1. Introducción

1.1 Tipos de Instalación

A la hora de instalar un sistema, tenemos que tener en cuenta el tipo de funciones que va a desempeñar.

- **Sistema de escritorio:** usado en tareas rutinarias
- **Estación de trabajo:** sistema de alto rendimiento, generalmente orientado a una tarea específica. Por ejemplo estaciones dedicadas al cálculo o estaciones gráficas.
- **Servidores:** ofrecen servicios a otras máquinas de la red
  - **Servicios de disco:** acceso a ficheros a través de FTP, servicio de disco transparente a través de NFS o Samba
  - **Servicios de aplicaciones**, por ejemplo, terminales, conexión remota (telnet, ssh) ...
  - **Servicios de directorio**, por ejemplo, LDAP, que almacena credenciales de usuarios, directorios, etc.
  - **Servicios de bd**
  - **Servicios de impresión**
  - **Servicios de red**

1.2 Instalación

Seguimos con detalle los pasos sugeridos por el instalador, al llegar a la parte del particionado del disco podemos instalar todo el sistema en una sola partición, aunque no es nada recomendable. Esto se debe a motivos de **seguridad, estabilidad, mantenimiento y rendimiento**. La estructura de directorios **UNIX** sigue el estándar **FHS** (*Fylesystem Hierarchy Standard*)

Fylesystem Hierarchy Standard

- **/ (root o raíz):** directorio raíz del sistema, todo cuelga de aquí
- **/boot/:** ficheros usados para el arranque, incluyendo el kernel
- **/bin/ (binaries):** ejecutables esenciales de configuración y administración para el superusuario.
- **/lib/:** librerías esenciales para los ejecutables de **/bin/** y **/sbin/**
- **/usr/ (Unix System Resources):** resto de las aplicaciones usadas por los usuarios y el superusuario. Incluye ejecutables, librerías, cabeceras para programación en

C, código fuente, manuales, etc, organizados en los siguientes subdirectorios.
    - **/usr/bin/:** la mayoría de las aplicaciones de usuario
    - **/usr/sbin/:** la mayoría de las aplicaciones para el superusuario
    - **/usr/lib/:** librerías que necesitan los ejecutables de **/usr/bin/** y **/usr/sbin/**
    - **/usr/share/:** datos independientes de la arquitectura, fundamentalmente manuales
    - **/usr/include/:** ficheros de cabecera (.h) estándar
    - **/usr/src/ (opcional):** código fuente del kernel y de las aplicaciones
    - **/usr/local/:** aplicaciones que no forman parte de la distribución y que el superusuario ha instalado manualmente. Replica la estructura anterior **usr/local/bin/, /usr/local/lib/, /usr/local/share/, etc.**
- **/opt/:** aplicaciones que requieren un subdirectorio separado del resto
- **/etc/:** ficheros y scripts de configuración, tanto del sistema como de las aplicaciones.
- **/var/:** ficheros variables (logs, bases de datos, etc.) Por ejemplo:
    - **/var/log/:** ficheros de log
    - **/var/spool/:** ficheros temporales de impresión, e-mail y otros
- **/tmp/:** ficheros temporales que se borran durante el reinicio del sistema
- **/srv/:** datos de servicios proporcionados por el sistema (páginas web, ftp, cvs, etc.)
- **/home/ (opcional):** directorio de usuarios (directorio inicial o home)
- **/root/ (opcional):** directorio *home* del superusuario

Otros directorios:

- **/media/:** punto de montaje para medios removibles (USB, CDROM)
- **/mnt/:** punto de montaje para otros sistemas temporales (por ejemplo una partición de otro SO)
- **/dev/:** directorio que contiene *seudoficheros* de acceso a periféricos, operando sobre estos ficheros damos ordenes a los dispositivos
- **/proc/:** directorio que contiene información del sistema (CPU, memoria, buses, interrupciones, procesos en ejecución, etc.)
- **/sys/:** similar al anterior, contiene información de dispositivos (por ejemplo, el brillo de la pantalla y la carga de la batería de los **portátiles**)

Esquemas de particionamiento

Dependiendo del tipo de sistema:

- **Máquina de escritorio** (un solo usuario trabajando a la vez).
    - **swap:** área de intercambio, es una zona del disco que en caso de emergencia puede utilizarse en sustitución de la memoria RAM. Se le suele otorgan al menos el doble del tamaño de la RAM.

    - **/home/:** cuentas de usuario, tamaño en función del número de usuarios
    - **/:** resto del disco (con el SO)
- **Sistema multiusuario**, además de las particiones anteriores crear particiones separadas para **/usr, /var y /tmp**.
    - **/usr** podría montarse en modo sólo-lectura después de que todo el sistema esté instalado
    - tener **/var** y **/tmp** en su partición evita que estos directorios crezcan hasta ocupar todo el disco
- **Particiones adicionales:**
    - **/boot:** para tener la función de arranque separada del resto. Permite incluso evitar las incompatibilidades de las BIOS con los discos duros grandes
    - **/chroot:** para aplicaciones en un entorno *enjaulado* que requieran seguridad y aislamiento
    - **/var/lib:** partición para gestionar del servidor de bases de datos (DNS, Apache) o del proxy (MySQL, Squid).

Sistemas de ficheros

Linux soporta varios sistemas de archivos:

- **ext4:** Fourth EXTended filesystem, es el sistema de ficheros estándar en Linux
    - Es un sistema de ficheros transaccional (como una bd)
    - Tiene características que le permiten reducir la fragmentación
    - Puede trabajar con discos y ficheros de gran tamaño.
    - Las opciones pueden configurarse con el comando `tuen2fs`
    - Las anteriores versiones `ext3` y `ext2` siguen estando disponibles
- **btrfs:** posible sucesor de ext4, pues presenta características avanzadas que además de mejorar el rendimiento, van dirigidas a la gestión y seguridad del almacenamiento:
    - Gestiona de manera integrada el almacenamiento, pues incluye funciones que antes formaban parte del sistema de ficheros, del controlador RAID y del gestor lógico de volúmenes LVM
    - Hace uso extensivo de *copy-on-write*, si varios recursos son idénticos se devuelve un puntero a un único recurso; en el momento en que se modifica una "copia" del recurso, se crea una copia auténtica para prevenir que los cambios sean visibles a las demás copias.
    - Permite *snapshots* de solo lectura o modificables
    - Etc
- **JFS, XFS:** otros tipos de sistemas transaccionales portados de otros sistemas Unix
- **NFTS, exFAT:** usados por Windows en ordenadores domésticos soportados también por linux

- **ReFS:** usado por windows para empresas, no disponible de forma nativa para windows

Instalación del gestor de arranque

Podemos tener diferentes distribuciones de Linux y Windows en el mismo ordenador, cada una con sus correspondientes particiones. El gestor de arranque (cargador o *bootloader*) nos permite seleccionar el SO a arrancar.

- Las distribuciones **Linux** usando el cargador **GRUB** (GRand Unified Bootloader)
- Cuando el sistema se inicia, la **BIOS** carga el gestor de arranque que nos permite seleccionar el SO y a continuación transfiere el control al programa de inicio del correspondiente SO (localizado en /boot)

Tenemos dos posibilidades a la hora de instalarlo:

- Instalarlo en el **MBR** o **GTP** del primer disco:
  - El MBR contiene información osbre las particiones del disco y un pequeño código del gestor de arranque. MBR se almacena solo en el primer sector del disco
  - El GTP es el nuevo estándar que sustituye al anterior, más fiable y asociado a los sistemas UEFI. GTP Crea múltiples copias redundantes a lo largo de todo el disco y su nombre hace referencia a que a cada partición se le asocia un identificador global único
- En caso de que tengamos otro cargador en el MBR o GTP, el gestor de arranque GRUB puede instalarse en el primer sector de la partición Linux que contenga /boot

Podemos configurar GRUB para evitar que sea modificado el menu de arranque. En concreto, podemos usar una contraseña para limitar:

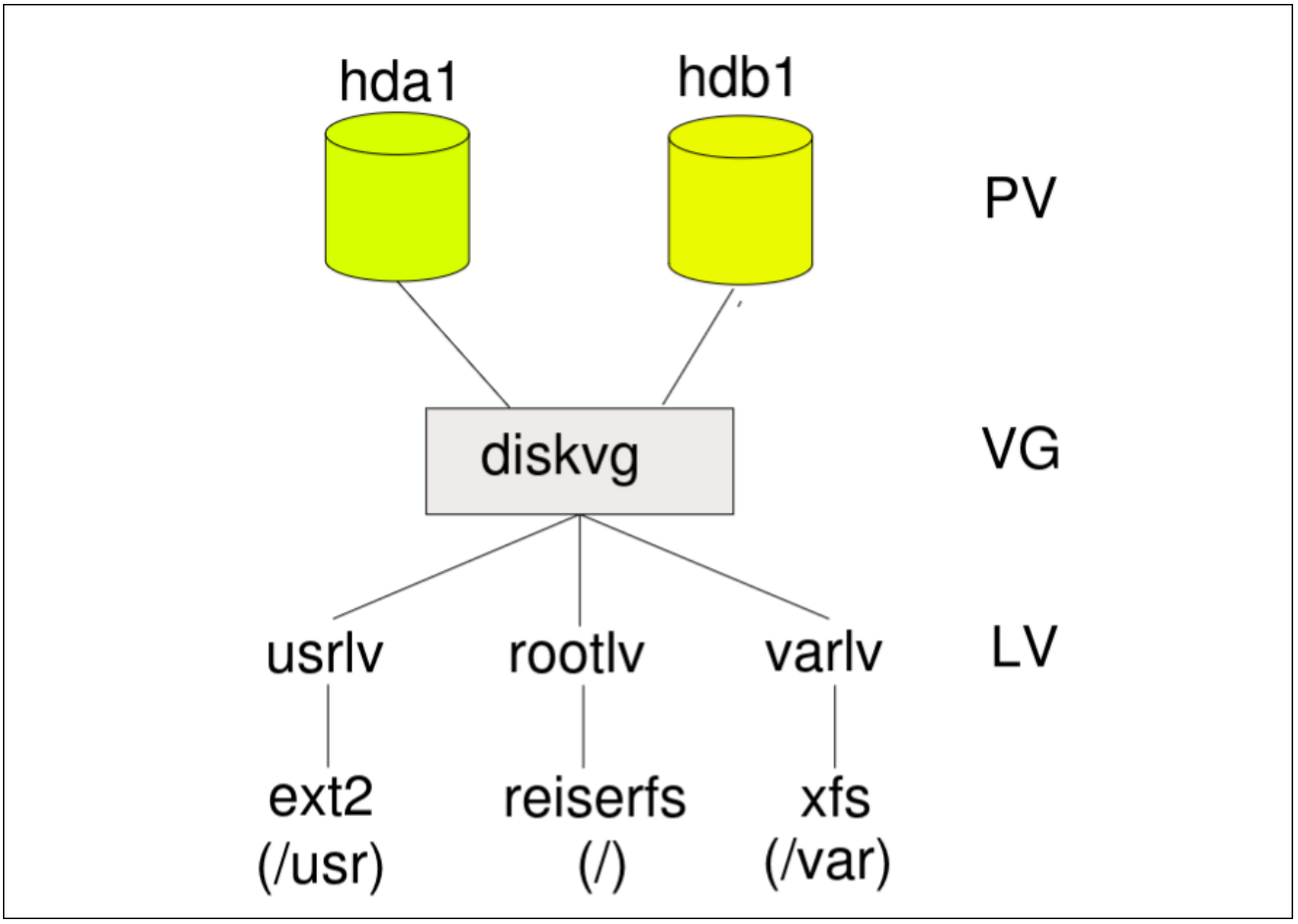
- la modificación de los parámetros iniciales
- el acceso a determiandas imágenes
- el acceso a opciones avanzadas

Logical Volume Management

Proporcionar una visión de alto nivel de los discos:

- permite ver varios discos como un único volumen lógico
- permite hacer cambios en las particiones sin necesidad de reiniciar el sistema
- permite gestionar los volúmenes en grupos definidos por el administrados
- **Volumen físico (PV):** discos duros, particiones de los discos y otros dispositivos
- **Grupos de volúmenes (VG):** agrupación de LV, que forman una unidad administrativa

- **Volumen lógico (LV):** particiones lógicas sobre las que se montan los sistemas de ficheros



1.3 Arranque del sistema

1. **BIOS:** el procesador se dirige a una posición de memoria específica, donde se encuentra la **BIOS**. Se ejecuta un programa que detecta los discos, carga el registro del arranque y ejecuta el gestor de arranque.
2. **Gestor de Arranque:** es un pequeño programa que usualmente muestra un menú con la lista de los SO disponibles, el usuario puede seleccionar lo que le interese y cargar el kernel.
3. **Kernel:** se localiza en **/boot/vmlinuz** y empiza a buscar y montar la partición que contiene el sistema de ficheros raiz, ejecutando el primer programa, **init**. Esto se suele hacer en 2 pasos:
    1. **initframs:** primera fase donde se carga en RAM el fichero de imagen de un pequeño disco virtual que contiene un partición raíz virtual y el programa **init**
    2. **init:** initframs cede el control al init real y la máquina inicia el proceso de arranque estándar.

Systemd es el sistema de inicio actualmente utilizando en Debian:

- ejecuta varios procesos encargados de la creación del sistema: teclado, controladores, sistemas de ficheros, red, servicios...
- esto hace que ofrezca una visión global del sistema tanto software como hardware

- muchos de estos procesos ofrecen servicion

Varias utilidades:

- **systemctl** es un comando de gestión de servicios específico de **systemd**. Si se ejecuta sin argumentos muestra la lista de servicios activos, mientras que si indicamos un servicio podemos realizar diferentes operaciones sobre el.

```
systemctl
systemctl acción servicio

Donde la acción puede ser:

status - muestra el estado del servicio
start - inicia el servicio
stop - detiene el servicio
restart - detiene el servicio y a continuación lo reinicia
enable - configura el servicio para ser iniciado en el arranque
disable - el servicio no será iniciado durante el arranque
```

```
# configura un arranque sin entorno gráfico
systemctl disable xdm
# comprueba si hay servidor de ssh
systemctl status ssh
# reinicia la red
systemctl restart networking
```

- **service:** es el comando que clásico que tiene una sintaxis similiar

```
service --status-all
service servicio acción
```

```
# comprueba si hay servidor de ssh
service ssh status
# reinicia la red
service networking restart
```

Systemd distingue varios target. Entre ellos destacan:

- **Rescate:** que arranca lo mínimo para intentar repara un sistema dañado
- **Emergencia:** abre un único shell
- **Multisuario:** multiusuario no gráfico
- **Gráfico:** multiusuario gráfico

```
# Muestra el target actual
systemctl get-default
# Conmuta al target especificado
systemctl isolate <name of target>.target
# Fija el target que se ejecutará en el arranque
systemctl set-default <name of target>.target
```

1.4 Verificación de la instalación

`dmidecode`	vuelca la información DMI de la BIOS
`lshw`	infomación general de hardware
`lscpu`	información sobre el procesador
`nproc`	devuelve el número de núcleos disponible
`lspci`	muestra los dispositivos PCI
`lsusb`	muestra los dispositivos USB
`fdisk -l`	muestra todas las particiones
`df -h`	muestra las particiones montadas
`free -h`	memoria total, usada, disponible
`uname -a`	muestra versión de kernel
`lsb_release -a`	muestra distribución de Linux
`lsmod, rmmod, insmod`	operaciones con módulos del kernel
`/proc, /sys`	directorios con información de hardware
`/dev/sdxx, /dev/nvmexx`	dispositivos de discos y particiones

2. Instalación de software

2.1 Formas de instalación

Hay dos formas de instalar programas en linux:

- Instalación desde paquetes precompilados: menos optimización, más sencilla
- Compilación e instalación desde las fuentes: optimización para nuestro sistema, más compleja

Gestores de paquetes

En la mayoría de distribuciones Linux, es posible obtener los programas precompilados en formato de paquetes

- **Ventajas:** fáciles de instalar y desinstalar, fáciles de actualizar, fácil control de los programas instalados
- **Inconvenientes:** Binarios menos optimizados, problemas de dependencias de paquetes, problemas si la base de datos de paquetes se corrompe.

Formatos de paquete más populares:

- **DEB**
- **RPM**

Gestión de paquetes en Debian

Existen varias herramientas para instalar paquetes:

- **dkpg:** herramienta de bajo nivel, para gestionar directamente los paquetes DEB

-i	--install	instalación del paquete
-r	--remove	eliminación del paquete
-P	--purge	eliminación completa del paquete
-l	--list	lista los paquetes
-s	--status	imprime el estado del paquete
-L	--listfiles	lista el contenido del paquete
-S	--search	busca el paquete al que pertenece un fichero
dpkg-reconfigure		reconfigura el paquete
/var/lib/dpkg/lock		fichero de bloqueo (cerrojo)

- **apt:** herramientas **APT**, permiten gestionar los paquetes descargándolos de varias fuentes (CDs, http). Apt cuenta con un fichero de configuración que contiene los distintos servidores desde los cuales se obtienen los paquetes (**/etc/apt/sources.list**)

- `/etc/apt/sources.list`

Ejemplo:

```
# See sources.list(5) for more information
deb ftp://ftp.rediris.es/debian/ stable main contrib non-free
deb http://security.debian.org/ stable/updates main contrib non-free
# Para descargar fuentes, a través de apt-get source
deb-src ftp://ftp.rediris.es/debian/ stable main
```

- formato de `sources.list`

  # Para binarios
  `deb` *uri distribución componentes*
  # Para ficheros fuente
  `deb-src` *uri distribución componentes*

- *distribución* puede ser:
  - `stable` (o el nombre que tenga la versión estable, actualmente, *trixie*), es la versión recomendada.
  - `testing` contiene versiones más recientes de paquetes, aunque todavía no probados exhaustivamente.
  - `unstable` (también denominada *sid*), es la versión en desarrollo, probablemente con errores sin corregir.
- *componentes* pueden ser (se admiten varios en la misma línea):
  - `main` - conjunto principal de paquetes
  - `contrib` - paquetes adicionales
  - `non-free` - paquetes que no son libres

`apt-get`	`update`	actualiza la lista de paquetes
	`upgrade`, `dist-upgrade`	actualiza los paquetes
	`install`	instala el paquete
	`remove`, `purge`	desinstala el paquete
	`autoremove`, `clean`	limpia paquetes innecesarios
	`source`	descarga un fichero fuente
	`build-dep`	descarga dependencias de compilación
`apt-cache`	`search`	busca paquetes para instalar
	`show`	muestra información del paquete
	`depends`	lista las dependencias del paquete
	`policy`	muestra fuentes y prioridades
`apt-get -f install`		corrige los errores de dependencias
`/var/lib/dpkg/lock`		fichero de bloqueo (cerrojo)

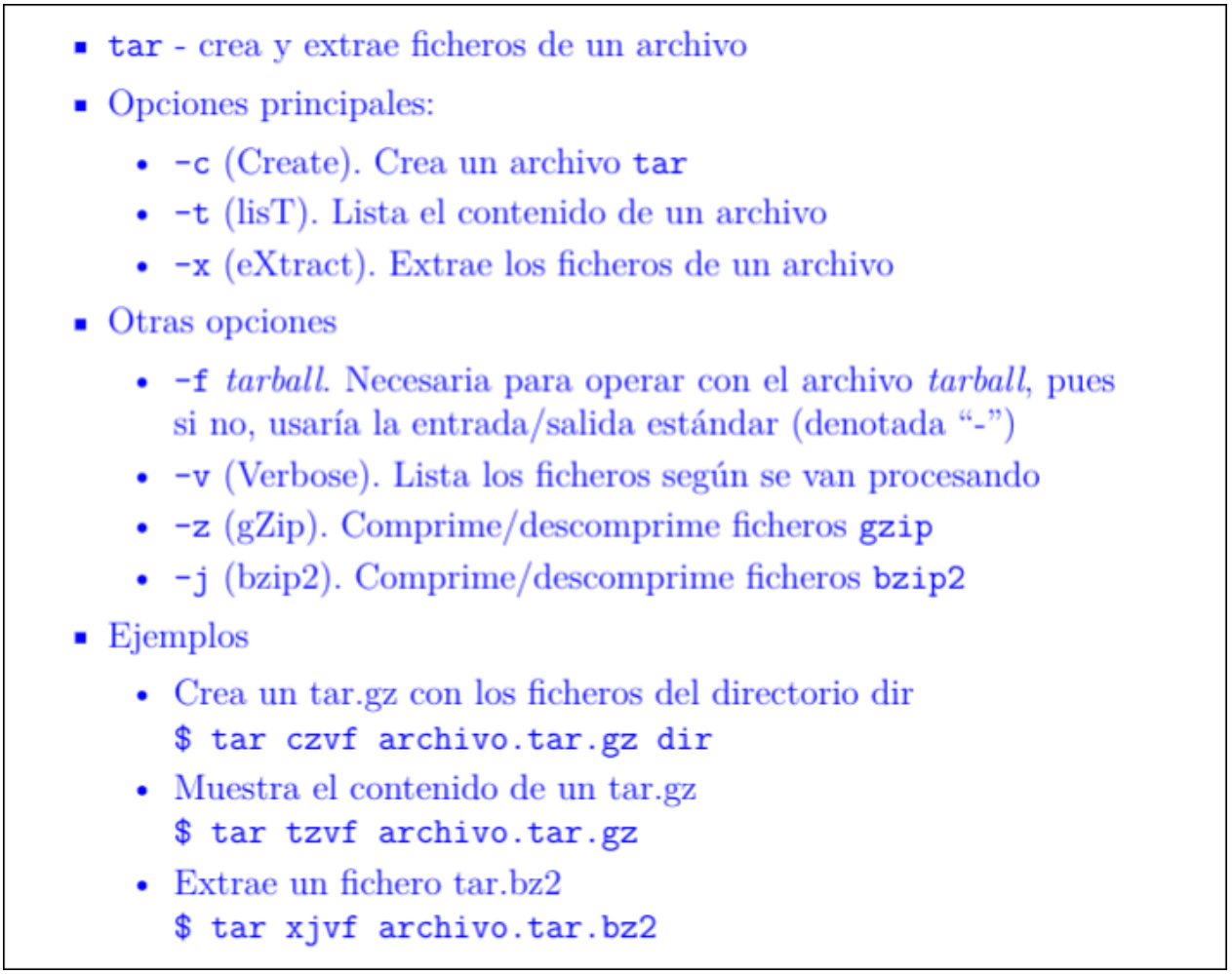
- Muchas otras herramientas con interfaz gráfico o semigráfico, a veces con formato *store*
- **alien:** permite convertir e instalar en Debian paquetes de otro tipo como RPM

- **snap:** estos paquetes son autocontenidos por lo que funcionan en una amplia gama de distribuciones linux. Incluyen dentro del paquete todas las componentes que necesitas como las librerías.
- **pip:** es el instalador de paquetes de Python

2.2 Instalación desde el código fuente

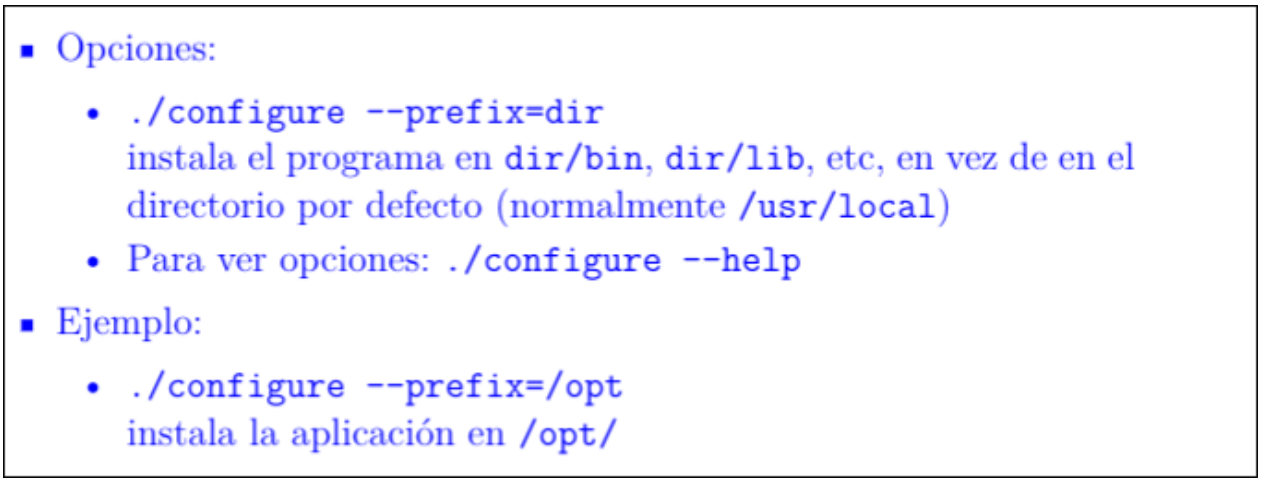
Los pasos para instalar manualmente una aplicación desde código fuente:

1. Descarga, normalmente se distribuyen en forma de *tarballs*:
   - .tar (empaquetado pero sin comprimir)
   - .tar.gz (empaquetado y comprimido, abreviado .tgz)
2. Desempaqeutado: comando tar (Tape ARchive format)



- `tar` - crea y extrae ficheros de un archivo
- Opciones principales:
  - `-c` (Create). Crea un archivo `tar`
  - `-t` (lisT). Lista el contenido de un archivo
  - `-x` (eXtract). Extrae los ficheros de un archivo
- Otras opciones
  - `-f` *tarball*. Necesaria para operar con el archivo *tarball*, pues si no, usaría la entrada/salida estándar (denotada "-")
  - `-v` (Verbose). Lista los ficheros según se van procesando
  - `-z` (gZip). Comprime/descomprime ficheros `gzip`
  - `-j` (bzip2). Comprime/descomprime ficheros `bzip2`
- Ejemplos
  - Crea un tar.gz con los ficheros del directorio dir
    `$ tar czvf archivo.tar.gz dir`
  - Muestra el contenido de un tar.gz
    `$ tar tzvf archivo.tar.gz`
  - Extrae un fichero tar.bz2
    `$ tar xjvf archivo.tar.bz2`

3. Leer el fichero INSTALL
4. Configuración, mediante el script de configure `./configure <opciones>`


- Opciones:
  - `./configure --prefix=dir`
    instala el programa en `dir/bin`, `dir/lib`, etc, en vez de en el directorio por defecto (normalmente `/usr/local`)
  - Para ver opciones: `./configure --help`
- Ejemplo:
  - `./configure --prefix=/opt`
    instala la aplicación en `/opt/`

5. Compilación `make all`
6. Instalación `make install`

Tipos de ejecutables

- **Enlazados estáticamente:** son completos
- **Enlazados dinámicamente:** para ejecutarse necesitan librerías instaladas en el sistema
  - Ocupan menos que los estáticos
  - Librerías compartidas por varios programas

Enlazamiento dinámico

El comando ldd nos permite ver las librerías que un ejecutable necesita. El formato de la salida es: `librería requerida => librería encontrada`. Por ejemplo:

```
# ldd /bin/ls
     libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6
     libpcre.so.3 => /lib/x86_64-linux-gnu/libpcre.so.3
     ...
```

Si no se encuentra una librería:

```
# ldd /bin/ls
     libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6
     libpcre.so.3 => Not found
     ...
```

En el caso de no encontrar una librería, puede deberse a dos problemas:

- La librería se encuentra en una localización no estándar. Esto puede resolverse indicando la ubicación de la librería.
- La librería que necesita el programa es de una versión diferente a la que tenemos en el ordenador. El problema puede intentar arreglarse haciendo un enlace

simbólico (comando ln) a la librería existente con el nombre de la librería requerida. Si las versiones de las librerías son muy diferentes, esta solución probablemente no funcionará y será necesario instalar la nueva versión de la librería (pueden convivir sin problemas ambas versiones de librería en el mismo ordenador). Por ejemplo:

```
ln -s libgdal.so.30.0.1 libgdal.so.30.0.2
```

El cargador dinámico (ld - dynamic linker) se encarga de enlazar y cargar las librerías que necesitan los ejecutables.

2.3 Automatización de tareas

Vamos a ver comandos que permiten automatizar tareas repetitivas

- Tareas que se deben ejecutar en momentos concretos o de forma periódica:
  - `at, batch` permiten ejecutar trabajos a una hora específica o bajo determinadas condiciones
  - `cron` permite correr trabajos a intervalos regulares
- Herramientas para automatizar la configuración de servidores

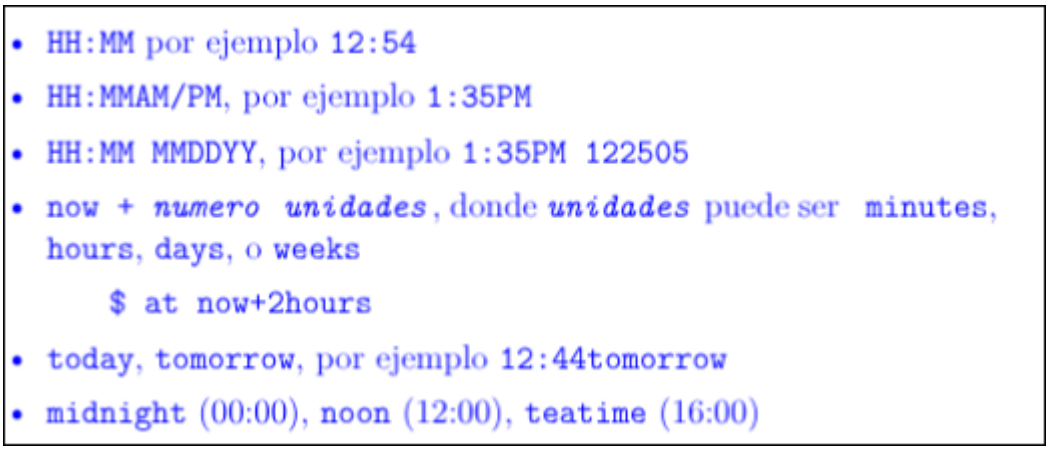
2.3.1 Tareas periódicas

`at` permite indicar el momento en que se quiere ejecutar un trabajo

- Sintaxis: `at [opciones] TIME
- Al ejecutar `at` pasamos a un nuevo prompt, que nos permite introducir comandos que se ejecutarán a la hora indicada.
  - para guardar el trabajo y salid CTRL-D
  - al terminar, la salida estándar se envía como un mail al usuario
  - el trabajo se ejecuta mientras el sistema esté encendido a la hora

```
$ at 11:45
warning: commands will be executed using /bin/sh
at> ls /tmp > lista
at> env DISPLAY=:0 zenity -info -text="Hola"
at> <EOT>
job 4 at Wed Nov 16 11:45:00 2005
```

- `TIME` se puede expresar como



- HH:MM por ejemplo 12:54
- HH:MMAM/PM, por ejemplo 1:35PM
- HH:MM MMDDYY, por ejemplo 1:35PM 122505
- now + *numero unidades*, donde *unidades* puede ser minutes, hours, days, o weeks

```
$ at now+2hours
```

- today, tomorrow, por ejemplo 12:44tomorrow
- midnight (00:00), noon (12:00), teatime (16:00)

- Comandos relacionados:
  - `atq` lista los trabajos pendientes del usuario
  - `atrm` borra trabajos identificados por su número de trabajo
  - `batch` ejecuta trabajos cuando la carga del sistema es baja

Para crear trabajos que se ejecuten periódicamente se utilizan el demonio `cron` y el comando `crontab`

- Sintaxis `crontab [-u usuario] {-e|-l\-r}`
  - `-e` edita o crea nuevos trabajos
  - `-l` muestra los trabajos
  - `-r` borra los trabajos
  - `-u usuario` para operar como otro usuario
- Sintaxis usando un fichero previamente escrito `crontab [-u usuario] fichero`

Los trabajos se especifican en un fichero que puede tener tres tipos de líneas:

- Comentarios que empiezan por `#`
- Definición de variables de tipo `nombre=valor`

```
# shell usada para ejecutar los comandos
SHELL=/bin/bash
# Usuario al que se envía (por mail) la salida del comando
# (por defecto, se envían al propietario del fichero)
MAILTO=pepe
```

- Especificación del trabajo y hora de ejecución:
  `minuto hora día mes día_semana comando`
- El días de la semana de 0 a 7
- - indica cualquier valor
- se pueden indicar rangos, listas o repeticiones
  - 1-5 para indicar de lunes a viernes
  - 0, 15, 30, 45 para indicar cada 15 minutos
  - 0-23/2 en el campo hora indicar realizar cada dos horas en todo el rango
    Además, el administrador puede crear scripts que se ejecuten con periodicidad horaria, diaria, semanal y mensual.

```
/etc/cron.hourly
/etc/cron.daily
/etc/cron.weekly
/etc/cron.monthly
```

`cron` está pensado para sistemas funcionando 24/7. Si el sistema está apagado a la hora de una acción cron, esa iteración no se realiza. Solución complementaria: `Anacron` ejecuta asíncronamente tareas periódicas programadas.

2.4 Copias de seguridad

Las copias de seguridad son vitales para recuperar información en caso de pérdida por errores, ataques, fallos hardware, desastres, etc.

Componentes clave:

- **Medios de almacenamiento:**
  Cintas, discos duros externos, discos ópticos, backup en la nube.
- **Programa de copia:** se encarga de copiar los ficheros seleccionados en el medio de almacenamiento.
  - *Basado en imagen* accede al disco a bajo nivel(ej. `dump`, `dd`): copia sectores del disco, restauración completa pero menos flexible.
  - *Fichero a fichero* (ej. `tar`): copia archivos individualmente, más flexible para restaurar ficheros sueltos.
- **Planificador:**
  Decide qué y cuándo se copia. Se suele usar `cron` para programar las tareas.

Tipos de backup:

- **Completo (nivel 0):** Se copia toda la información.
- **Diferencial:** Copia archivos modificados desde el último backup completo.
- **Incremental:** Copia archivos modificados desde el último backup (completo o incremental).

Ejemplo de planificación:

- Backup mensual completo (nivel 0), semanal diferencial (nivel 5), diario diferencial (nivel 9).
- Otra opción: mensual completo (nivel 0), semanal diferencial (nivel 2), diarios incrementales (niveles 3-6).
- Esquemas de niveles sirven para organizar los backups y saber qué cintas/dispositivos usar para restaurar.

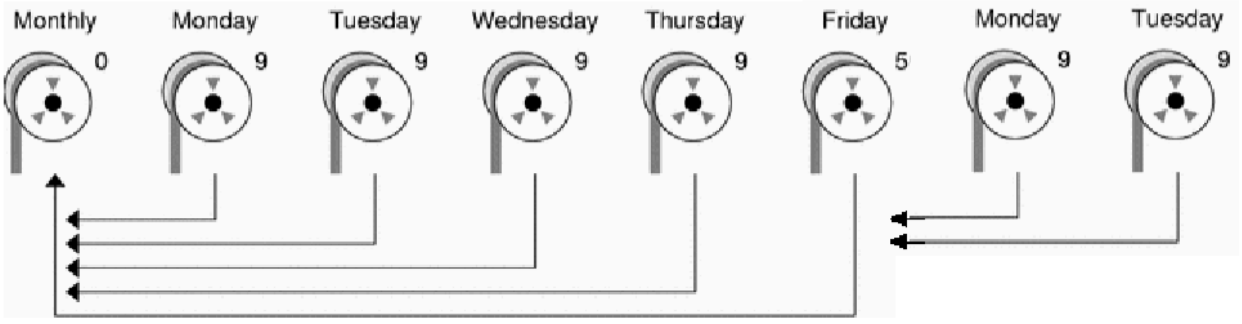
Restauración:
Para restaurar, necesitas las últimas copias de cada nivel relevante (completo,

semanal, diario).

Comandos básicos para copias de seguridad

- **dump**:
  Hace copias de sistemas de archivos completos (no de directorios sueltos), preservando permisos y permitiendo backups incrementales/multivolumen.
  Ejemplo:
  ```shell
  dump -0 -u -f /dev/st0 /home
  ```
- **restore**:
  Restaura ficheros salvados por dump. Permite restaurar completos, individuales, en modo interactivo, y desde sistemas remotos.
- **tar**:
  Empaqueta varios archivos/directorios en uno solo, útil para backup de ficheros individuales.
- **dd**:
  Copia y convierte datos a bajo nivel (imagen de disco, partición, etc.).
  Ejemplo:
  ```shell
  dd if=/dev/sda3 of=/tmp/imagen.img
  ```
- **mt**:
  Manipula directamente unidades de cinta (rebobinar, borrar, avanzar, etc.).

Ejemplo 1: backup mensual de nivel 0, semanal de nivel 5 y diario de nivel 9, siendo los dos últimos diferenciales (los diarios diferenciales con respecto al último semanal y los semanales diferenciales con respecto al mensual).



Aplicaciones y herramientas adicionales

- **Bacula**: Backup en red, modular, soporta varios sistemas y bases de datos.
- **Amanda**: Backup de red, usa dump y tar, soporta medios de backup variados.
- **Flexbackup**: Backup flexible, fácil de usar en sitios pequeños/medianos, soporta varios formatos.
- **rdiff-backup**: Copia y sincroniza directorios (locales o remotos), guarda diferencias, permite recuperar versiones antiguas.

- **DAR (Disk ARchiver)**: Backups de árboles de directorios, soporta multivolumen (ideal para CDs/DVDs).
- **BackupPC**: Backup de alto rendimiento a servidor/NAS, no requiere software en clientes, interfaz web.
- **UrBackup**: Sistema cliente-servidor para backups completos o incrementales, configurable, interfaz web.