# 102 - Instalación de Linux y administración de paquetes

## 1 - Dimensionar las particiones de disco

En Linux se accede a todos los sistemas de archivos en particiones por medio de montaje, en el cuál ese sistema queda "montado" en un directorio (punto de montaje).

**Sistema de archivos raíz**

El principal punto de monta es raíz "/" que es el primer punto de montaje que tendrá el equipo, esta partición queda identificada con el cód hex 83 (0x83 Linux Native) y ya podrán copiarse los archivos del sistema.

**Orden de montaje a partir del Boot**

El cargador de Boot carga el kernel y transmite la información sobre la localización de /. Se monta la raíz y los demás dispositivos de acuerdo a las instrucciones de /etc/fstab.

**Linux exige al menos dos particiones**

- *Partición Swap*: Es un espacio en disco que actúa como memoria adicional evitando la ocupación total de la RAM.
- *Otros puntos de montaje*: Aunque todo puede alojarse en / podemos crear particiones diferentes para algunos directorios
  - /var: Contiene cola de mails, impresión y DB bastante usada. También LOGS!
  - /tmp: Espacio temporal usado por programas, si se pone en otra partición evitamos colapso del directorio raíz /.
  - /home: contiene los datos personales de los usuarios, en otra partición limitamos el espacio.
  - /boot: contiene el kernel y archivos de bootloader Grub, es necesario cambiarlo a otra partición si la máquina requiere que el kernel esté antes del cilindro 1024 dsel hdd y cuando el bootloader no es capaz de trabajar con el sistema de archivos de /
  - usr: programas, código fuente y documentación. En otro dispositivo reduce la intensidad de acceso al mínimo.
  - Algunos directorios NO deben estar fuera de /, serían: /etc, /bin, /sbin, /dev, /proc y /sys.

## 2 - Instalar el gestor de arranque

El bootloader es el responsable de localizar y cargar el kernel y desempeña una etapa intermedia entre el final del procedimientos de la BIOS y el inicio del SO.

Después de terminar los chequeos BIOS, carga en memoria los datos de la MBR del disco definido como inicio en la BIOS.

El MBR ocupa el primer sector del disco que contiene la tabla de particiones y el cargador de arranque, los mas usados Grub y Lilo.

## LILO

Es el cargador mas tradicional, ya sólo lo usa slackware, se divide en:

- Lilo: cargador que se instala en la MBR y activa la segunda parte del cargador de arranque localizado en /boot/boot.b
- /etc/lilo.conf: archivo de configuración
- /sbin/lilo: comando que lee las conifguraciones e instala el cargador en la MBR

Lilo guarda la MBR el cargador de boot con la info de localización del Kernel y el directorio raíz.

**Opciones de lilo.conf**:

- boot --> /dev/hda o /dev/sda
- prompt --> Muestra al usuarioun menú para selección del SO
- image --> localización del archivo de kernel.
- other --> indica la partición de otro SO
- label --> Nombre para mostrar en el menú inicio
- root --> indica la partición raíz para el kernel indicado
- read-only --> partición montada solamente como lectura en la primera etapa del boot
- appen --> parámetros adicionales del kernel
- message --> indica mediante un archivo de texto el mensaje que se mostrará en el menú boot.
- delay --> la pausa en décimas de segundo para que user pulse Tab y se active el menú.
- timeout --> pausa para que se cargue la opción default
- vga --> Valor numérico que especifica las preferencias visuales del terminal

Después de modificar el fichero lilo.conf es importante reinstalar el cargador del MBR con el comando lilo.

## GRUB

Es el mas usado actualmente, también se instala en la MBR (/sbin/grub-install) y carga las configuraciones de /etc/grub/menu.lst (actualmente con grub2 /boot/grub/grub.cfg y el fichero de configuración en /etc/default/grub)

En grub2 (el actual) crearemos el fichero grub.cfg mediante el siguiente comando:

```
# La opción -o es para indicar el nombre y ubicación del fichero generado
grub-mkconfig -o /boot/grub/grub.cfg
```

**Opciones generales:**

- default: SO que iniciará por defecto 0.
- timeout: tiempo de espera en seg para iniciar el boot.

**Opciones individuales:**

- title: nombre del ítem
- root: localización del cargador de la segunda etapa y del kernel
- kernel: path al kernel
- ro: montar inicialmente en sólo lectura
- initrd: camino para la imagen initrd

## Dispositivos de inicio alternativo

Es posible iniciar el sistema con un liveCD y modificar esos archivos.

Una buena práctica es hacer backup del MBR con:

```
dd if=/dev/hda of=mbr.backup bs=1 count=512
```

Podemos restaurar la copia con:

```
dd if=mbr.backup of=/dev/hda
```

# 3 - Control de bibliotecas compartidas

Funciones comunes de diferentes programas se almacenan en bibliotecas. Para compilar un programa es necesario que pueda localizar las bibliotecas que necesita y así crear vínculos entres sus funciones y las bibliotecas.

Este vínculo puede ser estático (quedan incluidas en el programa compilado) o dinámico(quedan mapeados en bibliotecas externas).

**Identificar bibliotecas compartidas:**

Para conocer las bibliotecas necesarias por un programa se usa el comando **ldd**
Ejemplo:

```
ldd /usr/bin/vi
```

Si apareciera un "NOT FOUND", es que hay un error y no encuentra la biblioteca.

**Localización de las bibliotecas:**

El programa responsable de la carga de la biblioteca y de vincularla es **ld.so** que es invocado cada vez que un programa necesita una función de biblioteca externa.

El ld.so consigue localizar con la ayuda del mapeo encontrado en **/etc/ld.so.cache** (es un binario).

Las bibliotecas estándar del sistema suelen encontrarse en /lib y /usr/lib, si se añaden otros directorios se deben incluir en /etc/ld.so.conf o en /etc/ld.so.conf.d/

Actualiza la caché con las conficuraciones de ld:

```
ldconfig
```

La variable de entorno LD_LIBRARY_PATH contiene las rutas a las librerías del sistema, para añadir podemos usar el .profile del usuario o el general y realizar algo como:

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/home/ziko/work
```

Fijarse que añado la anterior LD_LIBRARY_PATH antes de exportar, pues si no sólo usaría la que acabo de poner.

# 4 - Utilización del sistema de paquetes DEBIAN

Permite la instalación si que el usuario se deba preocupar por las bibliotecas u otros programas necesarios para la ejecución del mismo ya que cada .deb contiene la información de sus dependencias.

Las principales herramientas de administración de paquetes .deb son:

- **dpkg**: para instalar paquete individuales.
- **apt-get**: busca paquete en repositorios remotos y los instala además de sus dependencias.
- **aptitutde**: alternativa a apt-get

La gran ventaja es que las dependencias del programa a instalar las baja automáticamente y las instala sin problemas.

**Repositorios:**

Para la resolución automática de dependencias es necesario indicar el origen de los paquetes, estos orígenes estan determinados en **/etc/apt/sources.list** y en algunos casos en **/etc/apt/sources.list.d/**

Cada linea determina un repositorio y cada distribución tiene repositorios propios oficinas y no oficiales.

Las diferentes opciones (aptitude es lo mismo sustituyendo el comando apt-get):

- Actualizar con los nuevos repositorios o orígenes.

```
apt-get update
```

- Busqueda de programas

```
apt-cache search nombre_paquete

# Descripción y detalles de un paquete seleccionado:
apt-cache show nombre_programa
```

- Instalación de programas

```
apt-get install nombre_paquete
```

- Eliminación

```
# Simple:
apt-get remove nombre_paquete

# Paquete y archivos configuración:
apt-get remove --purge nombre_paquete
```

- Actualización programas:

```
apt-get upgrade
```

- Inspección de paquetes con dpkg

```
# Estado paquete:
dpkg -l nombre_paquete

# Busca que paquete instaló el archivo x:
dpkg -S nombre_archivo_x

# Lista archivos instalados por un paquete:
dpkg -L nombre_paquete

# Lista del contenido del paquete seleccionado:
dpkg --contents paquete.deb
```

# 5 - Utilización del sistema de paquetes RPM y YUM / DNF

**RPM**

Es el comando de administración de paquetes rpm, semejante a dpkg.

Instalación de un paquete .rpm :

```
rpm -ivh nombre_paquete.rpm
```

Las diferentes opciones de RPM son:

- -i o --install: instala un paquete
- -U o --update: actualiza el paquete
- -F o --freshen: actualiza un paquete si estuviese instalado.
- -V o --verify: verifica tamaño, MDB, permisos, etc.
- -q o --query: investiga los paquetes y archivos
- -e o --erase: desinstala un paquete
- --nodeps: instala sin verificar dependencias
- --force: fuerza la instalación/actualización
- --test: muestra como sería la instalación pero no la hace
- --requires: la opción q muestra las exigencias del paquete
- --whatrequires: con opción 1 muestra los programas que dependen del paquete.

Hay subopciones que modifican su comportamiento, sobre todo con -q, ejemplo -qc: lista archivos de configuración, el mas interesante es el -q en el fondo.

```
# Consultamos el paquete nano en el sistema
rpm -q nano
>> nano-5.6.1-5.el9.x86_64

# Así me devolverá el fichero de configuración
rpm -qc nano
>> /etc/nanorc

# Así me devolverá los archivos de documentación
rpm -qd nano
>> /usr/share/doc/nano/AUTHORS
>>/usr/share/doc/nano/COPYING
>>/usr/share/doc/nano/ChangeLog
.... (hay mas, pero corto aquí el listado)

# Así me listará todos los archivos
rpm -qd nano
>>/etc/nanorc
>>/usr/bin/nano
>>/usr/bin/rnano
.... (hay mas, pero corto aquí el listado)
```

Con el comando rmp2cpio mostramos el contenido del archivo rpm:

```
rmpm2cpio nombre_paquete.rpm | cpio -t

# Para extraer info usaremos cpio de la siguiente manera:

rmpm2cpio nombre_paquete.rpm | cpio -ivd '*.pdf'

# Con eso extraemos los pdfs del nombre_paquete.rpm
```

**YUM**

El archivo de configuración es **/etc/yum.conf** y algunas de sus opciones son:

- cachedir --> directorio de almacenamiento de los paquetes (por defecto /var/cache/yum)
- keepcache --> 1 mantiene los archivos tras la instalación, 0 no los mantiene.
- reposdir --> lista de directorios donde yum buscará repositorios, por defecto /etc/yum.repos.d
- debuglevel --> nivel de mensaje de aviso del 0 al 10, por defecto el 2
- errorlevel -->nivel de mensaje de error del 0 al 10, por dfecto el 2
- logfile --> ruta al log de yum
- gpgcheck --> determina si yum debe verificar GPG, 1 o 0

Funciones del comando:

```
# Localiza un paquete
>> yum search nombre_paquete

# Instala un paquete
>> yum install nombre_paquete

# Desinstala un paquete
>> yum remove/erase nombre_paquete

# Localiza un paquete que ofrece un recurso determinado
>> yum provides/whatprovides recurso_a_buscar

# Actualiza los paquetes
>> yum update

# Igual que update pero también actualiza la distribución
>> yum upgrade
```

Con el siguiente comando, podemos descargarlo sin tener que instalar:

```
yumdownloader nombre_paquete
```

**ZYPPER**

Se trata del package manadger de Suse Linux, no me voy a extender en esta explicación, pues funciona similar a yum, añado el link de documentación de Suse linux.

https://documentation.suse.com/smart/systems-management/html/concept-zypper/index.html

# 6 - Virtualización en Linux

Estos serían los programas de virtualización mas usados.

| Software | Breve explicación |
|---|---|
| **KVM** | muy recomendable, funciona como hypervisor |
| **XEN** | muy antiguo, pero tiene mas seguridad |
| **proxmox** | dentro de las opciones opensource es la mas profesional |
| **VMWARE ESXI** | sistema de pago, hasta ahora el mas usado por empresas, pero con problemas desde la adquisición por parte de DELL con las licencias |
| **VirtualBox** | a nivel usuario muy bueno |

**D-Bus Machine ID**

Muchas instalaciones de Linux utilizarán un número de identificación de máquina generado en el momento de la instalación, llamado D-Bus machine ID.

El siguiente comando se puede usar para validar que existe una ID de máquina D-Bus para el sistema en ejecución:

```
dbus-uuidgen --ensure
```

Si no se muestran mensajes de error, existe una ID para el sistema. Para ver la ID actual de la máquina D-Bus, ejecute lo siguiente:

```
dbus-uuidgen --get
>> b59994c2eec7435ba817ce5e1286b6cf
```

La ID de la máquina se encuentra en /var/lib/dbus/machine-id.

**Acceso seguro (SSH)**

La forma mas habitual de acceso seguro sería mediante la creación de clave privada, clave pública, así que deberíamos ejecutar el siguiente comando para crear el par.

```
ssh-keygen
```

Se seguirán los pasos que va solicitando y se almacenarán las clave por defecto en ~/.ssh/

En ese momento ejecutaríamos:

```
ssh-copy-id -i <public_key> user@cloud_server
```

Copiando de esta manera nuestra clave pública al servidor destino, dicha clave se registrará en el archivo ~/.ssh/authorized_keys del servidor.