::
systemd-resolve:x:14871::::::
systemd-journal-remote:!!:16583::::::
systemd-journal-upload:!!:16583::::::
isouser2015:$6$inKgdgR.$SdB/ftf7wcj25UKk108JODLT3T3fUeALBdCeAUJLqkYPbTc0b8QF1RXq
4ZQda.jpBiVH0NkPRCS1qpmV5KP/Pz1:16596:0:99999:7:::
~
~
~
~
"/etc/shadow" 17 lines, 638 characters
[root@archvm ~]# su isouser2015
[isouser2015@archvm root]$_
```

2.- Grupo isogroup2015

Igual que en el ejemplo anterior, cambiaremos el nombre del grupo de los archivos `/etc/user` y `/etc/gshadow` de `isogroup2015` a `isochangedgroup2015`, y comprobamos mediante un `"id"` que en el usuario `isouser2015` se ha cambiado el nombre del grupo al que pertenecía (`isogrou2015`).

```
video:x::
audio:::
optical:::
floppy:x::
storage:x::
scanner:x::
input:x::
power:x::
nobody:::
users:::
systemd-journal:::
systemd-journal-gateway:::
systemd-timesync:::
systemd-network:::
systemd-bus-proxy:::
systemd-resolve:::
systemd-journal-remote:!!:
systemd-journal-upload:!!:
isochangedgroup2015:$6$QhW7Q//S$pu83EmY.BUvMeikpZuAWcFNPWDdbNCtymqCc5ZCKF1J5DHPR
NGd0IUy5HTwgFvACqAhWgA9gX.5NB/TMPKxde.:
"/etc/gshadow" 45 lines, 694 characters
[root@archvm ~]# id isouser2015
uid=1000(isouser2015) gid=1000(isochangedgroup2015) grupos=1000(isochangedgroup2
015)
[root@archvm ~]# _
```


Pregunta 11

Comprobar la existencia del directorio `/etc/skel` y qué archivos contiene. Crea un archivo en él y comprueba que a partir de ese instante todos los usuarios que creamos también poseerán este archivo en su directorio `~`

1.- Comprobar existencia de `/etc/skel`

Usamos un `"ls -l"` sobre `etc` para listar todos sus archivos. Filtraremos esta lista con un `grep`, para mostrar solo las líneas que contengan `skel`.

```
[root@archvm ~]# ls -l /etc/ | grep skel
drwxr-xr-x 2 root root 4096 may 28 18:41 skel
[root@archvm ~]# _
```

Podemos comprobar que existe un elemento llamado `skel`, y que efectivamente es un directorio (lo indica la letra `d` al principio de la palabra de permisos).

2.- Comprobación de su funcionamiento

Vamos a crear un pequeño archivo dentro de `skel` que debería replicarse a las nuevas cuentas de usuario.

```
[root@archvm ~]# echo "¡Bienvenido a ArchLinux!" > /etc/skel/Welcome
[root@archvm ~]# ls -l /etc/skel/
total 4
-rw-r--r-- 1 root root 26 jun 10 18:06 Welcome
[root@archvm ~]# _
```

Creamos un usuario `newisouser2015` y comprobamos su directorio `home`.

```
[root@archvm ~]# useradd -m -g users -s /bin/bash newisouser2015
[root@archvm ~]# ls -l /home/newisouser2015/
total 4
-rw-r--r-- 1 newisouser2015 users 26 jun 10 18:06 Welcome
[root@archvm ~]# cat /home/newisouser2015/Welcome
¡Bienvenido a ArchLinux!
[root@archvm ~]# _
```

Vemos que efectivamente se ha creado dicho archivo, y que contiene lo mismo que el contenido en `/etc/skel`.

Pregunta 12

Ordenes su y sudo. Observan con ejemplos el entorno de trabajo disponible según la forma de utilización de estas órdenes.

1.- su

La orden **su** (**s**ubstitute **u**ser) se utiliza para asumir la identidad de otro usuario en el sistema, normalmente root. Esto ahorra tener que cerrar la sesión en curso y volver a iniciarla después como usuario normal. En su lugar, puede iniciar sesión como otro usuario durante su sesión, iniciando una especie de «subsesión», y, luego, cerrar esta última sesión, para volver a la anterior, cuando haya terminado.

```
[isouser2015@archvm root]$ pacman -Syy sudo
error: no se puede realizar esta operación, a menos que sea superusuario.
[isouser2015@archvm root]$ su
Contraseña:
[root@archvm ~]# pacman -Syy sudo
:: Sincronizando las bases de datos de los paquetes...
core está actualizado
extra está actualizado
community está actualizado
:: Iniciando actualización completa del sistema...
resolviendo dependencias...
buscando conflictos entre paquetes....

Paquetes (11) dbus-1.8.18-1  gawk-4.1.3-1  libdbus-1.8.18-1  libtirpc-0.3.1-1
               linux-4.0.5-1  linux-firmware-20150527.3161bfa-1  lz4-130-1
               pacman-mirrorlist-20150531-1  pcre-8.37-2  zlib-1.2.8-4
               sudo-1.8.13-1

Tamaño total de la descarga:      86,43 MiB
Tamaño total de la instalación:  176,19 MiB
Tamaño neto tras actualizar:      3,43 MiB

:: ¿Continuar con la instalación? [S/n] _
```

En la imagen vemos como el sistema no deja usar el gestor de paquetes para instalar el programa sudo al no ser superusuario. Después de usar el comando su, nos logueamos como root y entonces nos deja proceder con la instalación.

2.- sudo

sudo («substitute user do») permite que un administrador del sistema delegue autoridad para dar a ciertos usuarios (o grupos de usuarios) la capacidad de ejecutar algunas

órdenes (o la totalidad) como root u otro usuario, al tiempo que permite auditar el rastro de las órdenes dadas y sus argumentos.

El archivo de configuración de sudo es `/etc/sudoers`. Sería conveniente que dicho archivo siempre se editase con la orden `visudo`. `visudo` bloquea el archivo `sudoers`, guarda las modificaciones en un archivo temporal y comprueba la gramática de ese archivo antes de copiarlo a `/etc/sudoers`.

La configuración que se suele usar, es dar al grupo wheel permisos para ejecutar sudo con la siguiente línea en la configuración:

```
%wheel          ALL=(ALL) ALL
```

Así, todos los usuarios que el administrados meta dentro del grupo wheel podrán usar el comando sudo, y tener privilegios administrativos.

En la siguiente imagen se muestra el contenido del archivo `sudoers`, modificado para que el comando sea accesible desde el grupo wheel.

```
##
## Runas alias specification
##

##
## User privilege specification
##
root ALL=(ALL) ALL

## Uncomment to allow members of group wheel to execute any command
%wheel ALL=(ALL) ALL

## Same thing without a password
# %wheel ALL=(ALL) NOPASSWD: ALL

## Uncomment to allow members of group sudo to execute any command
# %sudo ALL=(ALL) ALL

## Uncomment to allow any user to run sudo if they know the password
## of the user they are running the command as (root by default).
# Defaults targetpw # Ask for the password of the target user
# ALL ALL=(ALL) ALL # WARNING: only use this together with 'Defaults targetpw'
"/etc/sudoers.tmp" 91 lines, 2904 characters
[root@archvm ~]# _
```

Una vez configurado, podemos comprobar como un usuario no podrá usarlo hasta que el administrador lo meta en dicho grupo.

```
[isouser2015@archvm root]$ sudo pacman -Syu
We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:

    #1) Respect the privacy of others.
    #2) Think before you type.
    #3) With great power comes great responsibility.

[sudo] password for isouser2015:
isouser2015 is not in the sudoers file. This incident will be reported.
[isouser2015@archvm root]$ _
```

Metemos al usuario dentro del grupo mediante los comandos vistos anteriormente, reiniciamos su sesión (recordemos que los cambios en los grupos tienen efecto una vez cerrada la sesión) y después comprobamos que esta vez el comando que requiere privilegios precedido de sudo nos funciona correctamente.

```
[isouser2015@archvm ~]$ sudo pacman -Syu
[sudo] password for isouser2015:
:: Sincronizando las bases de datos de los paquetes...
core está actualizado      0,0   B  0,00B/s 00:00 [-----] 0%
extra está actualizado     0,0   B  0,00B/s 00:00 [-----] 0%
community está actualizado 0,0   B  0,00B/s 00:00 [-----] 0%
:: Iniciando actualización completa del sistema...
...el sistema ya está actualizado.
[isouser2015@archvm ~]$ _
```

Pregunta 13

Orden wall y archivo /etc/motd

1.- wall

La orden wall muestra un texto introducido por la entrada estandar o por un fichero en todos los usuarios logueados en ese momento.

En el ejemplo, iniciamos sesión en el tty1 con root y en el tty2 con isouser2015, y con root ejecutamos un comando wall.

```
Arch Linux 4.0.4-2-ARCH (tty1)
Hint: Num Lock on

archvm login: root
Password:
¡Bienvenido a ArchLinux!
Last login: Wed Jun 10 18:58:55 on tty1
[root@archvm ~]# wall "¡Hola a todos los usuarios conectados!"

Mensaje de difusión general (broadcast) de root@archvm (tty1) (Wed Jun 10 19:0
¡Hola a todos los usuarios conectados!
[root@archvm ~]# _
```

```
Arch Linux 4.0.4-2-ARCH (tty2)

archvm login: isouser2015
Password:
¡Bienvenido a ArchLinux!
Last login: Wed Jun 10 19:00:10 on tty1

Mensaje de difusión general (broadcast) de root@archvm (tty1) (Wed Jun 10 19:0
¡Hola a todos los usuarios conectados!
```

Podemos ver como efectivamente, en el tty2 aparece el mensaje transmitido por root.

2.- Message of the day

El contenido de `/etc/motd` (Message of the day) se muestra después de un login correcto, justo antes de la ejecución del shell del usuario.

```
[root@archvm ~]# echo "¡Bienvenido a ArchLinux!" > /etc/motd
[root@archvm ~]# _
```

Introducimos un mensaje dentro del archivo.

```
Arch Linux 4.0.4-2-ARCH (tty1)
Hint: Num Lock on
archvm login: isouser2015
Password:
¡Bienvenido a ArchLinux!
Last login: Wed Jun 10 18:55:32 on tty1
[isouser2015@archvm ~]$_
```

Y podemos observar como al inicio de la sesión de cualquier usuario se muestra el contenido de éste.

Pregunta 14

Comprobar contenido del directorio /dev. Tipos de dispositivos (de bloques, de caracteres, ...)

En el directorio /dev se almacenan los archivos de dispositivo, todos los periféricos son utilizados por Linux mediante archivos especiales. Estos controladores tienen información necesaria para utilizar el periférico, cuando el sistema operativo indica al periférico que realice una tarea el archivo de dispositivo le indica como tiene que hacerla. El contenido de /dev es el siguiente:

```

[root@archvm ~]# ls /dev
autofs          kmsg           sda            tty14          tty33          tty52          uinput
block           log            sda1           tty15          tty34          tty53          urandom
bsg             loop-control  sda2           tty16          tty35          tty54          vcs
btrfs-control   mapper        sda3           tty17          tty36          tty55          vcs1
bus             mcelog        sda4           tty18          tty37          tty56          vcs2
cdrom           mem           sda5           tty19          tty38          tty57          vcs3
char            memory_bandwidth sda6          tty2           tty39          tty58          vcs4
console         mqueue        shm            tty20          tty4           tty59          vcs5
core            net            snapshot       tty21          tty40          tty6           vcs6
cpu             network_latency snd             tty22          tty41          tty60          vcsa
cpu_dma_latency network_throughput sr0            tty23          tty42          tty61          vcsa1
cuse            null           stderr         tty24          tty43          tty62          vcsa2
disk            port           stdin          tty25          tty44          tty63          vcsa3
fd              ppp            stdout         tty26          tty45          tty7           vcsa4
full            psaux          tty            tty27          tty46          tty8           vcsa5
fuse            ptmx           tty0           tty28          tty47          tty9           vcsa6
hidraw0         pts            tty1           tty29          tty48          ttyS0          vfio
hpet            random         tty10          tty3           tty49          ttyS1          vga_arbiter
hugepages       rfkill         tty11          tty30          tty5           ttyS2          vhci
initctl         rtc            tty12          tty31          tty50          ttyS3          vhost-net
input           rtc0           tty13          tty32          tty51          uhid           zero

```

Estos dispositivos se identifican por el nombre y dos números, mayor number que coincide con todos los dispositivos del mismo tipo y minor number para diferenciar los dispositivos del mismo tipo.

```

brw-rw---- 1 root disk      8,    1 jun 10 18:33 sda1
brw-rw---- 1 root disk      8,    2 jun 10 18:33 sda2
brw-rw---- 1 root disk      8,    3 jun 10 18:33 sda3
brw-rw---- 1 root disk      8,    4 jun 10 18:33 sda4
brw-rw---- 1 root disk      8,    5 jun 10 18:33 sda5
brw-rw---- 1 root disk      8,    6 jun 10 18:33 sda6

```

En el ejemplo de la imagen, sda2 (que seria el nombre), tiene un mayor number 8, y un minor number 2.

Existen dos tipos de dispositivo:

- Bloque: Transmiten los datos en bloque, como un disco

duro por ejemplo, en la palabra de permisos empieza con una "b" y tienen un sistema de archivos montable (sda es para discos SCSI). Un ejemplo son los siguientes dispositivos:

```
brw-rw---- 1 root disk      8,  1 jun 10 18:33 sda1
brw-rw---- 1 root disk      8,  2 jun 10 18:33 sda2
brw-rw---- 1 root disk      8,  3 jun 10 18:33 sda3
brw-rw---- 1 root disk      8,  4 jun 10 18:33 sda4
brw-rw---- 1 root disk      8,  5 jun 10 18:33 sda5
brw-rw---- 1 root disk      8,  6 jun 10 18:33 sda6
```

- Carácter: Transmiten los datos carácter a carácter o como flujo de datos, como impresora por ejemplo, en la palabra de permisos empieza con una "c" y no son montables. Un ejemplo son los siguientes dispositivos:

```
crw--w---- 1 root tty        4, 14 jun 10 18:33 tty14
crw--w---- 1 root tty        4, 15 jun 10 18:33 tty15
crw--w---- 1 root tty        4, 16 jun 10 18:33 tty16
crw--w---- 1 root tty        4, 17 jun 10 18:33 tty17
crw--w---- 1 root tty        4, 18 jun 10 18:33 tty18
crw--w---- 1 root tty        4, 19 jun 10 18:33 tty19
```


Pregunta 15

Comprobar y documentar órdenes mount, umount, mkfs, df, fsck, e2fsck, dumpe2fs.

1.- mount

La orden mount sirve para montar un sistema de archivos.

```
[root@archvm ~]# mount /dev/sda2 /mnt/
[root@archvm ~]# ls /mnt
Archivos de programa    pagefile.sys    Recovery          Windows
autoexec.bat           PerfLogs        $Recycle.Bin
config.sys              ProgramData     System Volume Information
Documents and Settings  Program Files    Users
[root@archvm ~]# _
```

Las opciones de la orden mount son:

- -a: montar los ficheros que aparecen en /etc/fstab
- -f: realiza el procedimiento menos el montaje.
- -h: muestra ayuda.
- -n: monta sin escribir en /etc/mtab.
- -o: sirve para indicar una serie de opciones (noexec, nodev, user, etc).
- -r: monta como solo lectura.
- -t: indicar tipo de sistema de archivos.
- -u: actualizar puntos de montaje.
- -v: modo verbose, genera más información.
- -w: monta como lectura y escritura.

2.- umount

La orden umount sirve para desmontar sistemas de archivos.

```
[root@archvm ~]# umount /mnt
[root@archvm ~]# ls /mnt
[root@archvm ~]# _
```

Las opciones de la orden umount son:

- -a: se desmontan todos los sistemas de ficheros descritos en /etc/mtab.

- -h: muestra un mensaje de ayuda.
- -n: desmonta sin escribir en /etc/mstab.
- -t: indicar tipo de sistema de archivos.
- -r: en el caso de que el desmontaje falle, intenta volver a montarlo.
- -v: modo verbose, genera más información.

3.- mkfs

La orden mkfs sirve para formatear sistemas de archivos.

```
[root@archvm ~]# mkfs.  
mkfs.bfs      mkfs.ext3      mkfs.jfs      mkfs.reiserfs  
mkfs.cramfs   mkfs.ext4      mkfs.minix    mkfs.xfs  
mkfs.ext2     mkfs.ext4dev  mkfs.ntfs  
[root@archvm ~]# mkfs.reiserfs /dev/sda6  
mkfs.reiserfs 3.6.24  
  
Guessing about desired format.. Kernel 4.0.5-1-ARCH is running.  
Format 3.6 with standard journal  
Count of blocks on the device: 133632  
Number of blocks consumed by mkreiserfs formatting process: 8216  
Blocksize: 4096  
Hash function used to sort names: "r5"  
Journal Size 8193 blocks (first block 18)  
Journal Max transaction length 1024  
inode generation number: 0  
UUID: 5de740a7-d98d-4056-9a0e-61d1c3af1cb5  
ATTENTION: YOU SHOULD REBOOT AFTER FDISK!  
          ALL DATA WILL BE LOST ON '/dev/sda6'!  
Continue (y/n):y  
Initializing journal - 0%...20%...40%...60%...80%...100%  
Syncing..ok  
ReiserFS is successfully created on /dev/sda6.  
[root@archvm ~]# _
```

Existen variaciones de esta orden según el sistema de archivos al que queramos formatear:

- mkfs.ext2: Formateo en ext2.
- mkfs.ext3: Formateo en ext3.
- mkfs.ext4: Formateo en ext4.
- mkfs.ntfs: Formateo en ntfs.
- mkfs.reiserfs: Formateo en reiserfs
- mkfs.xfs: Formateo en xfs.

Se pueden instalar más controladores sistemas de archivos, y así dar soporte a más formatos como fat o hfs.

4.- df

La orden df es una orden para consultar los filesystems montados.

```
[root@archvm ~]# df
Filesystem          bloques de 1K Usados Disponibles Uso% Montado en
dev                  250248      0      250248   0% /dev
run                  252860     360      252500   1% /run
/dev/sda3            3997376 936372      2834908  25% /
tmpfs                 252860      0      252860   0% /dev/shm
tmpfs                 252860      0      252860   0% /sys/fs/cgroup
tmpfs                 252860      0      252860   0% /tmp
/dev/sda5             5534480 11296      5219008   1% /home
tmpfs                  50576      0       50576   0% /run/user/0
[root@archvm ~]# _
```

Las opciones de la orden df son:

- -a: incluye sistemas de archivos falsos.
- -h: mostrar los tamaños en formato legible por humanos (1K 234M 2G)
- -H: muestra tamaños en formato legible por humanos, pero utiliza potencias de 1000, no de 1024.
- -i: lista información de inodos en vez de uso de bloques.
- -l: limitar el listado a sistemas de archivos locales.
- -P: usar el formato de salida POSIX.
- -T: mostrar el tipo de sistema de archivos.

5.- fsck

El comando fsck chequea sistema de archivos, hay variaciones de este comando para especificar el sistema de archivos: fsck.ext2, fsck.ext3, fsck.ext4, fsck.reiserfs...

```

[root@archvm ~]# fsck.reiserfs -y /dev/sda6
reiserfsck 3.6.24

Will read-only check consistency of the filesystem on /dev/sda6
Will put log info to 'stdout'
#####
reiserfsck --check started at Wed Jun 10 20:21:45 2015
#####
Replaying journal: No transactions found
Checking internal tree.. finished
Comparing bitmaps..finished
Checking Semantic tree:
finished
No corruptions found
There are on the filesystem:
    Leaves 1
    Internal nodes 0
    Directories 1
    Other files 0
    Data block pointers 0 (0 of them are zero)
    Safe links 0
#####
reiserfsck finished at Wed Jun 10 20:21:45 2015
#####
[root@archvm ~]# _

```

Para discos ntfs, no existe un comando `fsck.ntfs`, sino que el paquete `ntfs-3g` nos provee de una orden llamada `ntfsfix` que hace esta funcionalidad. `fsck.ext2`, `fsck.ext3` y `fsck.ext4` son alias para `e2fsck`.

Las opciones de este comando son:

- `-a`: repara de manera automática, no pide confirmación.
- `-A`: chequea todos los dispositivos definidos en el fichero `/etc/fstab`.
- `-c`: busca bloques dañados y los agrega a la lista de bloques dañados.
- `-C`: muestra el progreso en tiempo real de un modo visual.
- `-f`: forzar la revisión.
- `-M`: no chequea sistemas montados.
- `-n`: reporta los problemas sin repararlos.
- `-r`: modo interactivo. Espera nuestra respuesta.
- `-t`: especifica el tipo o tipos de sistema de ficheros a chequear. Si lo acompañamos de la opción `-A`, solo chequearemos los sistemas que coincidan con `"fstype"`.
- `-v`: Modo verbose, genera más información.
- `-y`: si a todo.

6.- e2fsck

El comando e2fsck chequea sistemas de archivos ext.

```
[root@archvm ~]# e2fsck /dev/sda6
e2fsck 1.42.12 (29-Aug-2014)
/dev/sda6: limpio, 11/33440 ficheros, 6304/133632 bloques
[root@archvm ~]# _
```

Las opciones de este comando son:

- -c: revisar sectores que presenten problemas.
- -E: analiza la fragmentación de los archivos.
- -f: revisar una nueva partición.
- -k: añadir sectores malos a la lista de sectores malos que ya existe (combinar con -c).
- -p: reparar automáticamente.

7.- dumpe2fs

Muestra información del sistema de archivos, actúa incluso montado.

```
Last mounted on: <not available>
Filesystem UUID: 377934f3-9218-4d14-ba6f-46d1315f9b2e
Filesystem magic number: 0xEF53
Filesystem revision #: 1 (dynamic)
Filesystem features: has_journal ext_attr resize_inode dir_index filetype e
xtent flex_bg sparse_super large_file huge_file uninit_bg dir_nlink extra_isize
Filesystem flags: signed_directory_hash
Default mount options: user_xattr acl
Filesystem state: clean
Errors behavior: Continue
Filesystem OS type: Linux
Inode count: 33440
Block count: 133632
Reserved block count: 6681
Free blocks: 127328
Free inodes: 33429
First block: 0
Block size: 4096
Fragment size: 4096
Reserved GDT blocks: 32
Blocks per group: 32768
Fragments per group: 32768
Inodes per group: 6688
Inode blocks per group: 418
Flex block group size: 16
```

Las opciones de este comando son:

- -b: imprimir los bloques que se reservan en el sistema de archivos.
- -f: fuerza para mostrar un sistema de archivos a pesar de

que pueden tener algunas banderas de características del sistema de archivos que `dumpe2fs` no puede entender (y que puede causar alguna de la pantalla `dumpe2fs` de ser sospechoso).

- `-h`: sólo mostrar la información del superbloque y no cualquiera de la información del grupo de bloque de detalle descriptor.
- `-i`: mostrar los datos del sistema de archivos de un archivo de imagen creado por `e2image` , utilizando el dispositivo como la ruta al archivo de imagen.
- `-x`: imprimir los detalles del grupo números de bloque de información en formato hexadecimal.

Pregunta 16

Sistema de archivos ext.

1.- ext

El **sistema de archivos extendido** (**extended file system** o **ext**), fue el primer sistema de archivos creado específicamente para el sistema operativo Linux. Fue diseñado por Rémy Card para vencer las limitaciones del sistema de archivos MINIX. Fue reemplazado tanto por ext2 como xiafs, entre los cuales había una competencia, que finalmente ganó ext2, debido a su viabilidad a largo plazo. Actualmente linux ya no da soporte a este sistema de archivos.

2.- ext2

ext2 (*second extended filesystem* o "segundo sistema de archivos extendido") es un sistema de archivos para el kernel Linux. Fue diseñado originalmente por Rémy Card. La principal desventaja de ext2 es que no implementa el registro por diario (en inglés *Journaling*). En la actualidad, aun se da soporte a este sistema de archivos, aunque se recomienda migrar a ext3 o ext4.

Características:

- Tamaño máximo de bloque: 4KiB
- Dimensión máxima de archivo: 2TiB.
- Máximo número de archivos: 10^{18} .
- Tamaño máximo del nombre de archivo: 255 caracteres.
- Tamaño máximo del volumen: 16TiB.
- Caracteres permitidos en nombres de archivo: Cualquiera excepto NULL y '/'.

3.- ext3

ext3 (*third extended filesystem* o "tercer sistema de archivos extendido") es un sistema de archivos con *registro por diario* (journaling). Fue el sistema de archivos más

usado en distribuciones Linux, aunque en la actualidad ha sido reemplazado por su sucesor, ext4.

Las principales mejoras son:

- Incluye journaling
- El tamaño de bloque se puede aumentar a 8KiB, aumentando el tamaño máximo del sistema de ficheros a 32TiB

4.- ext4

ext4 (*fourth extended filesystem* o «cuarto sistema de archivos extendido») es un sistema de archivos transaccional (en inglés *journaling*), anunciado el 10 de octubre de 2006 por Andrew Morton, como una mejora compatible de ext3.

Las principales mejoras son:

- Soporte de volúmenes de hasta 1024 PiB.
- Soporte añadido de *extent*.
- Menor uso del CPU.
- Mejoras en la velocidad de lectura y escritura.

Problemas encontrados

El único problema encontrado ha sido el cambio de sistema de SystemVinit a systemd en la mayoría de sistemas linux, incluida la distribución elegida. Esto ha implicado una investigación más extensa comparando el sistema explicado en clase y el nuevo sistema, buscando las equivalencias, similitudes y diferencias.

Opinión personal

Alejandro Piqueras Sastre

Como usuario habitual de linux y de esta distribución en concreto, me ha parecido un trabajo relativamente sencillo. Sin embargo, he repasado algunos conceptos olvidados, y aprendidos algunos nuevos, cosa que siempre es de agradecer.

Daniel Rosselló Sanchez

Ha sido interesante experimentar con la instalación de una distribución como esta, prescindiendo de herramientas gráficas. Me ha permitido aprender mejor como funciona un linux por dentro.

Puntos de discusión

No ha habido ningún punto de discusión durante el desarrollo del trabajo.

Bibliografía

Webs

<https://wiki.archlinux.org/>

<http://es.wikipedia.org/>

<https://www.iceflatline.com/2009/09/how-to-dual-boot-windows-7-and-linux-using-bcdedit/>

Otros

Orden **man** de linux