Introducción a GNU/Linux (continuación)

Explicación de práctica

Introducción a los Sistemas Operativos
Conceptos de Sistemas Operativos

Facultad de Informática Universidad Nacional de La Plata

2025

Características - Configuración de discos

- Configuración de discos IDE (*Integrated Device Electronics*):
  - Master o Slave
  - Primer y Segundo bus IDE
- Denominación de los discos basada en los buses:
  - **/dev/hd<u>a</u>**: configurado como Master en el 1º bus IDE
  - **/dev/hd<u>b</u>**: configurado como Slave en el 1º bus IDE
  - **/dev/hd<u>c</u>**: configurado como Master en el 2º bus IDE
  - **/dev/hd<u>d</u>**: configurado como Slave en el 2º bus IDE
- Particiones primarias → 1 a 4
- Particiones lógicas → de 5 en adelante

Características - Configuración de discos

- Configuración de discos SCSI (*Small Computer System Interface*):
  - se basa en *LUN*.
- Denominación de los discos basada en la identificación de los buses:
  - **/dev/sd<u>a</u>**
  - **/dev/sd<u>b</u>**
  - **/dev/sd<u>c</u>**
  - **/dev/sd<u>d</u>**
  - **...**
- Particiones primarias:
  - Se numeran de la 1 a la 4.
  - Sólo estas se pueden marcar como activas (booteables).
- Particiones extendidas:
  - Sus particiones lógicas se numeran a partir de la 5.

Características - Configuración de discos

- Configuración de discos SATA (*Serial Advanced Technology Attachment*):
  - Denominación de los discos basada en la identificación de los buses, al igual que SCSI (/dev/sda, /dev/sdb, …).
- Particiones primarias:
  - Se numeran de la 1 a la 4.
  - Sólo estas se pueden marcar como activas (booteables).
- Particiones extendidas:
  - Sus particiones lógicas se numeran a partir de la 5.

Características - Configuración de discos

- Nueva nomenclatura utilizada:
  - Con la evolución de las distribuciones GNU/Linux, se comenzó a utilizar **"udev"** (*.rules*) como gestor de dispositivos:
    - Su función es controlar dinámicamente los archivos que hay en /dev, solo en base al hardware detectado.
    - Soporta **Persistent Device Naming**.
    - Motiva su uso, el no poder garantizar que tras distintos arranques del SO, los dispositivos se sigan llamando de la misma manera.
    - Reemplaza a *devfs* y *hotplug*.
    - No se basa en **Major** y **Minor Number**.
    - Se basa en eventos y permite que nuevos dispositivos sean agregados luego del arranque.

Características - Configuración de discos

- Desde Debian/Squeeze todos los dispositivos llamados **hdX** se pasaron a denominar **sdX**.
- Por esta y otras razones se adoptan 4 mecanismos nuevos para nomenclar:
  - Nombres persistentes por **UUID** (Universal Unique Identifier):

```
$ ls -l /dev/disk/by-uuid/
2d781b26-0285-421a-b9d0-d4a0d3b55680 -> ../../
sda1
31f8eb0d-612b-4805-835e-0e6d8b8c5591 -> ../../
sda7
```

  - Utilizando **labels**

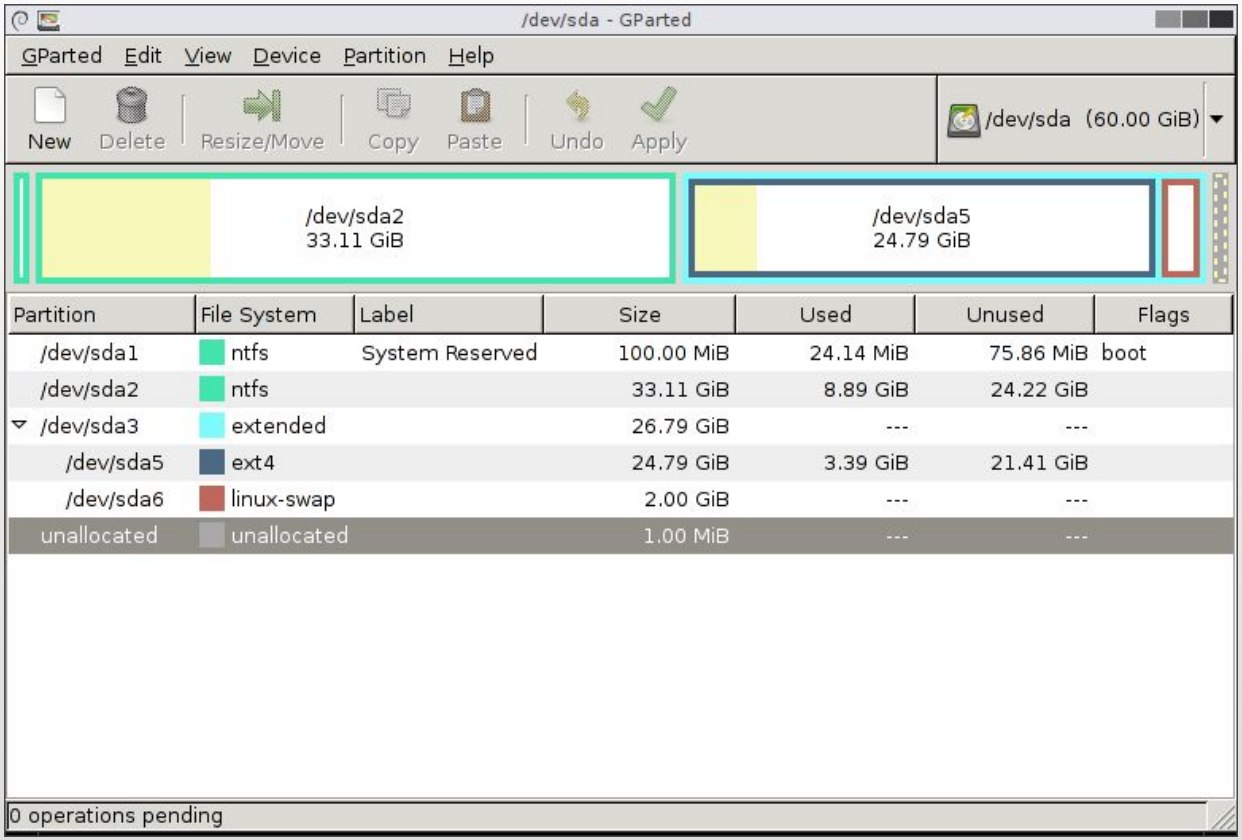
```
$ ls -l /dev/disk/by-label
data -> ../../sdb2
data2 -> ../../sda2
```

Soporte de instalación

- Existen diversos modos de instalar GNU/Linux:
  - Debemos tener en cuenta la arquitectura de hardware:
    - **amd64**: Arquitectura de 64 bits
    - **arm / armel**: Advanced Risc Machine
    - **i386**: Arquitectura de 32 bits
    - **ia64**: Intel Itanium o Intel Architecture-64
    - Otras muchas más
  - Podemos instalarlo desde una imagen ISO descargada de la web.
  - Podemos instalarlo desde un USB.
    - UNetbootin permite crear instaladores o LiveCD utilizando unidades USB.

Herramientas para particionar

- El particionado de un disco se lo puede realizar mediante:
  - Software **destructivo**: *fdisk*.
  - Software **no destructivo**: *fips*, *gparted*.



GNU/Linux: Características

- No existe el concepto de *extensión* en el nombre de un archivo.
- Los subdirectorios no se separan con el caracter \ (contrabarra) - se usa / (barra).
- Es case sensitive.
- En la línea de comandos (*CLI*), los elementos o tokens de una instrucción se separan utilizando espacios en blanco. Por ejemplo, entre un comando y sus parámetros debemos dejar obligatoriamente un espacio en blanco.
- Existe una separación entre el entorno gráfico y el de texto (*tty*).

*GNU/Linux: Editor de texto **vim***

- Presente en la mayoría de las distribuciones.
- Posee 3 modos de ejecución:
  - Modo Insert (**i**)
  - Modo Visual (**v**)
  - Modo de Órdenes o Normal (**Esc**)
- Se le puede enviar una serie de comandos útiles:
  - **w**: escribir cambios
  - **q** o **q!**: salir del editor
  - **dd**: cortar
  - **y**: copiar al portapapeles
  - **p**: pegar desde el portapapeles
  - **u**: deshacer
  - **/**frase: busca "frase" dentro del archivo

GNU/Linux: Otros editores de texto

- El editor que se usa es siempre una decisión personal.
- La curva de aprendizaje de editores como **vim** o **emacs** puede ser un desafío al comienzo.
- Si resulta muy complejo para comenzar, siempre se pueden utilizar alternativas menos potentes pero más sencillas, como:
  - **nano**
  - **micro**
  - **ne** (nice editor)

Usuarios

- Todo usuario debe poseer credenciales para acceder al sistema
  - **root**: es el administrador del sistema (superusuario)
  - usuarios regulares del sistema
  - usuarios regulares del sistema que pueden realizar tareas de superusuario (**/etc/sudoers**)
- Los usuarios se identifican con un nombre de usuario, que es único en el sistema.
- Cada usuario pertenece a uno o más *grupos*.
- Los usuarios pueden tener un directorio personal o *home*.

Usuarios

- La información de los usuarios se almacena en diferentes archivos de configuración:

```
$ cat /etc/passwd
ndelrio:x:2375:500:Nico del Rio,,,,Usuarios:/
home/admins/ndelrio:/bin/bash
```

```
$ cat /etc/group
infraestructura:x:500:
```

```
$ cat /etc/shadow
ndelrio:$1$HamkgCYM$TtgfLJLplItxutaiqh/u9
/:13273:0:99999:7:::
```

Usuarios

- Comandos para el manejo de los usuarios y grupos del sistema:
  - **useradd** nombre_usuario:
    - Agrega el usuario
    - Modifica los archivos /etc/passwd
    - Alternativa → **adduser**
  - **passwd** nombre_usuario:
    - Asigna o cambia la contraseña del usuario
    - Modifica el archivo /etc/shadow
  - **usermod** nombre_usuario:
    - **-g**: modifica grupo primario (Modifica /etc/passwd)
    - **-G**: modifica grupos adicionales (Modifica /etc/group)
    - **-d**: modifica el directorio *home* (Modifica /etc/passwd)
  - **userdel** nombre_usuario: elimina el usuario
  - **groupdel** nombre_grupo: elimina el grupo

Permisos

- Se aplican a nivel de sistema de archivos (directorios y archivos).
- Existen 3 posibles permisos: R, W y X.
- Se suelen expresar en octal acorde a la siguiente tabla:

Permiso	Valor	Octal
Lectura	R	4
Escritura	W	2
Ejecución	X	1

- Las combinaciones de permisos se expresan como la suma de los valores octales de cada permiso que se desea incluir:
  - **W + X** = 2 + 1 = **3**
  - **R + W** = 4 + 2 = **6**

Permisos

- Cada archivo o directorio del sistema de archivos tiene un usuario dueño y un grupo dueño.
- Los permisos pueden definirse sobre las siguientes agrupaciones de usuarios:
  - **Usuario**: aplicables al usuario dueño (**U**).
  - **Grupo**: aplicables a cualquier usuario perteneciente al grupo dueño que no sea el usuario dueño (**G**).
  - **Otros**: aplicables a cualquier otro usuario que no pertenezca a las agrupaciones anteriores (**O**).
- Los permisos de un archivo o directorio se gestionan mediante el comando **chmod**:

```
$ chmod 755 /tmp/script
```

Entorno

- Algunos comandos útiles:
  - **ls**
  - **cd**
  - **mkdir**
  - **rmdir**
  - **rm**
  - **mv**
  - **cp**
  - **man**
  - **info**

Proceso de arranque: ***Bootloader***

- El Bootloader o cargador de arranque es un programa que permite cargar e iniciar el Sistema Operativo. Puede llegar a cargar un entorno previo a iniciar el Sistema Operativo.
- Generalmente se utilizan los cargadores multietapas, en los que varios programas pequeños se van invocando hasta lograr la carga del Sistema Operativo.
- En cierto sentido, el código del *BIOS/UEFI* forma parte del bootloader, pero el concepto está m´as orientado al código que reside en el *Master Boot Record* (MBR, 512B).
- El MBR está formado por el *MBC* (446B) y la *Tabla de Particiones* (64B).
- Sólo el MBC del *Primary Master Disk* es tenido en cuenta.
- El MBR existe en todos los discos, ya que contiene la tabla de particiones de cada uno.

Proceso de arranque: ***SysV init***

1. Se empieza a ejecutar el código del BIOS.
2. El BIOS ejecuta el POST.
3. El BIOS lee el sector de arranque (MBR).
4. Se carga el gestor de arranque (MBC).
5. El bootloader carga el *kernel* y el *initrd* (*initial ram disk*).
6. Se monta el *initrd* como sistema de archivos raíz y se inicializan componentes esenciales (por ejemplo, el *scheduler*).
7. El Kernel ejecuta el proceso *init* y se desmonta el *initrd*.
8. Se lee el /etc/inittab.
9. Se ejecutan los scripts apuntados por el *runlevel 1*.
10. El final del runlevel 1 le indica que vaya al runlevel por defecto.
11. Se ejecutan los scripts apuntados por el runlevel por defecto.
12. El sistema está listo para ser usado.

*Proceso de arranque: SysV init - El proceso **init***

1. Su función es cargar todos los subprocesos necesarios para el correcto funcionamiento del Sistema Operativo.
2. El proceso *init* (ejecutado desde /sbin/init) posee el PID 1.
3. En SysV init se lo configura a través del archivo /etc/inittab.
4. No tiene padre y es el padre de todos los procesos (**pstree**).
5. Es el encargado de montar los filesystems y de hacer disponible los demás dispositivos.

*Proceso de arranque: SysV init - **Runlevels***

- Es el modo en que arranca GNU/Linux (por defecto: Runlevel 3 en Redhat y Runlevel 2 en Debian).
- El proceso de arranque se divide en niveles.
- Cada runlevel es responsable de iniciar o parar una serie de servicios, ya sea al entrar al Runlevel (arranque) o al salir de éste (apagado).
- Acorde al estándar, existen 7 (numerados del 0 al 6):
  - 0 → halt (parada o apagado).
  - 1 → single-user mode (modo monousuario).
  - 2 → multi-user without network support (multiusuario sin soporte de red).
  - 3 → multi-user console mode (modo multiusuario en consola).
  - 4 → N/A (no se utiliza).
  - 5 → X11 (modo multiusuario con entorno gráfico basado en X.org).
  - 6 → reboot (reinicio).

*Proceso de arranque: SysV init - **Runlevels***

- Se encuentran definidos en el archivo /etc/inittab:
  id:runlevels:acción:proceso
  - **id**: identifica la entrada en inittab (1 a 4 caracteres).
  - **runlevels**: el/los runlevels en los que se realiza la acción
  - **acción**: indica cómo se ejecutará **proceso**
    - **wait**, **initdefault**, **ctrlaltdel**, **off**, **respawn**, **once**, **sysinit**, **boot**, **bootwait**, **powerwait**, etc.
  - **proceso**: el comando exacto que será ejecutado.

```
$ cat /etc/inittab
id:2:initdefault: si::sysinit:/etc/init.d/rcS
ca::ctrlaltdel:/sbin/shutdown -t3 -r
```

*Proceso de arranque: SysV init - **Runlevels***

- Los scripts que se ejecutan se suelen guardar en **/etc/init.d**.
- En **/etc/rcX.d** (donde **X** es el número de runlevel entre 0 y 6) se hacen links simbólicos a los archivos que hay en **/etc/init.d**.
- Los nombres de los links siguen este patrón:

  [S|K]<orden><nombre>

  S indica que se debe iniciar el script (se invoca con el argumento **start**).

  K indica que se debe para el script (se invoca con el argumento **stop**).

  <orden> es un valor numérico en dos dígitos para garantizar el orden de ejecución de los scripts. Puede verse como una prioridad.

  <nombre> es el nombre lógico con que identificamos el script, no es relevante para la ejecución en sí.

```
$ ls -1 /etc/rcS.d/
S55urandom S70x11-common
```

*Proceso de arranque: SysV init - comando **insserv***

- Se utiliza para administrar el orden de los enlaces simbólicos de los directorios **/etc/rcX.d**, resolviendo las dependencias de forma automática.
- Utiliza cabeceras en los scripts de /etc/init.d que permiten especificar la relación con otros scripts rc → LSBInit (Linux Standard Based Init).
- Es utilizado por *update-rc.d* para instalar/remover los links simbólicos.

Proceso de arranque: SysV init - comando ***insserv***

- Las dependencias se especifican mediante *facilities* → **Provides** keyword.
- Las facilities que comienzan con $ se reservan para el sistema (**$syslog**).
- Los scripts deben cumplir con el estándar *LSB init*:
  - Proveer al menos **start**, **stop**, **restart**, **force-reload** y **status**.
  - Retornar un código de salida apropiado.
  - Declarar sus dependencias.

*Proceso de arranque: SysV init - comando **insserv***

- Ejemplo de un encabezado de script LSB init:

```
### BEGIN INIT INFO
# Provides:          scriptname
# Required-Start:    $remote_fs $syslog
# Required-Stop:     $remote_fs $syslog
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: Start daemon at boot time
# Description:       Enable service provided by daemon.
### END INIT INFO
```

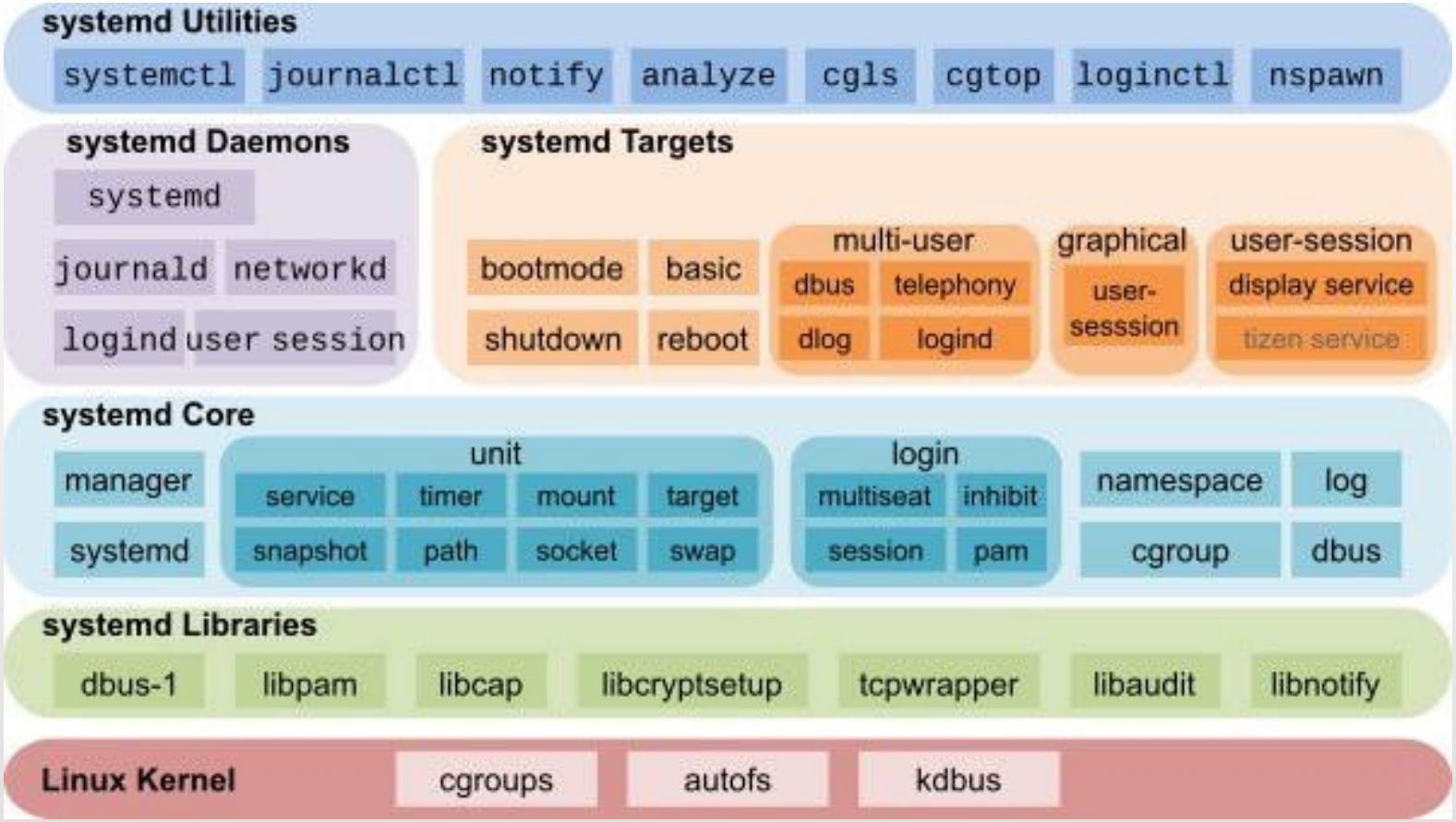
Proceso de arranque: ***SystemD***

- Es un sistema que centraliza la administración de demonios (servicios) y librerías del sistema.
- Mejora el paralelismo de arranque.
- Puede ser controlado con el comando **systemctl**.
- Compatible con SysV init → si es llamado como *init*.
- El demonio *systemd* reemplaza al proceso *init* y es el que tiene PID 1.
- Los runlevels son reemplazados por **targets**.
- No utiliza el archivo de configuración **/etc/inittab**.

Proceso de arranque: ***SystemD***

- Las unidades de trabajo son denominadas **units** y tienen distintos tipos, siendo los más relevantes:
  - **Service**: controla un servicio particular (*.service*).
  - **Socket**: encapsula *IPC*, un socket del sistema o file system *FIFO* (*.socket*) → *socket-based activation*.
  - **Target**: agrupa *units* o establece puntos de sincronización durante el arranque (*.target*) → dependencia de unidades.
  - **Snapshot**: almacena el estado de un conjunto de unidades para que pueda ser restablecido más tarde (*.snapshot*).
- Las *units* pueden tener dos estados → **active** o **inactive**.

Proceso de arranque: ***SystemD***



*Proceso de arranque: SystemD - **activación por socket***

- No todos los servicios que se inician en el booteo se utilizan:
  - Impresoras
  - Servidor en el puerto 80
  - etc.
- Es un mecanismo de inicio de servicios bajo demanda → podemos atender servicios sin que estén iniciados hasta el momento en que realmente se los necesita.
- Cuando el socket recibe una conexión, inicia el servicio y le pasa el socket para que reciba la petición.
- No hay necesidad de definir dependencias entre servicios → se inician todos los sockets en primer medida.

*Proceso de arranque: SystemD - **cgroups***

- Permite organizar un grupo de procesos en forma jerárquica.
- Agrupa conjuntos de procesos relacionados (por ejemplo, un servidor web NGINX con sus procesos dependientes)
- Tareas que realiza:
  - Seguimiento mediante subsistema *cgroups* → no se utiliza el PID individual → doble *fork* no funciona para escapar de SystemD.
  - Limita el uso de recursos (CPU, memoria, discos, etc).
  - etc.

Proceso de arranque: ***fstab***

- Define qué particiones se montan al arranque.
- Algunas opciones:
  - **user**: cualquier usuario puede montar la partición.
  - **auto**: monta la partición al inicio.
  - **ro**: read only, **rw**: read and write.
- Su configuración se encuentra en /etc/fstab:

```
$ cat /etc/fstab
# <file system> <mount point> <type> <options> <dump> <pass>
/dev/sda1 / ext4 errors=remount-ro 0    1

UUID=3FDE00F9523092AE /home/iso/datos ntfs user,auto
,rw,exec,uid=1000,gid=1000,umask=000 0 2

/dev/sda2 none swap sw 0 0
```

¿Preguntas?