Resumen PAS 01/04/2018

[1. Introducción a la administración de sistemas. 3](#_Toc511723579)

[1.1. El sistema informático y la organización. 3](#_Toc511723580)

[1.1.1. El sistema Informático. 3](#_Toc511723581)

[1.1.2. El Departamento de Informática. 3](#_Toc511723582)

[1.2. Administrador de Sistemas. 3](#_Toc511723583)

[1.2.1. Tareas más detalladas. 3](#_Toc511723584)

[1.2.2. Software libre. 4](#_Toc511723585)

[1.3. Superusuario o administrador. 4](#_Toc511723586)

[2. Organización de un sistema operativo tipo GNU/Linux. 5](#_Toc511723587)

[2.1. Ficheros. 5](#_Toc511723588)

[2.1.1. Sistema de ficheros. 5](#_Toc511723589)

[2.2. Procesos. 6](#_Toc511723590)

[2.3. Dispositivos. 6](#_Toc511723591)

[2.4. Estructura genérica del sistema de ficheros. 7](#_Toc511723592)

[3. Arranque y parada del sistema. 8](#_Toc511723593)

[3.1. Proceso de arranque. 8](#_Toc511723594)

[3.1.1. Proceso de arranque completo. 8](#_Toc511723595)

[3.1.2. Gestor de Arranque GRUB 8](#_Toc511723596)

[3.1.3. Modo monousuario. 8](#_Toc511723597)

[3.1.4. Niveles de ejecución de GNU/Linux. 9](#_Toc511723598)

[3.1.5. Ficheros de inicialización. 9](#_Toc511723599)

[3.1.6. Upstart. 9](#_Toc511723600)

[3.2. Parada del Sistema. 10](#_Toc511723601)

[3.3. Caídas del sistema y problemas de arranque. 10](#_Toc511723602)

[4. Gestión de Usuarios. 11](#_Toc511723603)

[4.1. Introducción. 11](#_Toc511723604)

[4.2. Usuarios. 11](#_Toc511723605)

[4.2.1. Fichero /etc/passwd, contraseñas y shadow passwords. 11](#_Toc511723606)

[4.2.1.2. Contraseñas. 11](#_Toc511723607)

[4.2.1.3. Shadow passwords. 12](#_Toc511723608)

[4.2.2. Restricciones de tiempo. 12](#_Toc511723609)

[4.2.3. Fichero de inicialización. 13](#_Toc511723610)

[4.2.4. Selección de interprete de órdenes. 13](#_Toc511723611)

[4.2.5. Cuentas restrictivas. 13](#_Toc511723612)

[4.2.6. Herramientas para crear/modificar cuentas de usuario. 13](#_Toc511723613)

[4.3. Grupos. 14](#_Toc511723614)

[4.3.1. Comandos de los Grupos. 14](#_Toc511723615)

[4.4. Usuarios y grupo estándar. 14](#_Toc511723616)

[4.4.1. Rangos del UID. 14](#_Toc511723617)

[4.4.2. Algunos usuarios y grupos estándar. 15](#_Toc511723618)

[5. Gestión de los recursos del sistema. 16](#_Toc511723619)

[5.1. Actividad de la CPU. 16](#_Toc511723620)

[5.1.1. Procesos. 16](#_Toc511723621)

[5.1.2. Prioridad y señales. 17](#_Toc511723622)

[5.1.3. Monitorizar el uso de la CPU. 18](#_Toc511723623)

[5.1.4. Programar ejecución de procesos. 19](#_Toc511723624)

[5.1.5. Rastreo de procesos. 19](#_Toc511723625)

[6. Ejercicios de cada tema. 20](#_Toc511723626)

[6.1. Tema 1. 20](#_Toc511723627)

[6.2. Tema 2. 20](#_Toc511723628)

[6.3. Tema 3. 23](#_Toc511723629)

[6.4. Tema 4. 26](#_Toc511723630)

# Introducción a la administración de sistemas.

## El sistema informático y la organización.

### El sistema Informático.

El sistema informático de una organización puede verse con ópticas para la organización (un departamento como cualquier otro), para los informáticos (conjunto de servidores, redes y PCs), para los usuarios (una herramienta más que organiza mejorando las tareas) y para la dirección (gran base de datos para hacer consultas que ayudan en la toma de decisiones).

### El Departamento de Informática.

El Departamento de Informática se encarga de mantener y gestionar el Sistema Informático. Existen varias partes dentro de este departamento:

* Sistema Informático Físico:
  + Hardware (servidores, PCs y cableado y electrónica de red)
  + Software (Sistemas Operativos, software empresarial de base y aplicaciones específicas.
* Personal: Responsable de Informática y Personal Técnico.

Las funciones del Departamento de Informática son la administración de servidores, de usuarios, de red, de datos, de la web, de la seguridad, el desarrollo y responsable de informática.

## Administrador de Sistemas.

Un Administrador de Sistemas es la persona que tiene la responsabilidad de implementar, configurar, mantener, monitorizar, documentar y asegurar el correcto funcionamiento de un sistema informático, o algún aspecto de este. También tiene por objeto garantizar el tiempo de actividad (uptime), rendimiento, uso de recursos y la seguridad de los servidores que administra.

La dedicación de un administrador de sistemas está pensada de tal manera para que solo se encargue de la administración, aunque casi siempre acaba compartiendo la labor de administración con otro tipo de trabajo. La labor del administrador de sistemas está pensada de tal manera de que sea uno de los miembros de su departamento, aunque casi siempre acaba haciendo todo el trabajo del departamento solo como un negro.

### Tareas más detalladas.

* Nivel más hardware: planificar y administrar el entorno físico, planificar costes de suministro, configurar y mantener la conectividad entre los hosts, etc.
* Mantenimiento: Mantenimiento software y documentación.
* Soporte a usuario: Formar a los usuarios (enseñarles), proporcionar soporte, etc.
* Servicios: Instalar y mantener las cuentas de usuario, determinar parches a instalar e instalar, configurar y administrar servidores web.
* Seguridad: Determinar cuotas de disco, políticas de manejo del espacio y monitorización de los ficheros de log. También configurar y manejar la seguridad del sistema.
* Copias de seguridad: Configurar y mantener backups del sistema.

### Software libre.

El software libre es aquel que concede cuatro libertades a sus usuarios:

* Libertad de usar el programa, con cualquier propósito.
* Libertad de estudiar cómo funciona el programa, y adaptarlo a sus necesidades.
* Libertad de distribuir copias.
* Libertad de mejorar el programa y hacer públicas las mejoras a los demás.

Por otro lado, no son software libre ni el software de dominio público, ni el software semilibre, ni el freeware, ni el shareware ni el software con fuentes.

## Superusuario o administrador.

El administrador o Superusuario es el usuario que tiene siempre todos los privilegios sobre cualquier fichero, instrucción u orden del sistema. En GNU/Linux ese usuario es root, que pertenece al grupo root. Mediante el comando sudo, el sistema permite a otros usuarios ejecutar ordenes como si fuesen el administrador. El archivo donde se encuentra configurado quien puede ejecutar que y como es /etc/sudoers.

# Organización de un sistema operativo tipo GNU/Linux.

## Ficheros.

En GNU/Linux, todos son ficheros, y si algo no es un fichero es un proceso. Con esto, damos por hecho que todos los programas, ordenes, dispositivos I/O, comunicación entre procesos, directorios y kernel son ficheros. GNU/Linux tiene una estructura jerárquica de directorios, conocida como sistema de archivos.

### Sistema de ficheros.

El sistema de ficheros guarda los ficheros del sistema, se organiza de manera jerárquica (en directorios) y no hay unidades.

#### Nodos-i.

Los ficheros se encuentran repartidos a lo largo del disco duro, pudiendo tener cada fichero sectores a lo largo de toda la superficie. Los nodos-i son metadatos sobre los ficheros que nos indican información sobre sus aspectos. Cada fichero tiene un nodo-i. Todos están localizados en un área del disco duro, la cual está limitada (num max de nodos-i).

#### Propietarios y permisos.

El acceso a los ficheros se gestiona de la siguiente forma:

* Propietarios:

Cada fichero tiene dos propietarios, el usuario y el grupo. Con *chown* cambiamos el usuario propietario y *chgrp* cambia el grupo propietario.

* Accesos:
  + R: (fichero) ver el contenido/ (directorio) listar el contenido
  + W: (fichero) modificar el contenido/ (directorio) crear o eliminar ficheros.
  + X: (fichero) ejecutar el fichero/ (directorio) entrar en el directorio

El acceso se establece de forma independiente para el usuario propietario (u), usuarios del grupo propietario (g) y resto de usuarios (o).

Otros permisos especiales destacables son la t (sticky bit) y la s (para usuarios suid y para grupos sgid) los cuales se encuentran fenomenalmente explicados en internet.

* Mascara de permisos:

Cuando un fichero nuevo se crea, se le asignan permisos mediante la aplicación de una máscara de permisos a los permisos base. La máscara de bits indica con 1 aquellos bits que deberán ser 0 en la cadena de permisos, es decir, indica que permisos están restringidos. Nota: Permisos base para directorios 777 y para ficheros 666.

* Tipos de ficheros
  + Normal
  + Directorio (d): son ficheros que contienen enlaces a otros ficheros.
  + Especial de bloque (b): fichero especial para interactuar con un dispositivo basado en bloques.
  + Especial de carácter (c): fichero especial para interactuar con un dispositivo basado en caracteres.
  + Named Pipes (p): tubería FIFO con nombre (comunicación de procesos de diferentes usuarios con tuberías).
  + Socket (s): como los pipes, pero con comunicación dúplex.
  + Enlaces:

Son archivos especiales que permiten que varios nombres (enlaces) se asocien a un único e idéntico archivo. Al ser varias instancias de un mismo archivo en diversos lugares de la estructura jerárquica sin tener que copiarlos, ayuda a asegurar la coherencia y ahorrar espacio en el disco. Existen dos tipos:

* + - Enlaces físicos: Representa un nombre alternativo para un archivo. Si eliminamos dicho enlace físico, no eliminamos el archivo original. Solo se puede hacer entre ficheros en la misma partición. No se puede realizar a directorios.
    - Enlace simbólico (l): Es un puntero virtual al archivo real. Es un fichero de texto (con su nodo-i independiente) que contiene la ruta del archivo al que apunta. Si se elimina el enlace simbólico, no se elimina el fichero original.

## Procesos.

Los procesos son programas en ejecución. Los atributos de un proceso son:

* PID (identificador del proceso)
* PPID (identificador del proceso padre)
* Nice number (prioridad asignada al ejecutarlo)
* TTY (terminal en el que se está ejecutando)
* RUID (identificador del usuario real, el que lo ejecuto)
* EUID (identificador del usuario efectivo (permido *suid*))
* RGID (identificador del grupo real, el grupo del usuario que lo ejecuto)
* EGID (identificador del grupo efectivo (permiso *sgid*)

Existen tres tipos de ficheros:

* Interactivos: hay alguien conectado al sistema que los inicia (en primer o segundo plano).
* Encolados: procesos que se mandan a un buffer para ser ejecutados.
* Demonios: programas ejecutados en segundo plano durante el arranque, que esperan de forma continua un determinado evento.

## Dispositivos.

Los dispositivos se representan/manejan como ficheros, ya sean, ficheros especiales de caracteres que representan a dispositivos de caracteres (como la cinta magnética, puerto paralelo o serie) o ficheros especiales de bloques que representan a dispositivos de bloques (disquete, partición de un disco duro o un pendrive). Al escribir/leer en un dispositivo se convierte en escribir/leer en el fichero correspondiente.

## Estructura genérica del sistema de ficheros.

Existen dos tipos de distinciones cuando hablamos del tipo de contenido de un directorio, ya sean, estáticos/dinámicos y compartibles/no compartibles.

* Estáticos: Contiene binarios, bibliotecas, documentación y otros ficheros que no cambian sin intervención del administrador. Pueden estar en dispositivos de solo lectura y no necesitan que se hagan copias de seguridad tan a menudo como los ficheros dinámicos.
* Dinámicos: Contiene ficheros que no son estáticos (wow). Deben de encontrarse en dispositivos de lectura-escritura. Necesitan copias de seguridad a menudo.
* Compartibles: Contiene ficheros que se pueden encontrar en un ordenador y utilizarse en otro.
* No compartibles: Contiene ficheros que no podemos utilizar en distintas maquinas.

Una vez aclarados estos conceptos, se muestra la estructura genérica del sistema de archivos:

* /bin: ficheros ejecutables básicos compartidos **(estático)**
* /dev: ficheros especiales de dispositivos.
* /etc: la mayoría de los ficheros de configuración locales del sistema (solo archivos de texto) **(no compartible)**.
* /root: directorio HOME del administrador.
* /sbin: ficheros ejecutables que, normalmente, solo el administrador puede ejecutar **(estático)**.
* /home: los directorios de trabajo de los usuarios.
* /lost+found: contiene “referencias” a los ficheros marcados como erróneos al chequear el sistema de ficheros.
* /lib: librerias necesarias para ejecutar los archivos.
* /proc y /sys: sistemas de ficheros virtuales, contienen información sobre procesos, núcleo, módulos cargados, dispositivos, sucesos...
* /tmp: ficheros temporales. Tiene el permiso t activo.
* /var: ficheros variables: colas de datos (spool) de impresión, e-mail..., ficheros del cron, atd, ficheros de log... **(dinámico)**.
* /boot: nucleo y ficheros necesarios para cargar el núcleo y ficheros de configuración del gestor de arranque **(estático)**.
* /mnt, /mount o /media: montaje de otros sistemas de ficheros: disquetes, cd-roms…
* /opt: paquetes de aplicaciones estáticas (no actualizables) **(compartibles)**.
* /usr: contiene subdirectorios de solo lectura, que no deben ser específicos de la máquina que los usa **(compartibles)**.

# Arranque y parada del sistema.

El proceso de arranque y parada se basa en la preparación del sistema para ser usado por los usuarios (arranque) y el proceso por el cual el sistema se deja consistente (parada). Todos estos procesos son sencillos y se basan en un conjunto de ficheros de configuración y de guiones Shell que determinan y controlan los procesos.

## Proceso de arranque.

El proceso de arranque tiene dos fases, las cuales son el arranque del hardware y el del Sistema Operativo (SO).

### Proceso de arranque completo.

1. Iniciador ROM:
   * Chequeo inicial del sistema
   * Lee y almacena en memoria el programa cargador del SO.
   * Pasa el control al cargador del SO, saltando a la dirección de memoria donde lo ha almacenado.
2. Cargador del sistema operativo (GRUB):

Carga el núcleo del SO y le pasa el control, sabe dónde está el núcleo.

1. Núcleo del SO:
   * Chequeo hardware.
   * Creación e inicialización de las estructuras de datos, tablas…
   * Crea el proceso Init y le pasa el control.
2. Proceso Init:

Termina el proceso de arranque, dejando el sistema preparado para ser usado (chequeo de SFs, montaje de SFs, activación de la swap, de cuotas, demonios, etc.).

### Gestor de Arranque GRUB

El GRUB (GRand Unified Bootloader) se instala en el master boot record (MBR) y hace de las funciones de master boot program (MBP, programa cargador). Se encarga de preguntar que SO arrancar. En sí, el GRUB permite (durante la selección del SO):

* Editar las entradas
* Mediante una consola interactiva GRUB, la ejecución de comandos para arreglar el arranque (seleccionar otro initrd, cargar módulos…).
* Terminología de GRUB, numerando los dispositivos según los reconozca la BIOS empezando en cero.

### Modo monousuario.

El modo monousuario es un estado del sistema, el cual está definido para realizar tareas administrativas y de mantenimiento que requieren un control completo y no compartido. Solo realiza el montaje del sistema de ficheros raíz (/), los otros SF están disponibles, pero no están montados. Se puede acceder a todo el sistema, pero:

* Muy pocos demonios están ejecución, solo los necesarios.
* Muchas utilidades no están activas.
* Solo las órdenes del SF raíz están disponibles (si /usr esta en otra partición, no está montado).

### Niveles de ejecución de GNU/Linux.

El SO puede estar en distintos niveles de ejecución (no solo modo monousuario y multiusuario). En GNU/Linux, los niveles de ejecución son:

* Nivel 0: Sistema apagado.
* Nivel 1: Modo monousuario, rescue o troubleshooting.
* Nivel 2: Modo multiusuario sin funciones de red.
* Nivel 3: Modo multiusuario con funciones de red y terminales de texto.
* Nivel 4: Sin usar, a redefinir por el administrador.
* Nivel 5: Modo multiusuario con funciones de red e inicio de sesión gráfico.
* Nivel 6: Sistema reiniciándose.

Para saer en qué nivel está el sistema se emplea “/sbin/runlevel” y para cambiar el nivel de ejecución se emplea “/sbin/telinit”. El nivel por defecto, establecido al arrancar, está en “/etc/inittab”.

### Ficheros de inicialización.

Se encargan de personalizar niveles de ejecución. Todos ellos son ejecutados por Init durante el arranque. Se ejecutan al arrancar o al cambiar de nivel.

### Upstart.

Es un proceso de arranque/parada del sistema basado en eventos, reemplazo de clásico Init. Este proceso realiza, de forma asíncrona, las siguientes tareas:

* Dirige el inicio de las tareas y demonios.
* Controla los demonios mientras el sistema esta encendido.
* Detiene los demonios durante el proceso de apagado.

En el directorio /etc/init/ hay una serie de ficheros de configuración de eventos (evento.conf) que Init ejecuta según el orden y las dependencias establecidas e los mismos. Estos eventos indican que tarea ejecutar, cuando y como, mediante su propio lenguaje.

Mediante initctl, el administrador interactúa con Init, para decirle que realiza determinadas acciones.

Dentro de los ficheros de configuración de eventos, encontramos diferentes funciones:

* exec <orden> <argumentos>: ejecuta la orden con los argumentos indicados.
* script … end script: ejecuta el guión shell indicado.
* start on <event>: describe bajo qué condiciones se lanzará ese evento.
* stop on <event>: describe bajo qué condiciones se parará ese evento.
* respawn: volver a lanzar ese proceso o demonio cuando se pare.
* console: hacia donde redirigir la salida del evento.
* pre-start: ejecutar la orden/guion Shell antes de lanzar ese proceso.
* pre-stop: ejecutar la orden/guion Shell antes de parar ese proceso.
* post-start: ejecutar la orden/guion Shell después de lanzar ese proceso.
* post-stop: ejecutar la orden/guion Shell después de parar ese proceso.

## Parada del Sistema.

Las acciones durante el proceso de parada son:

1. Se notifica a los usuarios
2. Procesos en ejecución -> enviar la señal de terminación (TERM)
3. Se paran los demonios
4. A los usuarios que quedan conectados se les echa del sistema.
5. Procesos que queden en ejecución -> enviar la señal de fin (KILL).
6. Actualizaciones de disco pendientes (integridad del SF) con *sync*.

Se puede apagar el sistema mediante el comando shutdown.

## Caídas del sistema y problemas de arranque.

Algunas de las posibles causas de caídas del sistema son los fallos hardware, errores de hardware irrecuperables, fallos de luz, otros problemas ambientales, problemas de entrada/salida o problemas de algún sistema de ficheros.

Los problemas de arranque se pueden deber a los fallos hardware, que no se pueda leer el sistema de ficheros de los discos de trabajo, que haya tareas dañadas en el disco que no pertenezcan al sistema de ficheros, hardware incompatible y errores en la configuración del sistema.

Al rearrancar se pueden mirar los mensajes que hay en el fichero “/var/log/messages”. La orden dmesg proporciona los mensajes producidos durante el arranque.

# Gestión de Usuarios.

## Introducción.

Un usuario es una persona que trabaja en el sistema, editando ficheros, ejecutando programas, etc.

Un pseudo-usuario es una entidad, que, sin ser una persona, puede ejecutar programas o poseer ficheros.

Un usuario tiene como características un nombre de usuario, un identificador de usuario (UID) (cabe destacar que el sistema emplea el UID y no el nombre de usuario en sí) y un identificador de los grupos a los que pertenece (GID).

Los ficheros de configuración son:

* /etc/passwd: contiene información de las cuentas de los usuarios.
* /etc/shadow: contiene passwords cifradas e información de “envejecimiento” de las cuentas.
* /etc/group: contiene la definición de los grupos y usuarios miembros.
* /etc/gshadow: contiene las passwords de grupos cifradas.

## Usuarios.

### Fichero /etc/passwd, contraseñas y shadow passwords.

#### Fichero /etc/passwd

Este fichero contiene la lista de usuarios del sistema y sus contraseñas.

Su formato es *nombre:password:uid:gid:gecos:home:shell* :

* Nombre: username
* Password: contraseña cifrada o:
  + “\*” o “!!” si la cuenta esta desactivada o bloqueada.
  + “x” si las shadow están activas, la contraseña cifrada está en /etc/shadow
* uid: identificador de usuario.
* gid: identificador del grupo primario al que pertenece
* gecos: campo de información referente al usuario
* home: Path del directorio $HOME del usuario
* shell: interprete de órdenes.

El propietario de este fichero es root y el grupo root. Los permisos del fichero son rw-r--r--. Mediante pwck verificamos la integridad de /etc/passwd y /etc/shadow. Se puede acceder al fichero /etc/passwd en modo lectura para leer información del usuario, pero no deberían poder accederse a las contraseñas (aunque estén cifradas).

### Contraseñas.

Mediante passwd <nombre\_usuario> asignamos una contraseña a un usuario (o la cambiamos). Se debe forzar de manera periódica, tanto a los usuarios como administradores a cambiar la contraseña.

### Shadow passwords.

Permiten que las contraseñas cifradas no se guarden en el fichero /etc/passwd sino en /etc/shadow, el cual es un fichero más restringido. Esta dirección tiene los permisos rw-------, y el usuario y grupo propietario es el root. Este fichero guarda para cada usuario del sistema, la contraseña cifrada junto con su información de envejecimiento, siempre y cuando, dichos usuarios tengan una “x” en /etc/passwd. Por defecto, están activas y se actualizan automáticamente. La estructura de este archivo es la siguiente:

*nom:pass:changed:minlife:maxlife:warn:inactive:expired:unused*

Mediante comandos como pwconv, creamos y actualizamos el fichero /etc/shadow, y mediante pwunconv, desactivamos los shadow passwords.

Para cifrar una contraseña, se utilizan algoritmos criptográficos de generación de resumen (función hash, H(.)). Los algoritmos de hash pueden ser de tipo:

* MD5 (Message-Digest algorithm):

Aplica funciones no lineales a los 17 segmentos de 32 bits de un bloque de 512 bits. Se obtiene un resumen de 128 bits. Para obtener una suma MD5 se aplica el comando md5sum, para comprobar la suma MD5 se emplea md5sum -c

* SHA (Secure Hash Algorithm):

Es el algoritmo estándar del NIST. Parecido a MD4, pero genera resúmenes más grandes, que lo hacen más seguro contra ataques de fuerza bruta o del cumpleaños. Se pueden considerar 160, 224, 256, 384 o 512 bits para el resumen. Para obtener la suma SHA se emplea shasum y para comprobar su suma se emplea shasum.

SHA-512 es el algoritmo empleado por defecto en GNU/Linux para guardar la contraseña.

### Restricciones de tiempo.

En /etc/shadow podemos encontrarnos restricciones de tiempo o envejecimiento para la validez de la cuenta o de la contraseña. Dichas restricciones de tiempo son:

* changed: fecha del último cambio de contraseña.
* minlife: número de días que han de pasar para poder cambiar la contraseña.
* maxlife: número de días máximo que se puede estar con la misma contraseña.
* warn: información para el usuario de cuantos días quedan antes de que la contraseña expire.
* inactive: número de días después de que la contraseña expire en que la cuenta se deshabilitara si no ha sido cambiada.
* expired: fecha en la que la cuenta expire y se deshabilita de forma automática.

Todos los valores por defecto para los parámetros anteriores se encuentran en /etc/login.defs y mediante el comando chage (administrador) podemos modificar dichos parámetros.

### Fichero de inicialización.

Los ficheros de inicialización son scripts de Shell que realizan tareas como dar valor a variables, nombrar alias, realizar funciones específicas, etc. Incluyen el PATH, variables de entorno, umask, funciones de inicialización… Su directorio es */etc/skel/*

### Selección de interprete de órdenes.

En el último campo del fichero de /etc/passwd, se establece el intérprete de órdenes que se ejecuta al entrar al sistema. En el fichero /etc/shells se indican los shells permitidos. Un usuario puede cambiar su Shell mediante chsh. Si un usuario no tiene asignado ningún interprete de órdenes, se empleará el shell por defecto (/bin/sh).

### Cuentas restrictivas.

Las cuentas restrictivas permiten limitar las acciones de los usuarios en el sistema. Se pueden crear de dos formas:

* Asignando como shell un fichero ejecutable que realice una tarea determinada y al terminar se sale del sistema.
* Usando el shell restrictivo /bin/rbash:

rbash es un enlace simbólico a /bin/bash (rbash = /bin/bash -r). Este interprete se comporta como un intérprete normal, salvo que el usuario no puede hacer determinadas tareas, como:

* + Cambiar de directorio.
  + Establecer o modificar los valores de $PATH o $HOME.
  + Especificar nombres u ordenes que contengan /.
  + Usar redirección.
  + Utilizar la orden exec para reemplazar el shell por otro programa.

A este tipo de usuarios hay que limitar los ficheros que pueden ejecutar, copiándolos a un fichero y que su PATH sea solo ese directorio. Si no se hace esto, es prácticamente como si no tuvieran restricciones.

### Herramientas para crear/modificar cuentas de usuario.

Las herramientas de creación de cuentas de usuario suelen realizar todas las tareas básicas del proceso, a excepción de las especificas:

* adduser/useradd: crea cuentas de usuario o modifica las ya existentes. Toma los valores por defecto de /etc/default/useradd y de /etc/login.defs
* usermod: modifica cuentas.
* deluser/userdel: elimina cuentas
* newusers: crea cuentas de usuario utilizando la información introducida en un fichero de texto (en batch), que ha de tener el formato del fichero /etc/passwd (no copia los ficheros de inicialización).
* users-admin: herramienta en modo gráfico.

## Grupos.

Los grupos son colecciones de usuarios que comparten recursos o ficheros del sistema. Sus características son el nombre del grupo (groupname) y el identificador del grupo (GID) (el sistema identifica al grupo por este número).

El fichero de configuración es /etc/group, con el siguiente formato:

nombre:x:gid:lista de usuarios

* Nombre: nombre del grupo.
* GID: Identificador del grupo.
* Lista de usuarios que pertenecen al grupo, separados por “,”.

Los grupos pueden tener contraseña, encontrada en /etc/gshadow. Si un usuario sabe la contraseña de un grupo, puede usarlo sin pertenecer a el con la orden newgrp.

Los grupos pueden ser primarios (grupo especificado en /etc/passwd) o secundarios (otros grupos (indicados en /etc/group)).

El funcionamiento de los grupos se basa en que, al crear un fichero se establece como grupo propietario el grupo activo del usuario en ese momento (el grupo activo será el mismo que el grupo primario a menos que usemos newgrp). Al determinar los permisos sobre un fichero, se usan todos los grupos del usuario.

### Comandos de los Grupos.

* addgroup grupo: crear un grupo.
* groupmod grupo: modifica un grupo existente.
* delgroup grupo: elimina un grupo.
* newgrp grupo: cambia de grupo activo (lanzando un shell).
* gpasswd grupo: asigna una contraseña a un grupo.

Nota!: Si el usuario no pertenece al grupo, pero el grupo tiene contraseña, se le solicita y pasa a ser su grupo activo.

* gpasswd -a user grupo: añade un usuario a un grupo.
* groups [usuario]: grupos a los que pertenece un usuario.
* id [usuario]: lista el identificador del usuario y los grupos a los que pertenece.
* grpck: comprueba la consistencia del fichero de grupos.

## Usuarios y grupo estándar.

### Rangos del UID.

* UIDs del 0 al 99 son usuarios que representan al propio SO.
* UIDs del 100 al 499 son usuarios especiales que representan servicios o programas.
* UIDs mayores o iguales al 1000 son usuarios normales.

### Algunos usuarios y grupos estándar.

* Usuarios estándar:
  + root: cuenta del administrador
  + bin, daemon, lp, sync, shutdown, etc: tradicionalmente usados para poseer ficheros o ejecutar servicios.
  + mail, news, ftp: asociados con herramientas o facilidades,
  + postgres, mysql, xfs: creados por herramientas instaladas en el sistema para administrar y ejecutar sus servicios.
  + nobody o nfsnobody: usado por NFS y otras utilidades, usuario sin privilegios.
* Grupos estándar:
  + root, sys.
  + bin, daemon, adm, lp, disk, mail, ftp, nobody, etc
  + kmem: grupo propietario de los programas para leer la memoria del kernel.
  + user o users: grupo de los usuarios normales (no siempre se usa).

# Gestión de los recursos del sistema.

## Actividad de la CPU.

### Procesos.

Un proceso es un programa en ejecución. Un sistema de tiempo compartido como GNU/Linux permite múltiples usuarios que ejecuten múltiples procesos, pero la CPU solo puede ejecutar un proceso a la vez. La CPU conmuta rápidamente de un proceso al siguiente, ejecutando un cuanto (p.ej. 100ms) de cada proceso. El SO es el encargado de decidir que proceso se ejecuta en qué lugar (planificación de la CPU).

Un proceso tiene dos modos de ejecución, diferenciados para proteger mejor las direcciones de memoria a las que puede acceder un proceso:

* Modo usuario: se ejecuta código normal del programa.
* Modo núcleo: se ejecutan las funciones del núcleo, lo cual conllevan llamadas al sistema, excepciones e interrupciones.

Existen varios tipos de procesos:

* Procesos de usuario:

Estos procesos son creados por un usuario real y se ejecutan en modo usuario

* Procesos demonio:

No están asociados a un usuario, o asociados a uno ficticio. Se ejecutan en modo usuario y realizan tareas periódicas relacionadas con la administración del sistema.

* Procesos núcleo:

No están asociados a un usuario, corresponden al código del kernel, se ejecutan siempre en modo núcleo y componen las tareas de administración más delicadas (planificación, intercambio de procesos, intercambio de páginas, etc).

Para monitorizar los procesos en ejecución, se emplea el comando ps, el cual informa de los siguientes aspectos:

* USER: usuario que lanzó el programa.
* PID: identificador del proceso.
* PPID: identificado del proceso padre.
* %CPU: porcentaje de la CPU consumido por este proceso.
* %MEM: fracción de memoria consumida.
* VSZ: tamaño virtual (código+datos+pila) en KB.
* RSS: memoria real usada en KB (VSZ incluye a RSS).
* TTY: terminal asociada al proceso.

Mediante STAT hallamos el estado del proceso hallado en ps, el cual, puede significar:

* R 🡪 en ejecución (listo para ejecutarse en cuanto la CPU este libre, ya que tiene todos los recursos que necesita y esta esperando su cuanto para ejecutarse).
* S 🡪 durmiendo (esperan a que ocurra un evento especifico).
* T 🡪 parado (el proceso ha sido detenido temporalmente mediante señales (Ctrl+Z) o porque está siendo examinado (trace). Solo continuaran ejecutándose tras otra señal)
* Z 🡪 proceso zombie (el proceso termina correctamente pero el padre no recoge su código de error)
* D 🡪 durmiendo ininterrumpible (algunas operaciones causan este estado, en el que el proceso no manejas señales, solo despertara cuando pase el evento)
* s 🡪 líder de sesión (los procesos se pueden agrupar, y si se manda una señal al grupo, se le manda a todos los procesos)
* l 🡪 hilos creados con CLONE\_THREAD.
* L 🡪 el proceso ha pedido al kernel bloquear determinadas paginas de memoria, para evitar que no se modifiquen mientras se hacen determinadas operaciones.
* + 🡪 foreground (proceso de primer plano)

### Prioridad y señales.

Cuando planificamos procesos por prioridades dinámicas, al lanzar dichos procesos, les son asignados un numero nice o prioridad estática (la cual, es heredada del proceso padre). La prioridad por defecto se obtiene mediante el numero nice, el cual, si toma un valor alto (positivo) tendrá menos prioridad que si toma un valor bajo (negativo), que tendrá más prioridad.

El rango de prioridad estática está entre -20 y 19. La asignación de prioridades mayores o menos que la actual se hace mediante el comando nice. Algunos ejemplos son los siguientes:

* *nice -5 nautilus*: lanzar nautilus con numero nice incrementado en 5.
* *nice –10 nautilus*: lanzar nautilus con numero nice decrementado en 10 (solo root).
* *renice 14 890*: prioridad 14 al proceso 890.
* *renice 5 -u v1cky*: prioridad 5 para todos los procesos del usuario v1cky.

Para enviar señales a los procesos (con el fin de pararlos, hacer que continúen, eliminarlos, etc.) se emplean los siguientes comandos:

* *kill pid*: manda señal por defecto al proceso pid
* *SIGKILL*: fuerza la salida del proceso. No se puede capturar!
* *SIGSTOP*: para un proceso.
* *SIGCONT*: reinicia un proceso
* *killall comando*: permite mandar una señal a todos los procesos con un determinado nombre de comando.
* *pskill o skill*: envía una señal usando el nombre u otros atributos o criterios.
* *HUP*: Hang up (terminal).

Para HUP, si se trata de demonios debería provocar que se reinicien (volviendo a leer su información), aunque si se tratan de procesos iniciados en una terminal, se manda al cerrar la terminal.

* *INT*: Interrumpir (equivalente a Ctrl+C). Se puede bloquear y si se manda a un intérprete de órdenes, podría cancelar la orden que esta ejecutando, pero no el programa completo.
* QUIT: Similar a TERM (parar software) pero hace un core dump.

Nota: Los procesos en estado D o Z no se detienen, aunque reciban la señal KILL.

### Monitorizar el uso de la CPU.

Mediante *uptime* podemos obtener cuánto tiempo lleva en marcha el sistema, el número de usuarios conectados y la carga media del sistema. Cabe destacar que si se devuelve un valor muy alto esto implica que el sistema se está usando mucho.

Mediante *pstree* podemos visualizar un árbol de los procesos en ejecución.

Mediante *top* obtenemos una visión continua de la actividad del procesador, en tiempo real, mostrando las tareas que hacen más uso de la CPU. Además, permite manipular procesos de forma interactiva. Las cinco primeras líneas muestran información general:

* Estadísticas uptime.
* Resumen de procesos del sistema.
* Porcentaje de tiempo de CPU gastado en modo usuario (us), modo núcleo (sy), procesos esperando eventos E/S (wa), tratando interrupciones (hi o si), espera involuntaria en virtualización (st).
* Estado actual de la memoria física (total disponible, usada, libre, usada en buffers).
* Espacio swap (total disponible, usada, libre, usada en buffers, usada en cache de página).

Para tomarlo como referencia, se podría decir que hallamos una tabla como ps excepto que se muestran los valores de SHR que muestran la memoria compartida disponible para ser utilizada. En este comando, los procesos son ordenados decrecientemente por uso de CPU y la lista se actualiza cada (normalmente) 5 segundos. Se pueden realizar tareas sobre los procesos como cambiar la prioridad, matar o enviar señales, ordenarlos según diferentes criterios, cambiar el número de procesos que se muestran, mostrar un usuario en específico, mostrar información independiente por cada procesador, etc. De forma equivalente, con htop podemos hacer lo mismo, pero con colores.

Mediante *vmstat* obtenemos información sobre memoria virtual, desde el ultimo reinicio, mostrando información como:

* r: numero de procesos esperando su tiempo de ejecución
* b: numero de procesos en espera ininterrumpible.
* us: tiempo de usuario como porcentaje de tiempo real (modo usuario).
* sy: tiempo del sistema como porcentaje de tiempo total (modo núcleo).
* id: tiempo de inactividad como porcentaje de tiempo total.
* wa: tiempo usado en espera de E/S.

Tanto pc como top leen la información que necesitan de */proc*. Cada proceso tiene una carpeta (cuyo nombre es el pid) y en cada carpeta hay información de este.

### Programar ejecución de procesos.

Mediante *at* podemos ejecutar tareas a una determinada hora. Permite recibir un fichero de texto con las órdenes a ejecutar. Dispone de un prompt para ir introduciendo las órdenes. Tiene diferentes variantes:

* atd: demonio que ejecuta las órdenes.
* atq: consulta la lista de órdenes.
* atrm: eliminar órdenes.

Mediante *cron* podemos ejecutar tareas periódicamente. Tiene las variantes:

* crond: demonio encargado de ejecutar las órdenes.
* crontab: establece las tareas a ejecutar (-e: añadir/modificar tareas, -l: lista tareas, -r: eliminar tareas).

El formato de crontab es “*minuto hora dia\_mes mes dia\_semana [user] comando*”. Se interpreta como una conjunción de condiciones, salvo para dia\_semana y dia\_mes.

De no estar encendido el ordenador cuando se ha especificado el lanzamiento del proceso, cron no lo lanza.

### Rastreo de procesos.

El comando *strace* nos permite observar que es lo que esta haciendo un proceso. Muestra cada llamada al sistema que hace y cada señal que recibe:

* strace -p pid: rastrea un proceso ya iniciado
* strace comando: inicia un proceso y lo rastrea.
* strace -o salida.txt comando: utiliza un fichero para guardar la salida.

Como administradores, debemos sospechar cuando un proceso acapara mucha CPU, ya que se puede tratar de un proceso acaparador. Si tras saber que están haciendo parecen legítimos, deberíamos suspender con STOP, reiniciarlos y reanudarlos con CONT; si no parecen legítimos simplemente matamos el proceso.

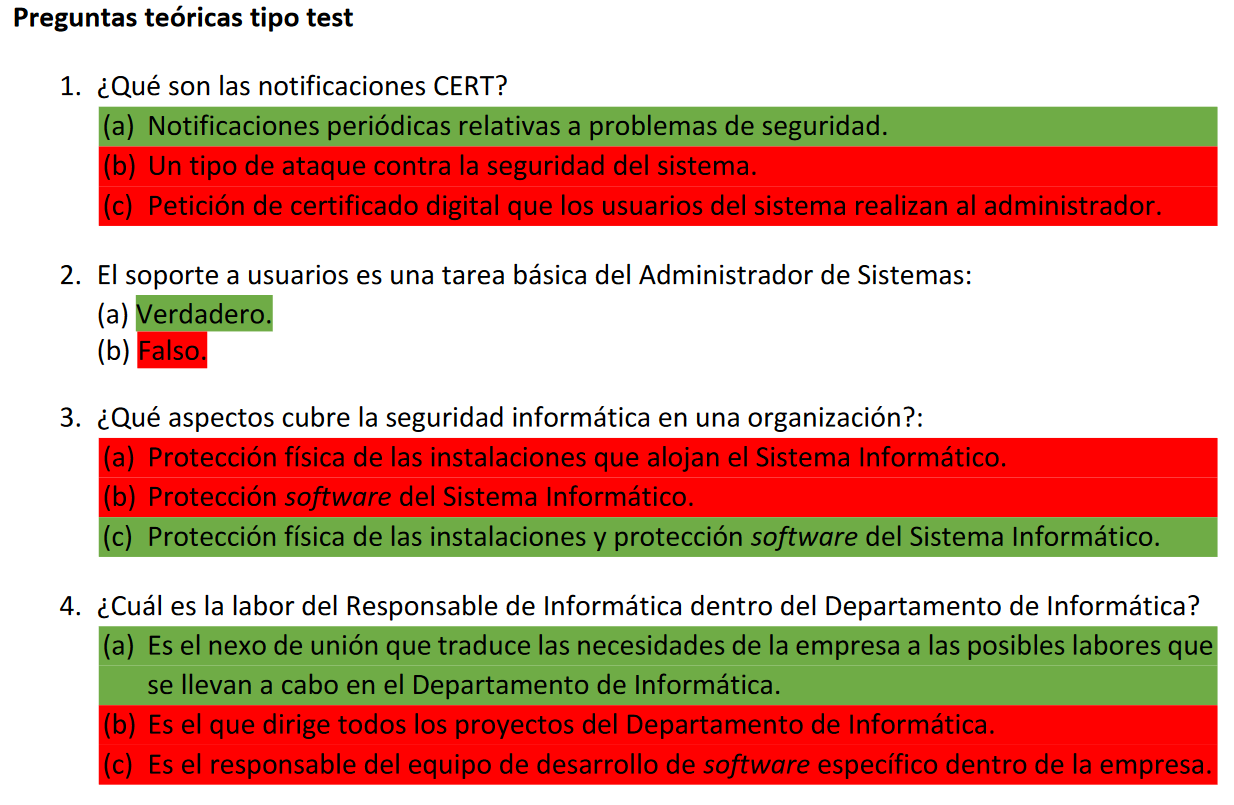
También debemos de gestionar la RAM y la zona de intercambio a base de intercambios y paginación. Para ello empleamos comandos como vmstat, el cual, nos muestra la cantidad de memoria virtual ocupada (swpd), sin usar (free), empleada como buffers para E/S (buff), cantidad de memoria empleada como cache de disco (cache), bloques recibidos (bi) y enviados a un dispositivo de bloques (bo), etc.

En cuanto al espacio para paginación, debemos de asignarle un tamaño basados en la demanda del sistema, la memoria requerida por los procesos, numero de procesos simultáneos, etc. Para gestionarlo empleamos el comando *swapon* el cual nos permite visualizar, activar o desactivar una partición. Mediante *free* obtenemos información sobre el uso de memoria. Mediante *df* se muestra la capacidad, el espacio libre y el punto de montaje de cada sistema de ficheros del equipo. Mediante *du* se muestra el espacio usado por cada subdirectorio del directorio actual. Mediante *iostat intervalo número* se presentan estadísticas de la CPU y los dispositivos y particiones de E/S, como numero de transferencias por segundo, bloques leídos/escritos por segundo y total de bloques leídos/escritos.

# Organización de sistemas de ficheros y discos.

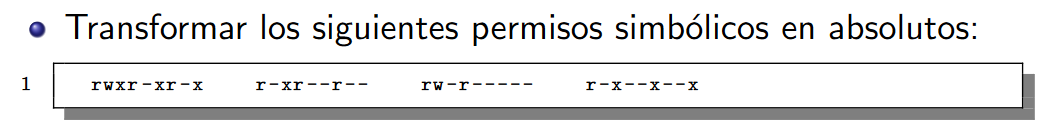
# Ejercicios de cada tema.

## Tema 1.

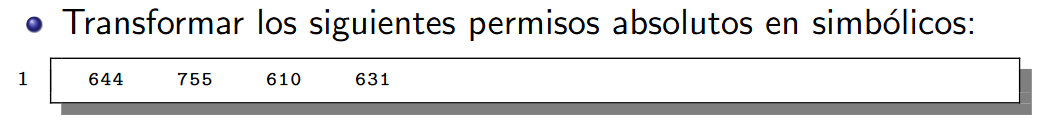


(Hay dos preguntas más en el Moodle pero (a mi parecer) son un poco por los memes más que por otra cosa, así que, allí se quedan)

## Tema 2.



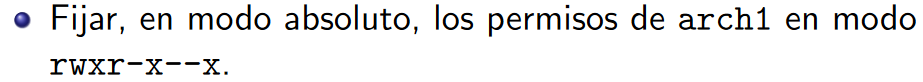
rwxr-xr-x 🡪 755 ; r-xr--r-- 🡪 544 ; rw-r----- 🡪 640 ; r-x—x—x 🡪 511



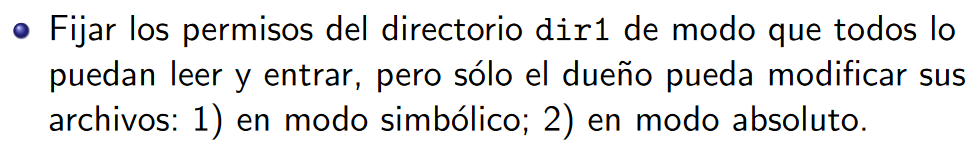
644 🡪 rw-r--r--r ; 755 🡪 rwxr-xr-x ; 610 🡪 rw---x--- ; 631 🡪 rw--wx--x



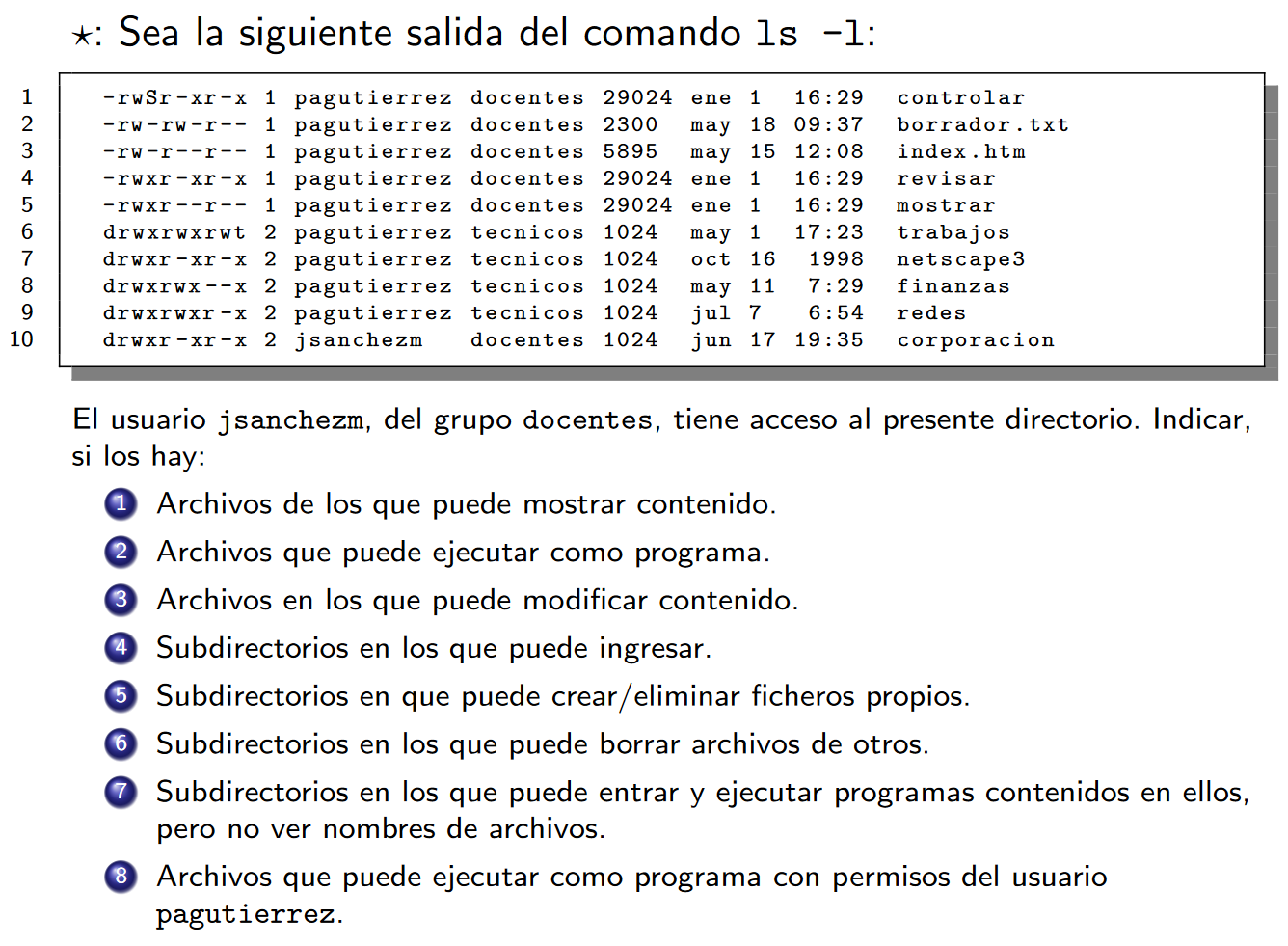
$chmod u+rwx, g+rx-w, o+r-wx arch1



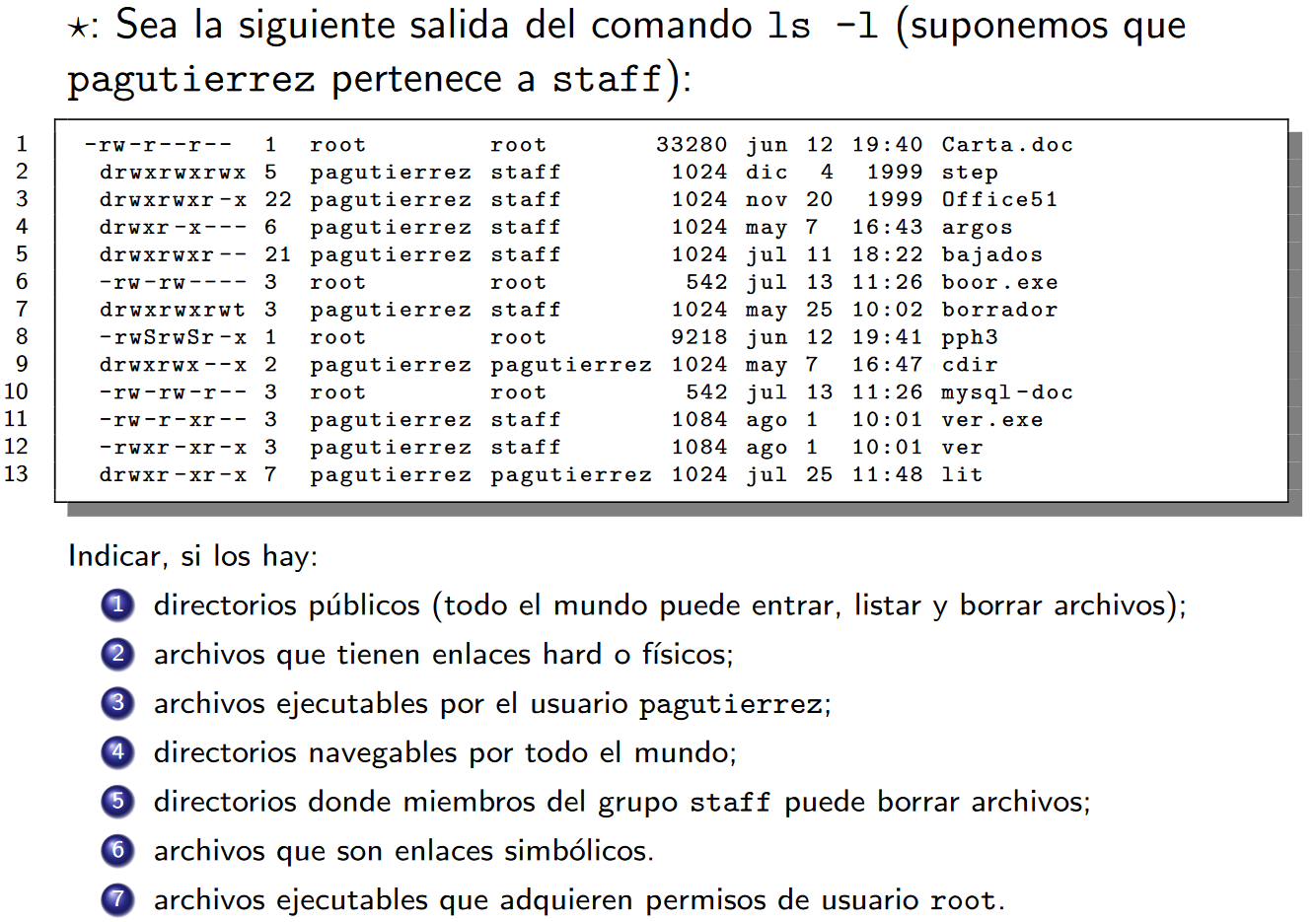
$chmod 751 arch1



1. $chmod a+rx-w, u+w dir1
2. $chmod 755 dir1

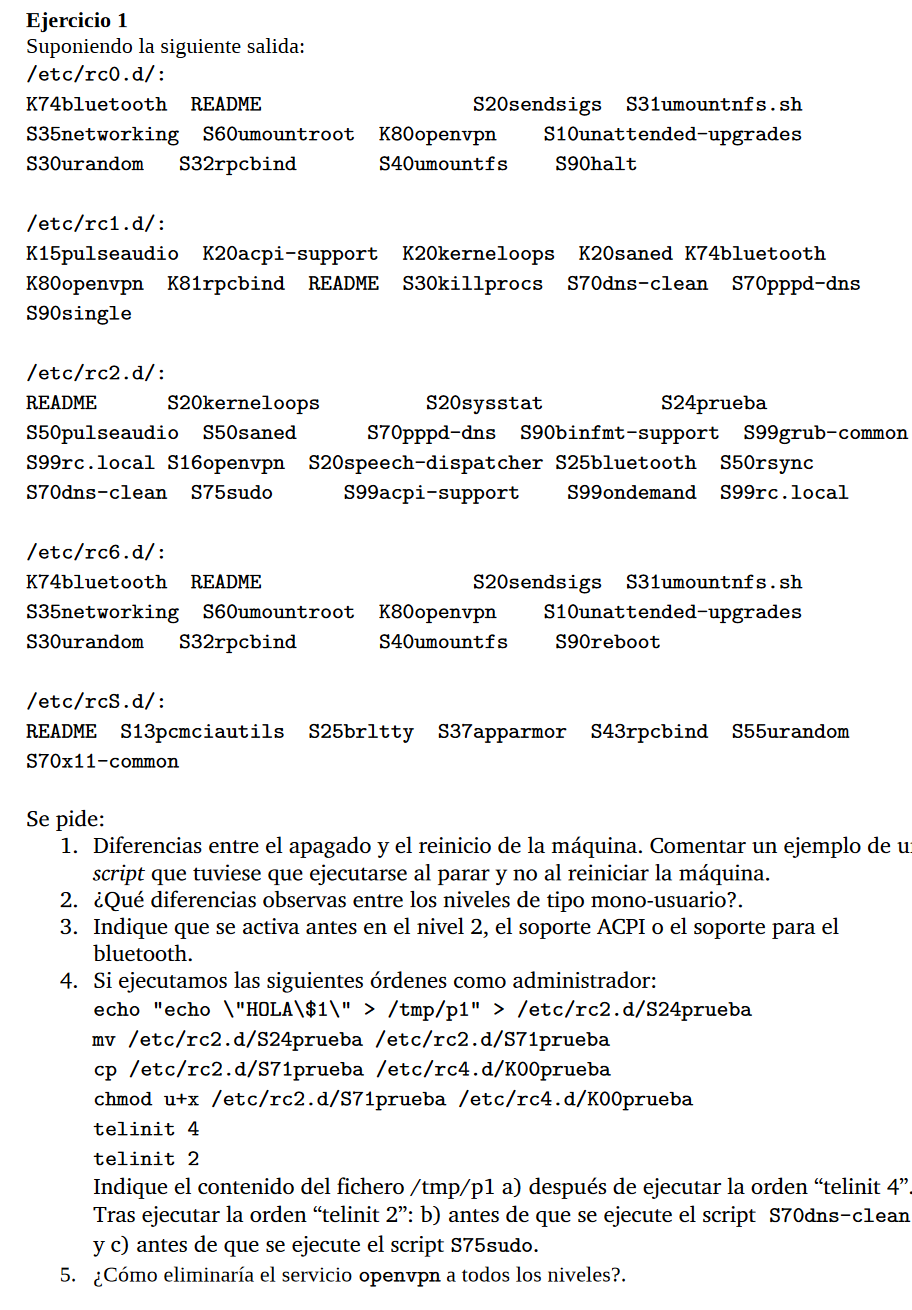


1. Todos los archivos (g+r)
2. Controlar y revisar (g+x)
3. borrador.txt (g+x)
4. trabajos, netscape3, finanzas y redes (o+x) corporación (u+x)
5. trabajos (o+w) corporación (u+w)
6. corporación (u+w)
7. finanzas (o+x, o-r)
8. controlar (g+x, u+s)



1. step, borrador (salvo borrar los de otros) (a+rwx)
2. Todos salvo Carta.doc y pph3 (2º columna > 1)
3. ver (u+x) y pph3 (o+x)
4. step, Office51, borrador, cdir, lit (o+x)
5. step, Office51, bajados, borrador
6. Ninguno, no hay flecha después del nombre ni una l al comienzo de los permisos.
7. pph3 (n+s)

## Tema 3.



1. La diferencia es el S90halt vs S90reboot.
2. La diferencia es que el rcS está pensado para arrancarse desde el grub, mientras que el rc1 no. Se ve porque el rc1 tiene una parte en la que mata demonios, mientras que el rcS no.
3. Bluetooth se ejecuta antes porque (S25 < S99), es decir, su prioridad es mayor.
4. Se crea un fichero /etc/rc2.d/s24prueba

echo “Hola $1” > /temp/p1

1. Lo renombramos a S71prueba
2. Lo movemos a rc4.d como k00prueba.
3. Damos permisos de usuario.

Apartado A:

Contiene “Hola stop”

Apartado B:

Contiene “Hola stop”

Apartado C:

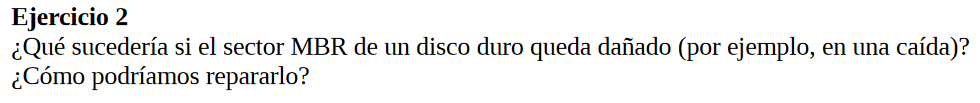
Contiene “Hola start”

Apartado 5 (no uso 5. porque Word esta especial hoy xd)

update-rr open-vpn remove

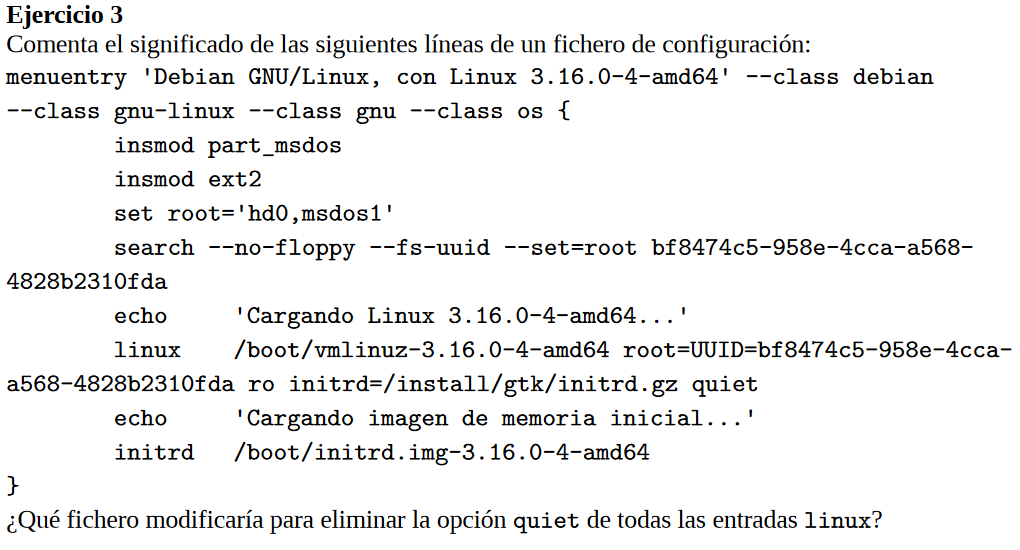
ó

rm /etc/rc?.d/\*open-vpn



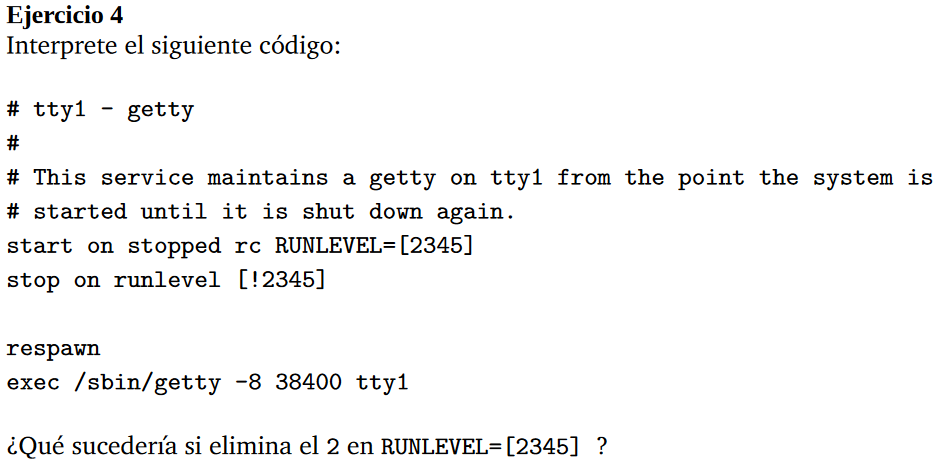
Perdemos el programa cargador, por tanto, no se puede arrancar el SO. Podríamos repararlo mediante:

$grub-install a través de un (LIVE CD/USB)



Fijamos una raíz, pero después buscamos la que realmente queremos. La opción quiet se puede modificar mediante:

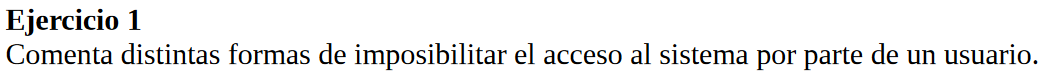
/etc/default/grub 🡪 GRUB\_CMDLINE\_LINUX\_DEFAULT



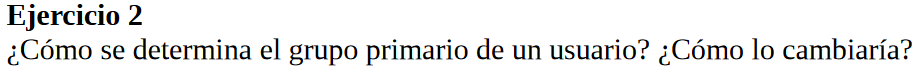
Es un fichero upstart para un fichero tty.

Si se elimina el 2, no se ejecutaría esta terminal en el modo 2 de arranque.

## Tema 4.



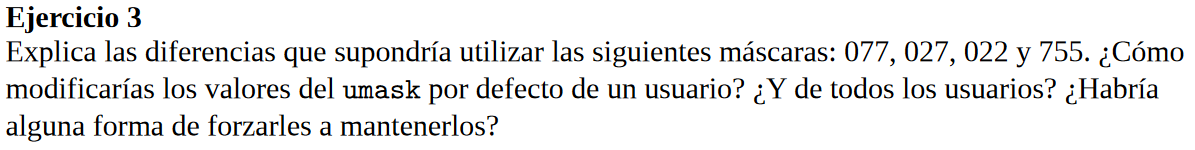
1. /bin/false /sbin/nologin (campo bash en /etc/passwd)
2. “!!” o “\*” en contraseña
3. Caducarle la cuenta



Sería el campo gid de /etc/passwd

Las formas de cambiarlo serian:

* Editar /etc/passwd a mano
* Mediante usermod

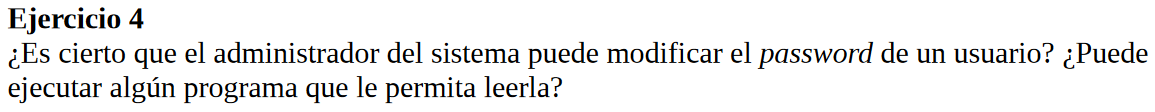


El resultado de aplicar las máscaras.

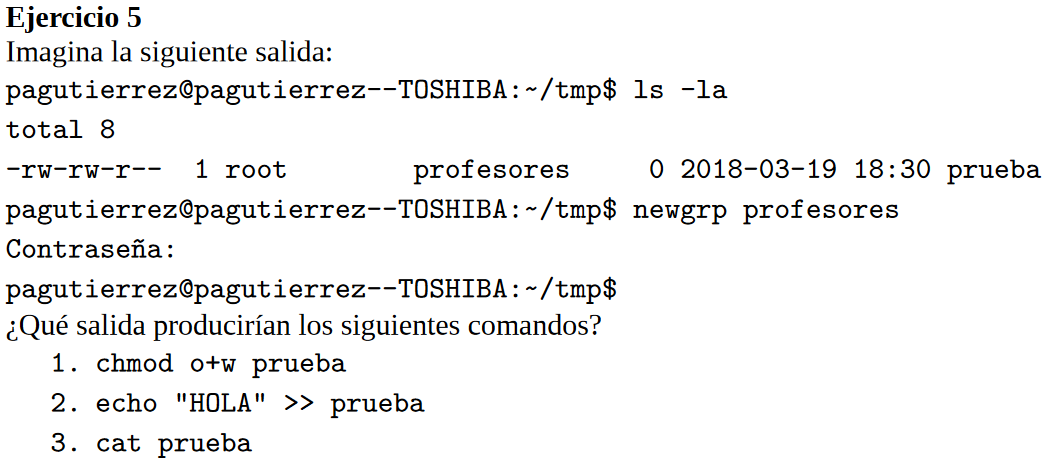
Para aplicar una máscara a un usuario habría que modificar .bashrc o .bash-profile para que aplique el cambio de mascara al inicio de la sesión del usuario.

Para obligar a que ese usuario use dicha mascara habría que quitarle los permisos sobre el comando, aunque en principio no habría forma.

Para realizar el cambio a todos los usuarios se usaría /etc/profile



1. Si puede modificar la contraseña mediante el passwd y editándolo.
2. No puede leerla en texto plano dado que este encriptado.



1. Daria un error porque solo el propietario de un fichero (o root) puede cambiar los permisos.
2. Si funciona, porque gracias al newgrp es como si perteneciese al grupo de profesores.
3. Si funciona, por el motivo anterior.