Introducción a la Administración Local de Linux

Ingeniería de Tecnologías de Telecomunicación

Departamento de Ingeniería Telemática (DIT)

Universidad de Sevilla

Fco. Javier Muñoz Calle

Francisco José Fernández Jiménez

ÍNDICE

TAREAS

En esta memoria encontrará explicaciones acompañadas de ejercicios que deberá realizar usted en el ordenador. Para ayudarle a distinguir lo que son explicaciones de lo que son actividades que usted debe realizar (solamente debe realizarlas, no hace falta que entregue nada), éstas últimas se encuentran señaladas con la palabra "**TAREAS**".

1. Objetivos y alcance (5 minutos)

1.1 Introducción

Linux Debian es un sistema operativo multitarea (tipo Unix) diseñado inicialmente para trabajar en PCs, aunque actualmente existen versiones para otro tipo de máquinas. La gran difusión de este sistema operativo se debe en parte al hecho de ser un software de libre distribución. Linux proporciona una buena plataforma para el desarrollo de servicios de información. Existen multitud de librerías de desarrollo y de paquetes software que añaden gran funcionalidad y servicios.

Como peculiaridades de la distribución Debian podemos citar las siguientes:

- Líneas de desarrollo: se mantienen 3 líneas o ramas: Stable (totalmente probada y estable, aunque con software con un año de antigüedad o más), Testing (en fase de pruebas para convertirse en estable, actualizada casi diariamente) y unstable o "sid" (en desarrollo, convirtiéndose los paquetes a testing tras un periodo de migración de varios dias).

- Versiones de la rama Stable: cada vez que se alcanza una nueva versión Stable, se le asocia un nombre en clave: Jessie (8), Wheezy (7), Squeeze (6.0), Lenny (5.0), Etch (4.0), Sarge (3.1), ...

Linux dispone de numerosas herramientas que permiten su administración tanto de forma gráfica como en modo texto desde la shell o intérprete de comandos. Asimismo, al igual que todas las plataformas Unix, dispone de mecanismos y herramientas de administración que facilitan el mantenimiento del sistema operativo.

1.2 Objetivo de la práctica

El objetivo fundamental de esta práctica es que se familiarice con las diferentes interfaces gráficas y de texto que posee Linux Debian. Concretamente se centrará la atención en interfaz gráfica “XFCE” así como en el uso del shell Bash (Bourne again shell). Además se proporcionarán las nociones necesarias para la administración básica del sistema operativo.

1.3 Documentación de apoyo

- "The Debian system: concepts and techniques". Martin F. Krafft. 2005. ISBN: 1593270690.
- "Linux bible". Christopher Negus. Indianapolis, IN. Wiley, 2011. ISBN: 9780470929988.
- "Pro Linux System Administration". James Turnbull, Peter Lieverdink, Dennis Matotek. Berkeley, CA. Apress, 2009. ISBN: 978-1-4302-1912-5.
- "Linux Network Administrator's Guide (Openbook)". Olaf Kirch, Terry Dawson. O'Reilly, 2000. ISBN: 1-56592-400-2
- Relación cronológica entre las distintas distribuciones Linux: http://futurist.se/gldt/
- Página Web de Linux Debian: http://www.debian.org/
- Recopilatorio de las distintas versiones de la rama Testing de la distribución Debian, por fechas: http://snapshot.debian.org/
- Versiones de Linux Debian: http://es.wikipedia.org/wiki/Debian
- Documentación de Debian: http://www.debian.org/doc/manuals/, http://wiki.debian.org/
- Gestores de sesión para X: http://wiki.debian.org/DisplayManager
- Gestores de ventana y escritorio para X: http://wiki.debian.org/WindowManager, http://wiki.debian.org/DesktopEnvironment, http://xwinman.org/
- Estándar de la jerarquía del sistema de archivos en sistemas Unix: http://www.pathname.com/fhs
- Estándar de “intérprete de comandos” de la especificación POSIX (IEEE Standard 1003.1 2004/ISO 9945.2): http://pubs.opengroup.org/onlinepubs/009695399/
- Documentación de XFCE: http://www.xfce.org/
- Estadística de uso de Sistemas Operativos para alojamiento de aplicaciones Web: http://w3techs.com/technologies/overview/operating_system/all

2. Uso básico de Linux (85 minutos)

2.1 Entorno de trabajo (5 minutos)

Se asumirá que los equipos tienen definidos dos usuarios:

Usuario	Clave
`dit`	`dit`
`root`	`root`

Tras el proceso de arranque, el sistema está configurado para que muestre la interfaz gráfica. Acceda al sistema (seleccione el tipo de sesión Xfce o “Default Xsession”) usando el usuario "dit" (por motivos de seguridad, en ocasiones el entorno gráfico está configurado para que el usuario "root" no pueda usarse para abrir una sesión gráfica).

De forma habitual, trabaje con el usuario "dit", empleando el usuario "root" cuando realice tareas de administración del sistema. Con esto se evitará posibles problemas como consecuencia de alguna operación incorrecta. Cuando trabaje en una consola de comandos, puede cambiar entre usuarios usando el comando "`su - nombre_usuario`" (volviendo al usuario anterior con "`exit`").

Existen muchas distribuciones de Linux (SUSE, Caldera, Slackware, Mandriva, Debian, Gentoo, Ubuntu, Guadalinex, etc). En el laboratorio se utilizará la distribución Debian. El número de la versión de Debian que actualmente está utilizando se encuentra en el fichero de texto "`/etc/debian_version`" (NOTA: en Linux, la mayoría de los archivos de configuración/administración corresponden a ficheros de texto, por lo que podrá consultarlos con cualquiera de las herramientas aptas para ellos: cat, emacs, nedit, ...).

TAREAS

Entre en el entorno gráfico con el usuario "dit". Usando los iconos del escritorio, abra un terminal y mire el contenido del archivo "`/etc/debian_version`" en su equipo.

2.2 Comandos de usuario del intérprete de comandos (40 minutos)

A continuación se resumen algunos comandos[1] de los intérpretes de comandos de Linux que debe conocer (si tiene alguna duda siempre puede consultar la ayuda con el comando man):

[1] Un comando es cualquier orden pasada a un "intérprete de comandos (shell)" para que sea procesada por éste (la orden de interpretación se realiza al pulsar la tecla [Enter], salvo que el último carácter sea el escape “\”, en cuyo caso el comando continúa en la siguiente línea). El comando más simple sería la invocación de un fichero ejecutable.

a) Comandosbásicos: `ls`, `cd`, `pwd`, `rm`, `mkdir`, `cp`, `mv`, `cat`, `echo`, `more`, `ps`, `kill`, `clear`, `reset`, …

b) Redirecciones y tuberías: uso de los descriptores de ficheros y operadores de redirección, especialmente:

`comando > fichero`	Guardar la salida estándar del "comando" en "fichero" (se crea si no existe; si existe se limpia antes).
`comando >> fichero`	Idem, pero añadiendo al final (sin limpiar antes el contenido del fichero)
`comando 2> fichero`	Guardar la salida de errores del "comando" en "fichero" (se crea si no existe; si existe se limpia antes).
`comando < fichero`	"comando" usa a "fichero" como entrada estándar
`comando1 | comando2`	La salida estándar de "comando1" es usada por "comando2" como entrada estándar

TAREAS

Sabiendo que el comando básico:

- "`cat`": imprime en pantalla el contenido de un fichero de texto (indicado como argumento) o lo leído de la entrada estándar (al ser invocado sin argumento de fichero).
- "`echo`": imprime en pantalla la cadena de texto pasada como argumento.
- "`more`": pausa el volcado en pantalla para ir visualizando la información pantalla a pantalla.
- "`clear`": limpia la consola, dejando sólo el prompt.
- "`reset`": resetea el terminal, resultando útil cuando manifiesta un comportamiento anómalo, mostrando caracteres extraños.

ejecute los siguientes comandos sobre un terminal con el usuario "dit" y analice el resultado:

```
echo "Texto"
echo "Texto" > /tmp/fichero
cat /tmp/fichero
echo "Texto2" > /tmp/fichero
cat /tmp/fichero
echo "Texto3" >> /tmp/fichero
cat /tmp/fichero
clear
cat /tmp/fichero
reset
ls /
ls / > /tmp/listado
cat /tmp/listado
clear
ls /dir-no_existe
ls /dir-no_existe > /tmp/fichero
ls /dir-no_existe 2> /tmp/error
cat /tmp/fichero
cat /tmp/error
cat < /tmp/error
clear
ls -l /
ls -l / | more
echo "Hola" | cat > /tmp/fichero
cat /tmp/fichero
```

c) Manuales de ayuda: `man [x] comando/fichero`[2] ("q" para salir, "espacio" página siguiente, "/" buscar, "n" siguiente ocurrencia de la palabra buscada, ...). Los manuales de ayuda se clasifican en secciones (valor de "x") según el tipo de información que contienen [5].

[2] La notación usada es escribir entre corchetes "[x]" aquellos parámetros opcionales.

A veces un mismo término puede hacer referencia a distintas informaciones (comandos, ficheros, funciones de programación) cada una de las cuales tiene una sección de man distinta. Si no se indica la sección, al escribir solamente "`man comando`", se verá la información de la primera sección (por ejemplo, "man 2 signal" y "man 7 signal" dan resultados diferentes; si se usa "man signal" se obtiene lo mismo que con "man 2 signal").

Además de mediante la consola de comandos, estos documentos se pueden consultar de forma gráfica, por ejemplo con el navegador konqueror (escribiendo "`man:comando`" en la barra de navegación). Recuerde que si en algún momento se le pide que ejecute algún comando y no recuerda cómo funciona, debe consultar la ayuda de dicho comando.

TAREAS

Realice las siguientes operaciones:

1º Consulte la ayuda del comando “passwd” mediante “man passwd” y compruebe que se obtiene la misma información con “man 1 passwd” (recuerde que para salir de un manual debe pulsar “q”). Observe cómo en la esquina superior derecha e izquierda aparece “PASSWD (1)”, lo que indica que es un manual de la sección “1”.

2º Consulte ahora la ayuda del fichero “/etc/passwd” mediante el comando “man 5 passwd”. Observe cómo ahora en la esquina superior derecha e izquierda aparece “PASSWD (5)”, lo que indica que es un manual de la sección “5”.

3º Consulte la ayuda del intérprete de comandos Bash usando “man bash”. Una vez dentro de la ayuda:

- Pulse los cursores abajo y arriba, viendo como va bajando y subiendola información mostrada en pantalla.
- Compruebe cómo cada vez que pulsa la tecla “Espacio”, la información mostrada avanza una pantalla.
- Busque la palabra “alias” en todo el manual de Bash: para ello, escriba el carácter “/” (pulsación “May-7”) escriba la palabra “alias” y pulse Enter. Verá que aparecen marcadas todas las palabras “alias”.
- Pulse la tecla “n” (Next”), comprobando que cada vez que la pulsa, la pantalla va avanzando hasta la siguiente aparición de la cadena buscada “alias”.
- Salga del manual (tecla “q”).

d) Directorios/ficheros especiales del sistema:

Nombre	Significado
`/`	Raíz del sistema de ficheros
`./`	Directorio actual
`../`	Directorio padre
`~/`	Directorio personal del usuario (“cd”, “cd ~”, “cd $HOME” ó “cd /carpeta_personal” son equivalente).
`.fichero`	Fichero oculto (visualizables con “ls –a”).“./” y “../” son “ficheros” (directorios) ocultos

Asimismo, recuerde que los nombres de ficheros y directorios son "case sensitive" (distinguen entre mayúsculas y minúsculas). Por ejemplo,"/tmp/file" y "/tmp/File" son ficheros distintos.

e) Sistema de permisos: Usuario/grupo propietarios y permisos de lectura, escritura y ejecución de ficheros/directorios (visualizables, por ejemplo, con el comando “`ls -l`”).

f) Inserción rápida de comandos (si bien estas características son dependientes del intérprete de comandos, suelen ser comunes a todos ellos):

- Scroll de pantalla ("May-RePag" y "May-AvPag"),
- Autocompletado (tecla de "Tabulación"): si sólo existe una opción, se completa; si existe más de una, muestra la lista de los mismos. Recuerde que si se usa en la primera palabra, completa el nombre del comando, mientras que si se emplea en la segunda y sucesivas completa los nombres de ficheros que estén en el directorio en curso.
- Histórico de comandos: comando "`history`" (muestra los comandos anteriormente introducidos, almacenados en el histórico de comandos -para el intérprete Bash, fichero "`.bash_history`" ubicado en el directorio personal del usuario-ver [9]) y cursores arriba/abajo (muestran los comandos anteriores y posteriores del histórico). Asimismo, se dispone, entre otras, de las siguientes instrucciones para obtener rápidamente comandos del histórico:

Comando	Función
`!<prefijo_comando>`	Ejecuta el último comando introducido que empiece por "<prefijo_comando>"
`!!`	Ejecuta el último comando introducido (equivale a "!-1")

Comando	Función
`!n`	Ejecuta el comando número "n" ("history" da el número de cada comando)
`!-n`	Ejecuta el comando número “n” empezando por el último
`!?<cadena>?`	Ejecuta el último comando introducido que contenga <cadena>
`Ctrl-R`	Busca "interactivamente" y ejecuta el último comando introducido que contenga una determinada cadena ("`history | grep <cadena>`" sólo muestra los comandos)

Recuerde asimismo que en Linux, tanto los comandos como los nombres de ficheros son sensibles a mayúsculas y minúsculas.

TAREAS

Para practicar con estos comandos y herramientas, realice las siguientes operaciones:

1º Usando el terminal que abrió antes, ejecute los siguientes comandos y analice cuál es la función que realiza cada uno de ellos (haga uso de las herramientas antes citadas para la inserción rápida de comandos, así como de las pulsaciones rápidas indicadas):

```
echo "Hola"
cat /etc/passwd
clear
cd /tmp/
pwd
cd $HOME
pwd
cd /tmp/; pwd
cat /etc/passwd
clear
cd /tmp/
pwd
cd $HOME
pwd
cd /tmp/; pwd
cd; pwd
pwd
ls
ls –l
ls –la
man man
man date
pwd
mkdir -p /tmp/dir/subdir
echo "hola" > /tmp/dir/subdir/fichero
cat /etc/fstab > /tmp/dir/subdir/fichero2
cp /etc/passwd /tmp/dir/usu
mv /tmp/dir/usu /tmp/dir/usu2
cat /tmp/dir/usu2
mkdir /tmp/dir2
cp -Rf /tmp/dir /tmp/dir2
ls /tmp/dir
ls /tmp/dir2
```

```
rm -Rf /tmp/dir /tmp/dir2
cd /home/dit
pwd
ls –la | more
cd
pwd
```

TAREAS

2º Escriba el prefijo "ba" y haga uso del autocompletado (pulse "Tabulación") para obtener los distintos comandos que poseen ese prefijo. Tras ello, haga uso del autocompletado para escribir el comando "`cat /home/dit/.bashrc`".

3º Cree el directorio "/tmp/pruebas" y ubíquese dentro de él.

4º Usando el comando "echo", cree en el directorio /tmp/pruebas el fichero "cabecera" que contenga la cadena "Lista".

5º Usando el comando "ls", cree en el directorio /tmp/pruebas el fichero "listado", de modo que su contenido incluya el listado de carpetas y ficheros que directamente (sin recursividad) se encuentran en el directorio raíz "/".

6º Añada al fichero "cabecera" el contenido del fichero "listado".

7º Usando el comando "rm", borre recursivamente y sin confirmación el directorio "/tmp/pruebas", incluyendo todos los ficheros que contiene.

8º Cambie al usuario "root" con el comando "`su -`" y ejecute el comando "`ifconfig –a`". Vuelva al usuario anterior "dit" con el comando "`exit`".

9º Cierre la consola de comandos con el comando "`exit`".

2.3 Distribución de consolas en modo texto y gráfico (25 minutos)

Las consolas virtuales proporcionan varias “pantallas” a través de las cuales el usuario puede entrar al sistema y ejecutar programas (en cada momento sólo se muestra una de estas “pantallas”, no pueden mostrarse varias simultáneamente). En los sistemas Linux habituales se crean varias consolas virtuales, cada una de las cuales tiene asignada una tecla de función [Fz] (F1, F2, etc.):

- "6" consolas virtuales en modo texto: con las teclas de función respectivas, habitualmente F1, ..., F6. La entrada a cada consola es controlada mediante un "login/password" (tarea realizada por el programa "/bin/login", el cual es ejecutado automáticamente por el sistema para controlar el acceso a cada una de estas consolas). Una vez identificado con un usuario determinado, el sistema cargará el intérprete de comandos configurado para dicho usuario (e.g. "/bin/bash"), disponiendo ya de una línea para la introducción de comandos (así pues, todas estas consolas corresponden a"consolas de comandos").

- "1" consola virtual, al menos, en modo gráfico (en general, cada consola es responsabilidad de un servidor X): aunque depende del sistema, suele tener asignada la tecla de función “F7” o “F8” (según si se ha reservado una consola para la salida de los mensajes durante el proceso de arranque del sistema); podrían existir más consolas gráficas, asociándoseles las siguientes teclas de función F8 o F9, F10, ... La entrada a una sesión gráfica suele ser controlada por un "gestor de sesiones", igualmente mediante un "login/password". Una vez identificado, el gestor de sesiones cargará las distintas aplicaciones X que componen el escritorio. Desde el modo gráfico pueden abrirse "emuladores de terminal" (programas tales como "konsole, rxvt, kvt, xterm, nxterm o eterm") que ofrecen consolas de comandos; al entrar en cualquiera de ellas, se cargará el intérprete de comandos configurado para el usuario que ha abierto la sesión gráfica, disponiendo ya de una línea para la introducción de comandos.

Para conmutar entre las diferentes consolas, saltando a la consola virtual asociada a la tecla de función "[Fz]", se utiliza la combinación de teclas "**[Control]-[Alt]-[Fz]**" (para saltar desde consolas en modo texto basta usar "[Alt]-[Fz]").

TAREAS

Conmute entre las distintas consolas y averigüe cuál es la tecla de función Fx en que se encuentra su consola gráfica. Tras ello, abra una sesión en una consola en modo texto y otra en la consola gráfica. Compruebe que el sistema es multitarea ejecutando distintos comandos en cada consola.

Puede determinar qué aplicaciones se ejecutan en cada consola virtual usando el comando (si lo necesita, consulte la ayuda del comando "man ps"):

```
ps aux
```

La notación empleada por dicho comando (y en general dentro del sistema Linux) para identificar las consolas virtuales desde las que se ha invocado cada proceso es la siguiente:

Identificador	Tipo de consola de comandos
`ttyX`	Consolas virtuales en modo texto (consolas de comandos) locales (Ctrl-Alt-F1/F6)
`pts/X`	Consolas de comandos bajo modo gráfico (emuladores de terminal) o sesiones remotas (SSH, ...)
`Carácter "?"`	Procesos invocados desde una consola virtual en modo gráfico (sin usar consolas de comandos)

Para obtener el identificador de la consola en la que actualmente está trabajando basta ejecutar el comando "`tty`".

Puede obtener las consolas virtuales y consolas de comandos abiertas, y/o el usuario que las abrió mediante cualquiera de los siguientes los comandos: "`who, w, finger, users`". Tenga en cuenta que cada consola pertenece al usuario que introduce el "login/pasword" al abrir la consola; cuando se hace uso del comando "su" o "su -", se está cambiando de usuario, pero no abriendo una nueva consola o sesión.

TAREAS

Ejecute el comando "`ps aux`" y analice la información que se le muestra sobre las consolas.

Utilice los comandos "`who, w, finger, users`". Consute el manual de cada uno para determinar la diferencia entre ellos

Es posible restringir a qué usuarios, de todos los definidos en el sistema, se les permite el acceso local a cada consola. Para ello, existe por ejemplo el fichero "/etc/securetty"[6]. En cuanto a las sesiones remotas, el control de acceso dependerá del servidor. Por ejemplo, el servidor "telnet" aplica el fichero "/etc/securetty", mientras que el servidor SSH controla directamente si el usuario "root" tiene permiso de acceso.

2.4 Organización del Sistema de Archivos de Linux (15 minutos)

A continuación se comenta de forma muy básica la estructura y organización de algunos (ver otros en [7]) de los directorios del sistema de archivos que utiliza el sistema operativo Linux (sigue el estándar FHS -Filesystem Hierarchy Standard-, que puede consultar en la ayuda "`man hier`"):

Directorio	Contenido
`/`	[Obligatorio] Directorio raíz. Es la base de la jerarquía del árbol de directorios.
`/bin`	[Obligatorio] Contiene ejecutables que son necesarios en modo monousuario sin tener en cuenta los privilegios de ejecución que tengan asignados. Este directorio no está pensado para tener subdirectorios, de tal forma que, aquellos comandos que estén pensados para que sean ejecutados por los usuarios, deberán estar ubicados en el directorio "`/usr/bin`".
`/dev`	[Obligatorio] Contiene archivos que representan a los dispositivos del equipo (de tipo carácter y bloque).
`/etc`	[Obligatorio] Contiene archivos de configuración para la máquina. Este directorio está ramificado en subdirectorios para incluir de forma estructurada los archivos de configuración de determinadas aplicaciones.
`/home`	[Opcional] Contiene los subdirectorios propios de cada uno de los usuarios individuales. Algunos administradores prefieren ubicar los anteriores subdirectorios en otras carpetas, como por ejemplo "`/var/users`".

Directorio	Contenido
`/media`	[Opcional] Suele contener acceso a dispositivos de almacenamiento (discos duros, USB, …).
`/proc`	[Opcional] Contiene un sistema de archivos virtual, es decir: que no corresponde a ningún dispositivo físico. Dicho sistema de archivos contiene información del kernel y procesos que actualmente se estén ejecutando en esa máquina.
`/sbin`	[Obligatorio] Contiene ejecutables orientados a la administración y que son de uso exclusivo para el administrador del sistema.
`/tmp`	[Obligatorio] Se usa cuando una aplicación necesita escribir un archivo que será borrado cuando dicho programa termine su ejecución. Normalmente se limpia cada vez que se reinicia el sistema.
`/var`	[Obligatorio] Contiene archivos de datos variables como por ejemplo archivos de log y ciertos archivos de datos específicos de los procesos.

TAREAS

Como ejercicio muévase a lo largo de la jerarquía de directorios comentada, en modo gráfico o consola de comandos, e inspeccione el contenido de dichos directorios de forma que pueda contrastar la información suministrada anteriormente.

3. Entorno gráfico: Sistema X-Windows (20 minutos)

Los entornos gráficos ofrecen al usuario un entorno más amigable que el funcionamiento en modo texto, haciendo uso de iconos, barras de herramientas, interfaces gráficas y un uso intensivo del ratón. Cada entorno gráfico se caracteriza por su aspecto y comportamiento particulares aunque todos suelen tener características en común. Linux ofrece un entorno gráfico basado en el sistema X, que veremos a continuación. Esta práctica asume que usted está habituado al uso de entornos gráficos, al menos a nivel de usuario.

3.1 Funcionamiento del Sistema X-Windows (5 minutos)

A diferencia de otros sistemas operativos como MS Windows, el entorno gráfico de Linux se ofrece mediante una comunicación entre procesos basada en el modelo cliente-servidor mediante el protocolo de aplicación X11 (sobre TCP o UDP y puerto estándar 6000):

a) Servidor X (por ejemplo, Xorg o Xfree86): ofrece el servicio de representar en pantalla la información (así como gestionar la entrada de información mediante teclado y ratón) de los clientes X que los soliciten.

b) Clientes X: cualquier aplicación gráfica (e.g. nedit, firefox, chromium, xclock, ...) que solicita al servidor X que la represente en pantalla.

Al conjunto de clientes X que ofrecen una solución completa de interfaz gráfica de usuario se le denomina Entorno de escritorio. Linux dispone de múltiples entornos de escritorio, tales como XFCE (el disponible en su equipo de trabajo), KDE, Gnome, ...

Si en algún momento su sistema X presentase un comportamiento anómalo, resulta posible resetearlo usando la pulsación especial "`[Alt Gr] - [Impr Pant] - K`".

3.2 Acceso remoto al servidor X (15 minutos)

Bajo un uso local, la funcionalidad ofrecida por el sistema X Window es completamente análoga a la de cualquier otro sistema operativo como MS Windows. Una de las principales ventajas del funcionamiento bajo un modelo cliente/servidor del sistema X se obtiene cuando el cliente y servidor X se encuentran en máquinas remotas. Este apartado intentará hacer uso de esta capacidad, ejecutando aplicaciones en un sistema remoto y obteniendo la salida en su equipo local:

TAREAS

Desde una sesión gráfica (tipo de sesión Xfce o "Default Xsession"), realice las siguientes operaciones **con el usuario "dit"**:

1º Usando el comando "`/sbin/ifconfig`", averigüe la dirección IP de su equipo y la dirección IP de otro equipo de su sala que esté encendido (si alguno de los PCs adyacentes al que está empleando se encuentra disponible, use dicho PC; en otro caso, solicítele a su compañero que le indique la dirección IP de su equipo y que active su servidor SSH, como se indica al principio de la práctica, para poder acceder remotamente a su equipo).

2º Inicie una sesión remota SSH hacia ese segundo PC del que ha obtenido su dirección IP (el parámetro "`-X`" hace que el cliente SSH solicite al servidor SSH que le canalice toda la salida gráfica de las aplicaciones ejecutadas hacia la máquina del cliente, encapsulando el protocolo X11 sobre el túnel SSH):

```
ssh  -X  dit@IP_otro_PC
```

3º Sobre esa sesión remota, ejecute el comando "`/sbin/ifconfig`", debiendo comprobar cómo obtiene la dirección IP de la máquina remota. Tras ello, ejecute una aplicación gráfica, por ejemplo "`nedit &`", comprobando cómo la ventana gráfica de la misma es representada en su ordenador local, aunque está siendo ejecutada en la máquina remota. Puede comprobar esto, por ejemplo, guardando un fichero desde esa aplicación gráfica y comprobando con la consola de comandos que el fichero se ha guardado en el equipo remoto, y no en su equipo local.

4º Cierre la sesión remota ejecutando el comando "`exit`" (tras este paso, el equipo remoto puede desactivar el servidor SSH).

4. Funcionamiento en modo texto: comandos de administración del intérprete de comandos (110 minutos)

Cada consola de comandos (o interfaz de línea de comando CLI) ofrece acceso a un intérprete de comandos. El interprete de comandos o shell es un programa que recoge los comandos que un usuario introduce por el teclado y los envía al sistema operativo para que los ejecute. En Linux existen múltiples intérpretes de comandos, tales como bash, sh, ksh, tcsh, csh, sh, zsh, ... Los shells disponibles en su sistema se encuentran recogidos en el fichero "`/etc/shells`".

TAREAS Compruebe cuáles de los intérpretes de comandos citados existen en su equipo y consulte su ayuda.

Su sistema está configurado para que al abrir una consola de comandos con los usuarios "dit" o "root", se cargue el intérprete de comandos "bash" (Bourne Again SHell), el cual es el intérprete usado por defecto para los usuarios en la mayoría de los sistemas Linux. "bash" ofrece una potente consola de comandos (puede consultar "man bash" para ver las múltiples opciones que ofrece).

El funcionamiento de cualquier intérprete de comandos está condicionado por el valor que tienen sus denominadas "variables de shell" (variables de entorno que permiten modificar el comportamiento del shell), tales como "HOME", "PATH", "LANG", "PWD", ... (las variables de entorno también son sensibles a mayúsculas y minúsculas). Para:

- Imprimir el valor de una variable se usa: `echo $VAR`
- Modificar el valor de una variable en la sesión actual puede usarse: `VAR=nuevo_valor`

Entre todas ellas, resulta de especial relevancia la variable de shell PATH, que contiene los directorios (separados por ":") donde el shell busca los comandos cuando se invocan sólo a partir de su nombre (sin indicar explícitamente la ruta completa del comando), es decir, el shell usa la variable PATH cuando el nombre del comando a ejecutar no contiene el carácter "/". Puede ver el valor de dicha variable con el comando:

```
echo $PATH
```

Para modificar su valor en la sesión actual del intérprete de comandos se puede usar el comando:

```
PATH=$PATH:/directorio_nuevo
```

Es frecuente que el directorio actual "." no se encuentre en la variable PATH, en cuyo caso para ejecutar un comando ubicado en el directorio actual habría que indicar su ubicación explícitamente, por ejemplo de forma relativa mediante:

```
./comando
```

Por otro lado, el intérprete permite definir alias de comandos mediante el comando "alias" (man bash). La sintaxis es:

```
alias  nombre_alias="comando"
```

Los alias tienen prevalencia sobre los comandos (si existen un alias y un comando con igual nombre, el shell usa el alias). Los alias definidos con el comando "alias" están disponibles en el proceso shell actual. Pueden obtenerse los alias disponible en dicho proceso con el comando "`alias`". El uso de los alias se desaconseja, recomendándose en su lugar el empleo de funciones.

TAREAS

Realice las siguientes operaciones relativas a las variables de entorno:

1º Abra una consola de comandos y consulte en la ayuda del intérprete Bash las distintas variables de entorno que utiliza.

2º Consulte el valor actual de todas las variables de entorno (de shell y usuario) definidas en el intérprete de comandos actual mediante el comando: "`set`" (sin argumentos).

3º Ejecute los siguientes comandos para modificar el valor de la variable "HOME" y analice cómo afecta al resultado del comando "`cd ~; pwd`". Por ejemplo, ejecute:

```
echo $HOME
cd ~
pwd
HOME=/usr
cd ~
pwd
```

4º Modifique el valor de la variable "LANG" y analice cómo afecta al idioma de la ayuda que muestra el comando "`man diff`" (puede obtener los posibles valores de la variable "LANG" usando el comando "`cat /usr/share/i18n/SUPPORTED`", como se indica en el manual "`man locale`"). Por ejemplo, ejecute:

```
echo $LANG
man diff
LANG="en"
man diff
LANG="es-ES.UTF-8"
man diff
```

TAREAS

5º Modifique el valor de la variable "PATH" y analice cómo afecta al comportamiento del intérprete de comandos. Por ejemplo, ejecute estos comandos y analice el resultado (el operador ". ", equivalente al comando "source" en Bash, interpreta el script indicado con el proceso shell actual):

```
echo $PATH
mkdir /tmp/dir
cp /bin/ls /tmp/dir/
ls /tmp/dir/
ls /
PATH=""
ls /
cd /tmp/dir/
ls /
PATH="."
ls /
cd /tmp
ls /
PATH=$PATH:/tmp/dir/
ls /
. /etc/profile
```

6º Ejecute el comando "alias" para conocer cuales son los alias definidos en su intérprete actual. Haga uso de alguno de ellos.

7º Defina la palabra "usuarios" como alias del comando "cat /etc/passwd", ejecutando:

```
alias  usuarios="cat /etc/passwd"
```

Compruebe cómo dicho alias aparece al ejecutar el comando "alias", y que efectivamente funciona: si usa el alias ejecutando la orden "usuarios", obtendrá el mismo resultado que si ejecuta el comando "cat /etc/passwd".

Las variables de shell son configuradas para cada usuario en múltiples ficheros durante el inicio de sesión (puede consultar la referencia [1]). A continuación se analizan los principales comandos destinados a la administración básica de un equipo dotado del sistema operativo Linux (exceptuando los comandos de red, que serán objeto de la otra práctica):

4.1 Comandos para la gestión de Cuentas de Usuario (30 minutos)

En Linux, los usuarios existentes en el sistema están definidos en el fichero "/etc/passwd", en el que cada línea pertenece a un usuario y presenta los siguientes parámetros separados por ":" (si el campo "shell" está vacío, se asume "/bin/sh"):

/etc/passwd

```
login:clave:uid:gid:descripción:carpeta_personal:shell
```

Por ejemplo, el contenido del fichero podría ser el siguiente:

/etc/passwd

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
postgres:x:2:2:sql::/dev/null:/sbin/nologin
...
dit:x:1000:1000:Alumno FAST,s2,s3,s4:/home/dit:/bin/bash
```

El valor e interpretación de los subcampos del campo “descripción” es realizada por algunos comandos como por ejemplo el comando "finger" (por ejemplo, ejecute el comando “`finger dit`”).

En este fichero se definen todos los usuarios del sistema, los cuales pueden clasificarse en dos categorías conforme a su utilidad:

- Usuarios “humanos” o reales: están pensados para que los usuarios físicos (personas) deseados puedan acceder y usar el sistema. En el ejemplo, serían los usuarios “root” y “dit”.

- Usuarios virtuales: son usuarios empleados únicamente por procesos internos del sistema con objeto de asegurar un control adecuado de los permisos de acceso. Dado que estos usuarios no requieren autenticarse (normalmente un proceso perteneciente al “root” se cambia a uno de estos usuarios), en el fichero “/etc/passwd” se les indica como shell “/sbin/nologin”, de modo que ninguno de estos usuarios podrá entrar al sistema autenticándose con un login/clave. En el ejemplo, serían los usuarios “bin” y “postgres”.

El parámetro numérico "uid" corresponde al identificador (único) del usuario. El parámetro numérico "gid" corresponde al identificador (único) del grupo principal al que pertenece el usuario (el usuario puede pertenecer a otros grupos -secundarios- como se explica más adelante).

Para permitir que las aplicaciones puedan conocer las características de cada usuario, es necesario que el fichero "/etc/passwd" tenga permisos de lectura para todos los usuarios del sistema. Por ejemplo:

- Cada fichero tiene asociado un “uid” y un “gid”, correpondientes a los identificadores del usuario y grupo al que pertenece. El comando “ls” requiere poder leer el fichero “/etc/passwd”, para poder mostrar el nombre del usuario propietario, y el fichero “/etc/group” (como se explica más adelante) para poder mostrar el nombre del grupo propietario, en lugar de mostrar el uid y gid de cada fichero.

- Para mostrar la información de un usuario, el comando “finger” requiere acceder al fichero “/etc/passwd”.

El campo “login” presenta una serie de restricciones (puede consultar la referencia [2]).

El segundo parámetro (“clave”) correspondería a la clave del usuario en formato encriptado (si este parámetro estuviese vacío, se permitiría el acceso del usuario sin clave). Dado que, como se ha justificado, es necesario que el fichero “/etc/passwd” pueda ser leído por cualquier usuario, la inserción de las claves en este fichero sería insegura. Por dicho motivo, las claves de los usuarios reales se suelen insertar en el fichero "/etc/shadow" (usándose el valor “x” para el campo “clave” de “/etc/passwd”), el cual sólo puede ser leído por el usuario root (los distintos programas encargados del control de acceso al sistema, tales como “/bin/login”, “/bin/su”, ..., son ejecutados con permisos de usuario “root”, lo que les permite por tanto leer este fichero). Cada línea del fichero “/etc/shadow” corresponde a un usuario y presenta el siguiente formato:

/etc/shadow

```
login:clave_encriptada:f1:f2:f3:f4:f5:f6:reservado_sin_uso
```

Si en el parámetro “clave_encriptada” aparece un carácter no válido de “clave encriptada”[3] tal como “*” o “!”, no se permite el login del usuario. Una clave encriptada que comienza por “!” suele indicar que el usuario está bloqueado (lo que aparece después de “!” es la clave encriptada del usuario, pero se ha añadido dicho carácter para impedirle temporalmente su acceso). Los usuarios con clave "!!" suelen ser usuarios "de sistema" (usados desde el superusuario para arrancar procesos con ese usuario). Si este campo está vacío, el usuario no tendrá clave (puede consultar la referencia [3]).

Como algoritmo de encriptación pueden usarse varias opciones (DES, MD5, SHA1, ... véase “man crypt”), siendo el algoritmo “DES” la opción por defecto en su sistema. En cualquiera de los casos, el algotirmo de encriptación siempre será una función de un sólo sentido, de modo que a partir de la clave encriptada no puede obtenerse la clave en texto plano. Cuando un usuario introduce su login y clave para abrir una sesión, el sistema encripta la clave introducida por el usuario. Tras ello, compara la cadena resultante de la encriptación con la existente para ese usuario en el fichero "/etc/shadow", sólo permitiendo el acceso si coinciden. Este mecanismo proporciona un control de acceso seguro, garantizando al mismo tiempo que ningún usuario (incluyendo el "root") pueda obtener la clave en texto plano de los demás usuarios del sistema.

Además de los usuarios, el sistema de permisos de Linux también define grupos. Al igual que con los usuarios, cada grupo queda identificado unívocamente con un “gid”. Un grupo no es más que un nombre bajo el cual se referencia a múltiples usuarios. La utilidad de los grupos es asignar permisos a un conjunto de usuarios (distintos al usuario propietario) en una sola operación. Los grupos están definidos en el fichero "/etc/group", en el que cada línea define un grupo y tiene el siguiente formato (los miembros de un determinado grupo serán los usuarios indicados en este fichero “/etc/group”, más los usuarios que en el fichero “/etc/passwd” usan este grupo como grupo principal):

[3] En general, la clave que escriba el usuario puede usar cualquier carácter ASCII, recomendablemente imprimible. La encriptación de esta clave es lo que se guarda en “/etc/shadow” (ésta es la que no puede contener “!” o “*”).

/etc/group

```
nombre_grupo:x:gid:user1,user2,...,userN
```

Esto permite que un usuario pueda pertenecer a varios grupos (secundarios o suplementarios), además del especificado en el fichero /etc/passwd (grupo principal o “grupo de login inicial”).

Al igual que para los usuarios, es posible definir una clave para los grupos, contenidas encriptadas en el fichero "/etc/gshadow", aunque no suele ser utilizada.

El comando “`ls -l -n`” mostraría los ficheros indicando el UID/GID numéricos (en lugar de traducirlos a su valor literal consultando “/etc/passwd” y "/etc/group"). Usualmente "ls" (al definirse como alias de “ls --color=auto”) representa en verde los ficheros que tienen permisos de ejecución (lo cual es modificable con el comando “dircolors”, que cambia la variable de entorno “LS_COLORS”, puede consultar la referencia [4]).

Resulta importante que diferencie entre los usuarios definidos existentes en el sistema (los recogidos en el fichero "/etc/passwd"), de las carpetas del directorio "/home" (que pueden ser carpetas personales o no, y no todos los usuarios tienen que tener aquí su carpeta personal) y de los usuarios con sesiones abiertas ya analizados anteriormente (comandos "`who, w, finger, users`"). Linux dispone de múltiples comandos para la gestión de los usuarios y grupos del sistema, entre los que destacan:

Básicos	`su, sudo, useradd, userdel, passwd, usermod, groupadd, groupdel, id, groups, finger, users, who, w, last`
Para Profundizar	`gksu, whoami, login, newgrp, newusers, chsh, chage, chfn, groupmod, gpasswd, chpasswd, mkpasswd, pwck, grpck, umask, quota, quotacheck, edquota, quotaon, quota_nld, repquota, warnquota`

PARA PROFUNDIZAR...

La información acompañada del título "**Para Profundizar...**" se incluye para aquellos alumnos que deseen ahondar en la práctica, pero su conocimiento no se exigirá en las evaluaciones de la Asignatura.

Por su frecuente uso, deben resaltarse especialmente los siguientes comandos:

Comando	Función
`useradd [-g grupo] user`	Crea el usuario "user" con distintos parámetros (POSIX)[4].

[4] "`useradd usuario`" crea el usuario sin clave ("`adduser usuario`" sí solicita la clave), por lo que no puede usarse para abrir una sesión (su campo clave en /etc/shadow vale "!"). Para que el usuario pueda ser empleado para abrir una consola de comandos o gráfica, se le debe asignar una clave mediante "`passwd usuario`".

Comando	Función
`userdel [-r] user`	Borra el usuario (su entrada en /etc/passwd). También borra la carpeta personal si se indica "-r" Tras borrar un usuario, su UID está libre para un nuevo usuario (heredaría los ficheros, aun existentes en el sistema, del usuario borrado).
`passwd [user]`	Cambiar la contraseña de un usuario.
`usermod [-d dir | -u UID | -g grupo | -l login | [-a] -G grupo1,grupo2,... | -s shell | ... ] user`	Cambia la configuración (carpeta personal, UID, grupo principal, nombre de usuario, grupos secundarios, shell, ...) del usuario
`groupadd grupo`	Crea un grupo con distintos parámetros (POSIX).
`groupdel grupo`	Elimina el grupo.
`groups [user]`	Muestra los grupos a los que pertenece un usuario.
`finger user`	Información detallada de un determinado usuario obtenida de "/etc/passwd".
`finger, users, who, w`	Muestran los usuarios a los que pertenece cada sesión de shell (local o remota) actualmente abierta en el sistema
`last`	Muestra un listado de las últimas conexiones de shell realizadas en el sistema

Asimismo, merecen una explicación adicional los siguientes comandos:

- "`su [-] [user]`": ejecuta un intérprete de comandos con el usuario "user" indicado, con el "uid" y "gid" de su grupo principal; si el usuario al que se desea cambiar es "root", puede omitirse el "user", usando simplemente "`su`"; para volver al usuario anterior al cambio, debe ejecutar el comando "`exit`" (**no use nuevamente "su" dado que ello provoca una anidación de sesiones** que puede llevar a una situación inestable del funcionamiento de la consola). Entre las dos opciones "`su user`" y "`su - user`", la segunda asegura que todas las variables de shell se configuren a las esperadas para el usuario "`user`", tal como se esperaría tras un inicio de sesión normal ("/bin/login").

- "`sudo comando`": permite que un usuario pueda ejecutar ciertos comandos (e.g. mount) como otro usuario (incluso según la según IP), según se configure en "`/etc/sudoers`". El contenido básico de dicho fichero suele ser "`%sudo ALL=(ALL) ALL`", cuyo efecto es que cuando cualquier usuario del grupo "sudo" ejecute "`sudo comando`", se intentará invocar dicho comando como superusuario. No obstante, debe tenerse en cuenta que "`sudoers`" admite muchas más configuraciones (`man sudoers`).

TAREAS

1º Consulte el manual de ayuda de los distintos comandos mencionados para conocer su funcionalidad.

2º Usando el usuario root, vea el contenido de los ficheros "`/etc/passwd`", "`/etc/shadow`", "`/etc/group`" y "`/etc/gshadow`".

3º Editando manualmente (con un editor de textos, e.g. emacs o nedit) los ficheros anteriores (salvo la generación de la clave, que deberá hacerla mediante el comando "passwd"), cree el usuario "user1" (con clave de acceso "12345", carpeta personal "/home/user1", y shell "/bin/bash") perteneciente a "group1" como grupo principal.

4º Usando los comandos apropiados (sin editores de texto), cree el usuario "user2" (con clave de acceso "12345", carpeta personal "/home/user2", y shell "/bin/bash") también perteneciente a "group1" como grupo principal.

5º Usando los comandos adecuados, cree el grupo "group2" al que pertenezca el usuario "user1".

6º Usando el comando adecuado, cambie la definición de "user2" para que su grupo principal sea "group2".

7º Usando los comandos adecuados, modifique "group2" para que contenga los usuarios "dit", "user1" y "user2".

8º Usando el comando adecuado, modifique el shell del usuario "user1" para que sea "/bin/sh".

9º Usando el comando adecuado, obtenga los grupos a los que pertenece el usuario "dit".

10º Abra una sesión con el usuario "user1" y cree un fichero en la carpeta "/tmp/". Por ejemplo, con el comando:

```
echo "hola" > /tmp/fichero
```

Use el comando "ls" para ver los permisos de dicho fichero.

11º Usando los comandos adecuados, elimine los usuarios "user1" y "user2", incluyendo sus carpetas personales. Elimine también los grupos "group1" y "group2".

12º Vuelva a usar el comando "`ls`" para ver los permisos del archivo "`/tmp/fichero`" y justifique a que se debe el valor indicado como usuario y grupo propietarios.

13º Ejecute los comandos "`ls -l /tmp`" y "`ls -l -n /tmp`" y analice la diferencia.

4.2 Comandos relacionados con el Sistema de Archivos (30 minutos)

Linux dispone de múltiples comandos relacionados con el uso del sistema de archivos y su control de permisos, entre los que destacan:

Básicos	`ln, chown, chmod, find, whereis, which, lsattr, chattr, dirname, basename, rename`
Para Profundizar	`chgrp, chroot, locate`[5]`, updatedb, stat`

Por su elevado uso, deben destacarse los siguientes comandos:

Comando	Función
`ln [-s] fichero enlace1 enlace2`	Crea enlaces duros (sin "-s") o simbólicos (con "-s") al fichero indicado. La opción "-f" permite *modificar* el destino al que apunta un enlace ya existente.
`chmod [-R] xxx dir/file`	Asigna los permisos "xxx" (en formato numérico o literal "rwx") al fichero o directorio indicado
`chown [-R] user[:group] dir/file`	Asigna el usuario (y grupo) propietarios al fichero o directorio indicado. Para cambiar los propietarios de un fichero es necesario disponer de permisos de superusuario.

`find [ruta] [-name patron]`
`[-path patron]`
`[-regex reg_exp]`
`[!] [-type f/d/l]`
`[-perm -u/g/o=rwx]`
`[-inum inode]`
`[-size nnn] [...]`
	Buscar ficheros ("ruta=." por omisión). Por ejemplo "find /dir/ -name file" busca en la carpeta "dir" un fichero que tenga el nombre "file" ("-name" sólo nombre, "path" ruta y nombre, pudiendo ser relativa a "ruta"). Por omisión, "-regex" busca ficheros por nombre (puede tener ruta) usando la expresión regular "reg_exp" como básica BRE (modificable con "-regextype tipo"). Ejemplos adicionales en [8].

[5] El comando "`locate fichero`" busca un fichero por su nombre en la base de datos "`locatedb`", no realiza una búsqueda interactiva.

Comando	Función
`whereis comando`	Indica la ubicación de "comando" (binarios, fuentes y man) buscándolo en un conjunto de carpetas predefinidas.
`which comando`	Indica la ubicación de "comando" (sólo) buscándolo en los directorios de la variable PATH (si hay varias coincidencias, sólo muestra la primera).
`lsattr, chattr`	Permiten ver y asignar "atributos" a los ficheros. Por ejemplo, con "chattr +i fichero", se protege al fichero indicado para que no pueda ser eliminado mientras no se le quite antes el atributo (con "chattr -i fichero").
`dirname /dir/fichero` `basename /dir/fichero`	Devuelven sólo el nombre (fichero) o sólo la ruta de directorios (/dir) de la ruta indicada, respectivamente

Respecto a los enlaces, resultan relevantes las siguientes aclaraciones:

- Un enlace simbólico no tiene permisos ("`ls`" los muestra con todos los permisos activos). La aplicación del comando "`chmod`" sobre un enlace simbólico cambiará los permisos del fichero real apuntado por el enlace. Sin embargo, el enlace simbólico sí pertenece a un determinado usuario y grupo (inicialmente, al usuario que lo crea). La aplicación del comando "chown" sobre un enlace simbólico cambiará los propietarios del fichero apuntado, sin afectar al enlace; por el contrario, la aplicación del comando "chown -h" sobre un enlace simbólico cambiará sus propietarios sin afectar al fichero apuntado.

- Un enlace duro usa el mismo i-nodo (entrada al sistema ficheros) que el fichero apuntado. Para localizar los distintos enlaces duros que apuntan a un mismo fichero puede usarse el comando "`find /ruta -inum número_inodo -print`". Para visualizar el número de i-nodo asociado a cada fichero puede usarse el comando "ls" con el parámetro "-i", e.g. "`ls -i /dir`"..

- Por defecto, los sistemas Linux no permiten la creación de enlaces duros a directorios para evitar bucles infinitos. Por su parte, los enlaces simbólicos sí pueden apuntar a directorios al ser ficheros especiales que resultan detectables, pudiendo "no seguirse" para romper los bucles (por ejemplo, el comando "`find`" no sigue los enlaces simbólicos por omisión; para seguirlos hay que indicarlo expresamente con "`find -follow ...`").

Por último, recuerde que el efecto de los permisos de lectura (r), escritura (w) o ejecución (x) sobre un fichero varía según se trate de un fichero regular o de un fichero tipo directorio.

TAREAS

1º Consulte el manual de ayuda de los distintos comandos mencionados para conocer su funcionalidad.

2º Trabajando como superusuario, cree el fichero "`/tmp/archivo`" con el contenido "texto".

3º Use el comando adecuado para que el usuario y grupo propietarios de "`/tmp/archivo`" pasen a ser "dit".

4º Abra una sesión con el usuario "dit" e intente cambiar el usuario propietario de "`/tmp/archivo`" para que vuelva a ser "root". Analice qué sucede.

5º Con el usuario "dit", intente cambiar cambiar los permisos de "`/tmp/archivo`" a "644". Analice si el sistema se lo permite.

6º Vuelva a la sesión del superusuario e intente cambiar otra vez el grupo propietario de "`/tmp/archivo`" para que vuelva a ser "root", comprobando que ahora sí es posible.

7º Abra de nuevo una sesión con el usuario "dit". Cree la carpeta "`/tmp/dir/`" y haga una copia de "`/tmp/archivo`" con el nombre "`/tmp/dir/otro`". Analice cuáles eran los permisos y propietarios del fichero "`/tmp/archivo`" y cuales son los del resultado de la copia "`/tmp/dir/otro`".

8º Con el usuario "dit" y usando el comando "`echo`", cree el fichero "`/tmp/dir/otro2`" con el contenido "linea". Mire cuales son los permisos del fichero "`/tmp/dir/otro2`" y analice por qué se ha creado con dichos permisos (a esta tarea le puede ayudar consulte "`man bash`" y busque el comando interno "`umask`"; ver [10] en el Anexo).

9º Trabajando como superusuario, cree un nuevo usuario y grupo denominados "dit2". Modifique el grupo (no el usuario) propietario del directorio "`/tmp/dir/`" y del fichero "`/tmp/dir/otro2`" para que ahora sea "dit2".

10º Trabajando como superusuario, modifique los permisos del directorio "`/tmp/dir/`" para que sean "700". Abra una sesión con el usuario "dit" y vea el contenido del archivo "`/tmp/dir/otro2`". Abra ahora una sesión con el usuario "dit2" y vuelva a intentar ver el contenido del archivo "`/tmp/dir/otro2`", comprobando que ahora no es posible, a pesar de que dicho usuario posee (a través del grupo propietario) permisos de lectura sobre el archivo. Justifique a que se debe esto.

TAREAS

11º Con el usuario "dit", cree la carpeta "`/tmp/dit/`" con permisos "700", y dentro de ella cree el archivo "`/tmp/dit/file`" con permisos "600". Realice las siguientes operaciones y analice el motivo de los resultados:

- Compruebe cómo funcionan correctamente: "`ls /tmp/dit/`", "`cat /tmp/dit/file`" y "`echo nuevo > /tmp/dit/nuevo`".
- Asigne a "`/tmp/dit/`" permisos "600" y compruebe que funciona "`ls /tmp/dit/ 2> /dev/null`", pero no "`echo nuevo > /tmp/dit/nuevo`" ni "`cat /tmp/dit/file`" .
- Asigne a "`/tmp/dit/`" permisos "300" y compruebe que funcionan "`cat /tmp/dit/file`" y "`echo nuevo > /tmp/dit/nuevo`", pero no "`ls /tmp/dit/`" .
- Asigne a "`/tmp/dit/`" permisos "500" y compruebe que funcionan "`ls /tmp/dit/`", "`cat /tmp/dit/file`" y "`echo nuevo > /tmp/dit/nuevo`" (éste último porque "nuevo" ya existía), pero no funciona "`echo nuevo > /tmp/dit/otro`".
- Asigne a "`/tmp/dit/`" permisos "300" y a "`/tmp/dit/file`" permisos "000". Compruebe cómo, a pesar de esos permisos, sí funciona "`rm /tmp/dit/file`".
- Asigne a "`/tmp/dit/`" permisos "700" y cree dentro el directorio "`/tmp/dit/sdit/`" con permisos 700 y el fichero "`/tmp/dit/sdit/sfile`" con permisos "600". Compruebe que "`cat /tmp/dit/sdit/sfile`" funciona. Cambie los permisos de "`/tmp/dit/`" a "600" y compruebe que "`cat /tmp/dit/sdit/sfile`" ya no funciona.

12º Modifique el grupo "dit2" para que el usuario "dit" sea miembro del mismo. Cree un tercer usuario "dit3".

13º Trabajando como superusuario,vuelva a modificar los permisos del directorio "`/tmp/dir/`" para que sean "777", y los del archivo "`/tmp/dir/otro2`" para que sean "046" (compruebe que ambos pertenecen al usuario "dit" y al grupo "dit2"). Abra una sesión con los siguientes usuarios y justifique los siguientes resultados:

- "dit3": compruebe que puede ver y modificar el archivo "`/tmp/dir/otro2`".
- "dit2": compruebe que puede ver pero no modificar el archivo "`/tmp/dir/otro2`".

TAREAS

- "dit": compruebe que no puede ver ni modificar el archivo "`/tmp/dir/otro2`".

Justifique estos resultados.

14º Trabajando como superusuario, vuelva a modificar los permisos del archivo "`/tmp/dir/otro2`" para que sean "000". Intente ver su contenido con los usuarios "dit" y "root". Analice qué sucede.

15º Trabajando como superusuario, cree el fichero "`/tmp/prueba`" con contenido nulo. Tras ello, aplique el comando "`chattr`" con el atributo especial necesario para protegerlo contra escritura. Consulte con el comando "`lsattr`" que ha sido bien aplicado. Intente borrar el fichero ("`rm /tmp/prueba`") y compruebe que no es posible. Quite el atributo especial al fichero e intente eliminarlo de nuevo con el comando "`rm`", comprobando que ahora sí puede.

16º Utilice los comandos "`find`", "`whereis`" y "`which`" para localizar el directorio en que se encuentre el comando "`ifconfig`". Analice la diferencia de funcionamiento entre dichos comandos.

17º Para el fichero "`/tmp/dir/otro2`", cree el enlace simbólico "`/tmp/link-soft`" y el enlace duro "`/tmp/link-hard`". Vea el resultado con el comando "`ls`"

18º Use el comando "`cat`" sobre los archivos "`/tmp/link-soft`" y "`/tmp/link-hard`" y compruebe que el resultado es el mismo. Borre "`/tmp/link-soft`" y compruebe que el comando "`cat /tmp/dir/otro2`" sigue funcionando. Borre el fichero "`/tmp/dir/otro2`" y compruebe que el comando "`cat /tmp/link-hard`" sí funciona. Justifique el motivo.

19º Para el directorio "`/tmp/dir/`", intente crear el enlace simbólico "`/tmp/link-soft-dir`" y el enlace duro "`/tmp/link-hard-dir`". Analice qué sucede. Haga uso del comando "`ls`" sobre el enlace simbólico "`/tmp/link-soft-dir`" y sobre el directorio "`/tmp/dir/`", y compruebe que coinciden. Haga uso del comando "`cd /tmp/link-soft-dir`" y compruebe cómo equivale a "cd`/tmp/dir/`" (para ello puede usar los comandos "`pwd -P`" y "ls").

20º Ejecute el siguiente comando para crear el fichero "/home/dit/fich" con el contenido:

```
cd /home/dit/
echo "Contenido del fichero" > ./fich
```

Tras ello, Ejecute los siguientes comandos de uso de enlaces simbólicos, y analice los resultados (incluyendo todos los campos de información del comando "ls -l" o su alias "ll"):

```
cat fich
ln –s  ./fich  ./enl_simb
ls -l
cat enl_simb
chmod 760 ./enl_simb
ls -l
rm –f enl_simb
ls -l
ln –s ./fich  ./enl_simb
rm –f ./fich
ls -l
ln –s /tmp ./enlace_a_dir
```

21º Ahora, ejecute los siguientes comandos (similares a los anteriores) de uso de enlaces duros, y analice los resultados (incluyendo todos los campos de información del comando "ls -l" o su alias "ll"), comparándolos asimismo con los resultados obtenidos con los enlaces simbólicos:

```
cd /home/dit/
echo "Contenido del fichero" > ./fich
cat fich
ln     ./fich  ./enl_duro
ls -l –i
cat enl_duro
chmod 760 ./enl_duro
ls -l
rm –f enl_duro
ls -l
ln    ./fich  ./enl_duro
rm –f ./fich
ls -l
ln    /dir ./enlace_a_dir
```

22º Ejecute los siguientes comandos tanto con el usuario “root” como con el usuario “dit”, compare las salidas obtenidas y determine por qué los comandos están dando esas salidas:

```
whereis ls
which ls
whereis ifconfig
which ifconfig
```

23º Ejecute los siguientes comandos y analice la salida:

```
dirname /usr/bin/ssh
basename /usr/bin/ssh
cd /usr/bin
dirname ./ssh
basename ./ssh
dirname ssh
basename ssh
```

TAREAS

4.3 Comandos relacionados con el control de dispositivos (20 minutos)

4.3.1 Identificación de los dispositivos en Linux

Los dispositivos físicos en Linux tienen asociados unos ficheros especiales en el directorio /dev (denominados ficheros de dispositivo). Cada vez que se accede a estos ficheros en realidad se está ejecutando un controlador de dispositivo (driver), que a su vez permite el acceso al hardware. Por tanto, los ficheros en /dev se comportan de manera diferente a los ficheros ordinarios. A continuación se resumen los dispositivos más comunes.

a) Discos duros: dentro de un PC pueden haber normalmente hasta cuatro discos IDE en total (dos por canal). Los discos duros (incluyendo los USBs) en Linux se identifican con el valor "/dev/sda", "/dev/sdb", ...

b) Unidades de CD/DVD-ROM: suelen tener asociado el identificador "/dev/sr0", "/dev/sr1", ..., aunque se les suele vincular un alias "/dev/cdrom0", "/dev/cdrw1", ...

d) Puertos serie: los dos dispositivos mas comunes que se conectan a los puertos serie suelen ser ratones y módems. En Linux los dispositivos son /dev/ttyS0 (COM1 en DOS) y /dev/ttyS1 (COM2 en DOS). Por lo general, aquellos que tengan un ratón conectado a un puerto de serie lo tendrán en /dev/ttyS0. Si el ratón es tipo PS/2 entonces estará conectado en un dispositivo especial llamado /dev/psaux y no usará ningún puerto serie.

e) Puertos paralelo: el uso más habitual para un puerto paralelo en el PC es la conexión a la impresora. El primer puerto paralelo, donde se suele conectar la impresora, se llama bajo Linux /dev/lp0 (line printer 0).

f) Consolas y terminales: las consolas virtuales en modo texto utilizan dispositivos especiales del tipo “/dev/ttyXX” (terminal). Las consolas de comandos bajo modo gráfico o remotas usan los dispositivos "/dev/pts/X".

Los puertos USB no tienen asignado un identificador concreto, sino que dependerá del tipo de dispositivo USB conectado (disco de almacenamiento, ratón, impresora, ...).

4.3.2 Dispositivos de almacenamiento

4.3.2.1 Particionamiento y formateo de dispositivos de almacenamiento

En Linux existen múltiples comandos para la segmentación de los discos de almacenamiento y asignación de un sistema de ficheros (`man filesystems`) determinado, destacando:

Básicos	`mount, umount, df, lsusb, fdisk, parted, gparted, mkfs.ext3, mkfs.vfat, mkfs.ntfs`
Para Profundizar	`du, udisks, cfdisk, sfdisk, e2label, mkfs, mlabel, ntfslabel, mkswap, swapoff, swapon, debugfs, fsck, e2fsck, tune2fs, dumpe2fs, resize2fs, findfs, blkid, hdparm, dd, partimage, fsarchiver, fdformat, mformat`

Todos estos comandos corresponden a tareas de administración, por lo que en general será necesario trabajar como "root" para poderlos utilizar.

Como operaciones básicas con estos comandos pueden hacerse las siguientes:

a) Para obtener un listado de los dispositivos de almacenamiento actualmente conectados al equipo y su identificador, basta ejecutar el comando:

```
fdisk –l
```

b) Para particionar el disco /dev/sda puede usarse el comando (recuerde que "fdisk" debe invocarse con el identificador de un disco -e.g. "`fdisk /dev/sda`"-, **no** de una partición -e.g. "`fdisk /dev/sda2`"-):

fdisk /dev/sda

Ello abre una consola en la que las principales pulsaciones son las siguientes:

Pulsación	Funcionalidad
h	Ayuda
p	Listado de particiones del disco
n	Crear nueva partición o unidad lógica (admite múltiples "K", "M", "G", ...)
d	Borrar una partición o unidad lógica
l	Listar códigos de sistemas de ficheros: `83` (ext2/3/4), `b` (FAT32), `7` (NTFS),...
t	Cambiar el sistema de ficheros de una partición (adviértase que sólo lo establecve, no le da formato)
w	Aplicar al disco los cambios antes realizados y salir (no olvidar invocar esta pulsación, si no los cambios se pierden)
q	Salir sin aplicar cambios

c) Para arreglar el sistema de ficheros de la partición /dev/sda1 puede ejecutar:

```
e2fsck –f –y /dev/sda1
```

d) Para formatear un disco USB en vfat con identificador /dev/sda3 (consultar "man mkfs") el comando sería:

```
mkfs.vfat    /dev/sda3
```

Como aclaraciones adicionales, deben realizarse las siguientes:

- "fdisk" sólo modifica la tabla de particiones, no la zona de almacenamiento de información del disco. Esto implica que "fdisk" realiza el particionamiento y asignación del tipo de ficheros (vfat, ext3, ...), pero no formatea (no prepara el contenido de la partición con ese sistema de ficheros), debiendo usar comandos "mkfs.xxx" para ello. El acceder a la tabla de particiones con una aplicación y a la zona de información con otra, puede llevar a incoherencias entre ambas, como por ejemplo, en el siguiente caso:

  - Con "fdisk" se crea una partición asignándole tipo de archivos "ext3".
  - Con "mkfs.vfat" se formatea dicha partición con sistema de archivos "FAT32".

  Sucederá que "fdisk" indicaría que la partición es de tipo "ext3" (este es el valor guardado en la tabla de particiones), mientras que estructura de fichros realmente existente en la partición es de tipo FAT32. Para evitarlo, deben usarse estas herramientas con precaución.

- "gparted" permite tanto particionar como formatear, asegurando la coherencia entre el tipo indicado en la tabla de particiones y el formato realmente aplicado a la zona de información.

- Por motivos de seguridad, tanto "fdisk" como "gparted" no aplican realmente los cambios sobre el disco hasta que no se indica expresamente: con "w" en fdisk, o pulsando en salir y seleccionando "Aplicar cambios" en gparted. Así, sea consciente que fdisk o gparted van mostrando las modificaciones de particiones que vaya indicando, pero si sale del programa sin indicar expresamente que se apliquen, dichas modificaciones no se habrán realizado (como podrá comprobar si vuelve cargar otra vez esas aplicaciones y mira el estado del disco).

TAREAS

Realice las siguientes tareas como superusuario:

1º Usando la herramienta "gparted", realice las siguientes operaciones:

- Analice cual es el particionamiento que tiene aplicado el disco de su equipo.
- Compruebe si su disco dispone de alguna partición extendida. Si no existe, créela (con, al menos, 200 MB de espacio).
- Sobre la partición extendida, cree dos unidades lógicas, ambas de 40 MB, la primera con sistema de ficheros NTFS (este sistema de ficheros no admite particiones inferiores a 32 MB), y la segunda con ext3.
- Salga de gparted (cuando se le pregunte. seleccione que se apliquen los cambios).

2º Ejecute el comando “cd /; umount /dev/sda5; umount /dev/sda6”. Tras ello, empleando el comando "fdisk", realice las siguientes operaciones:

- Corrobore que, efectivamente, el estado del disco es el esperado tras las operaciones realizadas con gparted.
- Elimine las dos unidades lógicas que creó antes con gparted.
- Vuelva a crear de nuevo dos unidades lógicas, pero ahora de 50 MB cada una, asignándoles el tipo de sistema de archivos FAT32 y ext3, respectivamente.
- Indíquele a “fdisk” que aplique los cambios (orden “w”) y salga de fdisk.

3º Ejecute el comando “cd /; umount /dev/sda5”. Tras ello, use los comandos "mkfs.ext3" y "mkfs.vfat" para formatear las unidades lógicas que acaba de crear con fdisk, la primera con sistema de ficheros FAT32, y la segunda con ext3. Observe si ambos comandos funcionan correctamente, o alguno de ellos da error.

4º Obtenga el espacio libre en el disco mediante los comandos “df -m” y “df -h”. Analice el significado de sus salidas.

4.3.2.2 Montaje de dispositivos de almacenamiento

Los archivos disponibles para sistemas Unix están dispuestos de forma jerárquica en un árbol cuya raíz está en “/”. Estos ficheros pueden estar distribuidos sobre varios dispositivos y sobre diferentes máquinas. Con el comando "mount" es posible acceder (montar) a cualquier dispositivo de almacenamiento (real o virtual) conectado al equipo, a través del sistema de archivos de la máquina en la que se utiliza dicho comando. La sintaxis general de este comando es:

```
mount –t tipo dispositivo directorio_montaje
```

Este comando le indica al núcleo del sistema operativo que anexe el sistema de archivos que encuentre en "dispositivo" (que es del tipo "tipo") al directorio "directorio_montaje". En muchos casos, el tipo de ficheros "-t tipo" no es necesario indicarlo, al reconocerlo el comando automáticamente (lo que equivale a "-t auto"). Puede obtener los posibles valor de "tipo" consultando el manual "`man mount`".

El comando "umount" realiza el proceso inverso. Sólo se pueden desmontar dispositivos que previamente se han montado y que no están en uso en ese momento. Este comando admite cualquiera de las dos sintaxis siguientes:

```
umount dispositivo
```

```
umount directorio_montaje
```

En el fichero "`/etc/fstab`" (man fstab) se guardan los dispositivos que se desean montar en el arranque del sistema (pasando su información al comando "mount"), o aquellos que se desean preparar para su automontaje (e.g., dispositivos USB). Los comandos "mount –a" o "umount -a" intentan montar o desmontar, respectivamente, todos los dispositivos recogidos en el fichero "/etc/fstab".

El fichero "`/etc/mtab`" contiene los dispositivos actualmente montados en el sistema, que también pueden obtenerse con los comandos "mount" o "cat /proc/mounts".

Salvo que se configure lo contrario, el des/montaje de dispositivos es una tarea de administración, sólo permitida al superusuario.

En el caso de los dispositivos USB, las disribuciones Linux modernas suelen disponer de un servicio de automontaje, encargado de montar el disco nada más conectarlo (en un directorio de la carpeta /media), sin necesidad de ejecutar manualmente el comando "mount". Si no funcionase correctamente, siempre podrá montarlo manualmente. Por ejemplo, para realizar el montaje de un disco USB con identificador /dev/sdb1 en el directorio /media/usb con el sistema de archivos FAT32 bastaría ejecutar:

```
mkdir /media/usb
```

```
mount -t vfat /dev/sdb1 /media/usb
```

y para desmontarlo bastaría "`umount /dev/sdb1`" o "`umount /media/usb`".

Para obtener el identificador de dispositivo asociado a un dispositivo USB recién conectado al equipo puede usarse, entre otras opciones, el comando "fdisk -l" (muestra todos los dispositivos de almacenamiento existentes) o también "`dmesg`" (muestras las operaciones de control realizadas por el kernel, debiendo aparecer en sus últimas líneas el dispositivo USB recién conectado).

TAREAS

Realice las siguientes operaciones:

1º Trabajando como superusuario, monte la unidad lógica "ext3" que creó y formateó en el apartado anterior, y cree en ella un fichero. Tras ello, desmóntela.

2º Monte la unidad lógica "FAT32" que creó y formateó en el apartado anterior, y cree en ella un fichero. Tras ello, desmóntela.

4.4 Comandos Orientados a la gestión de Procesos (20 minutos)

Linux dispone de múltiples comandos para la gestión de los procesos, entre los que destacan:

Básicos	`ps, pstree, top, pidof, kill, pkill, bg, jobs, fg, poweroff, reboot, shutdown, exec`
Para Profundizar	`xargs, watch, killall, suspend, pm-hibernate, sleep, pgrep, free, nice, nohup, halt, init, wall, ulimit`[6]

Por su frecuente uso, deben resaltarse especialmente los siguientes comandos:

Comando	Función

`ps [a][u][x][...]`
`[o pid,euser,cmd,...]`
 	
`pstree [PID]`
	Muestran los procesos en ejecución (suele ser útil aplicar filtrado con grep: "ps | grep nombre"). El parámetro "o" de "ps" permite especificar que datos de los procesos se desea imprimir.

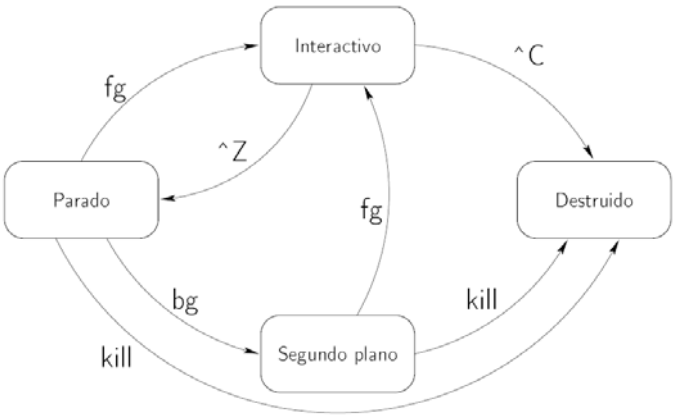
`pidof [-x] /ruta/programa`
 	
`pidof [-x] nombre_proceso`
	PIDs de los procesos correspondiente al programa o nombre de proceso indicado. Sin "-x" sólo busca entre los procesos del usuario actual y sin login shells (con "-x" busca entre todos los procesos del sistema).

`kill [-señal] PID`	Eliminar un proceso a partir de su PID.
`pkill reg_exp`	Eliminar todos los procesos cuyo nombre de proceso o comando asociado cumpla la expresión regular extendida (ERE) usada en"reg_exp".
`top`	Muestra los procesos del equipo que más CPU consumen

[6] "`ulimit`" es una orden interna del intérprete de comandos.

Comando	Función
`jobs`	Lista de los trabajos en background, parados y recien terminados en el shell actual (un comando terminado con el carácter &, e.g., "`comando &`", intenta ejecutarse en background).
`bg [Nº_trabajo]`	Intenta pasar a background un trabajo que se encuentra en estado parado[7] ("Ctrl-Z" pasa a estado parado el actual proceso en primer plano)[8]. Por omisión, "Nº_trabajo" es el último trabajo al que se le ha cambiado el estado), apareciendo marcado en jobs con el símbolo "+".
`fg [Nº_trabajo]`	Paso a primer plano un trabajoque se encuentra en estado background o parado ("Ctrl-C" termina el actual proceso en primer plano). Por omisión, "Nº_trabajo es el último trabajo al que se le ha cambiado el estado), apareciendo marcado en jobs con el símbolo "+".
`reboot`	Reinicio del equipo (equivale a "shutdown –r now" ó "init 6").
`poweroff`	Apagado del equipo (equivale a "shutdown –h now" ó "init 0").
`exec orden`	Cierra el shell actual y ejecuta el comando indicado.

En Linux cada proceso cargado en memoria tiene asignado un identificador único o PID (Process IDentifier). Un trabajo (identificado por un número indicado por "jobs") puede estar formado por varios procesos.



[7] El cambio de estado de "Detenido" a background será posible si la naturaleza del comando le permite trabajar en segundo plano (e.g., "firefox" sí lo permite, mientras que "cat" no, dado que necesita estar en primer plano para poder leer la información de la consola).

[8] Al suspender un proceso con "Ctrl-Z", aparece en pantalla el mensaje "`[Nº_proceso] Detenido comando`".

Por ejemplo, el comando:

- "`ls -lR /usr/ | wc -l &  ps`": muestra los procesos actuales, cada uno (con su PID) en una línea.
- "`ls -lR /usr/ | wc -l &  jobs`": muestra los trabajos actuales, pudiendo observar que en un mismo trabajo están implicados los procesos de varios comandos.

Un trabajo puede encontrarse en cuatro estados posibles: en primer plano o "foreground" (ocupa el intérprete de comandos), en segundo plano o "background" (en ejecución siempre, permitiendo que el intérprete de comandos quede libre para seguir utilizándolo, es marcado por "jobs" como [Ejecutando]"), parado (marcado por "jobs" como [Detenido]") y destruido o recién terminado (marcado por "jobs" como [Hecho]"). Cuando un trabajo puesto en background termina autónomamente, en la salida del siguiente comando ejecutado en la consola aparecerá el mensaje "[Nº_trabajo] Hecho comando" (e.g., al ejecutar el comando "cat fichero &"). Al ejecutar un proceso en background, puede pasar automáticamente a estado parado si su funcionalidad no le permite trabajar en segundo plano (e.g., comando "cat &").

En Linux, para "matar" un determinado proceso, el comando "kill" (o similares) envía una determinada señal al proceso (puede ver las distintas señales posibles con "`man kill`" y "`man 7 signal`"). Por ejemplo, "`kill -9 PID_proceso`", envía al proceso la señal "KILL", que obliga al proceso a cerrarse (no puede ser bloqueada por el proceso). El comando "`kill -1 PID_proceso`" envía la señal "HUP", que solicita al proceso su cierre ordenado.

Para matar el último proceso puesto en background puede usarse el comando "`kill $!`" (la variable "$!" devuelve el PID del último proceso puesto en background).

TAREAS

1º Consulte el manual de ayuda de los distintos comandos mencionados para conocer su funcionalidad.

2º Ejecute los siguientes comandos y determine la diferencia en sus salidas:

```
ps
ps a
ps au
ps aux
pstree
pidof bash
pidof b
xclock &
pkill xclock
xclock &
pkill -9 xclock
xclock &
pidof xclock
kill  escriba_aqui_el_PID_de_xclock
```

TAREAS

3º Analice cuál es la diferencia de funcionalidad entre los comandos: `reboot, poweroff, shutdown`.

4º Haga uso del comando "`exec xclock`" y analice cual es su utilidad.

5º Use el comando "ps" para listar los procesos arrancados desde el proceso shell actual. Tras ello, ejecute el comando "ps ax" para obtener un listado de todos los procesos existentes en sistema (de todos los usuarios). Compare ambas salidas. Por último, ejecute el comando "`pstree`" y analice que significa la información que muestra.

6º Ejecute en backgroung las aplicaciones gráficas "`xclock`" y "`xcalc`". Vuelva a poner a "`xcalc`" en foreground (primer plano) y luego vuelva a ponerla en background usando la pulsación "Ctrl-Z "(si hubiese usado "Ctrl-C" habría cerrado el proceso en lugar de pasarlo a foreground).

7º Ejecute el comando "`firefox &`", comprobando que se abre una ventana gráfica con el navegador "Iceweasel" y que ésta funciona correctamente (acceda, por ejemplo, a Google). En la consola, vea que ésta sigue operativa para escribir nuevos comandos (proceso en background). Ejecute el comando "`jobs`", comprobando que aparece el trabajo "firefox" en estado background ([Ejecutando]). Cierre la ventana gráfica de "Iceweasel". Vuelva a ejecutar el comando "`jobs`" y compruebe que ya no aparece el trabajo "firefox".

8º Ejecute el comando "`firefox`", comprobando que vuelve a abrirse una ventana gráfica con el navegador "Iceweasel" y que ésta funciona correctamente (acceda, por ejemplo, a Google). En la consola, vea ahora que no está operativa para escribir nuevos comandos (el proceso y trabajo "firefox" está en primer plano). Pulse "Ctrl-Z" y ejecute el comando "jobs", comprobando que el trabajo "firefox" se encuentra ahora en estado parado. Vaya a la ventana gráfica de Iceweasel y compruebe cómo ahora no funciona (no puede operar con ella al estar el trabajo detenido).

9º En la anterior ejecución del comando "jobs", observe el número de trabajo asociado a "firefox" y ejecute el comando "`bg Nº_trabajo`" usando dicho número. Vuelva a ejecutar el comando "`jobs`" y compruebe que ahora el trabajo "firefox" se encuentra en estado background ([Ejecutando]). Vaya a la ventana gráfica de Iceweasel y compruebe cómo vuelve a estar operativa (trabajo ejecutándose en background).

10º En la consola, ejecute el comando "`fg Nº_trabajo`" usando el número de trabajo de "firefox". Vaya a la ventana gráfica de Iceweasel y compruebe que sigue estando operativa (trabajo en primer plano). Finalmente, pulse "Ctrl-C" en la consola, lo que terminará el proceso y trabajo "firefox" en primer plano, cerrándose consecuentemente su ventana gráfica.

TAREAS

11º Arranque **en background** "5" instancias de la aplicación gráfica "`xclock`" y dos instancias de la aplicación gráfica "`xcalc`".

12º Utilice los comandos necesarios para cerrar la primera y quinta instancia de la aplicación "`xclock`", pero no las demás.

13º Utilice el comando necesario para cerrar las dos instancias de la aplicación gráfica "`xcalc`", sin hacer uso del PID y sin afectar a las instancias "`xclock`".

14º Vuelva a arrancar una instancia de la aplicación gráfica "`xcalc`". Tras ello, utilice el comando necesario para cerrar en una sola instrucción, y sin tener que indicarle los PIDs, las diferentes instancias aún existentes de las aplicaciones gráficas "`xclock`" y "`xcalc`".

4.5 Comandos relacionados con la instalación de software (5 minutos)

En Linux existen varias maneras de instalar software. A continuación se comentan las más habituales:

4.5.1 Instalación con paquetes

El software para distribuciones Linux Debian suele venir en "paquetes". Un paquete contiene un programa software con todos los archivos necesarios así como información de donde copiarlos. Existen distintos formatos de paquetes y cada distribución utiliza uno u otro. Linux Debian utiliza los paquetes tipo ".deb". Cada paquete es específico de un kernel y una distribución. Un paquete creado para otra distribución probablemente no se instalará bien, ya que suelen ser archivos compilados con las características específicas de una distribución.

El comando para gestionar manualmente paquetes en Linux Debian es "dpkg" (man dpkg). Se trata de una herramienta de gestión de paquetes de software que se puede utilizar para instalar, verificar, actualizar y borrar otros paquetes de software de forma individual. También permite comprobar si un paquete está instalado y de qué versión se trata:

Instalar	`dpkg –i paquete.deb`
Desinstalar	`dpkg –r paquete.deb`

Asimismo, la instalación de paquetes puede hacerse mediante repositorios, usando el comando "apt-get", "aptitude" o la herramienta gráfica "synaptic". Estas utilidades buscan el paquete que se desea y resuelven las dependencias con otros paquetes de manera automática. Por ejemplo:

Instalar	`apt-get install paquete[=version]`
Desinstalar	`apt-get remove paquete`
Actualizar índice de paquetes	`apt-get update`
Actualizar paquetes	`apt-get upgrade [paquete]` `apt-get dist-upgrade`
Búsqueda en repositorios (nombre y descripción paquetes)	`apt-cache search cadena`
Búsqueda de paquetes y detalles	`apt-cache policy paquete`

La herramienta "apt-cache" permite muchas otras opciones de búsqueda (dependencias de un paquete, lista de paquetes instalados, ...).

TAREAS Usando "apt-get", intente instalar el editor de textos "jedit" (observará cómo, además del paquete del editor, la herramienta también instalará otros paquetes que éste necesita). Luego desinstálelo.

4.5.2 Instalación a partir de archivos binarios

Algunos programas se instalan a partir de un fichero ejecutable al igual que ocurre en los sistemas Windows. En este caso sólo será necesario ejecutar dicho programa.

4.5.3 Instalación a partir del código fuente

En Linux suele ser habitual que el código fuente sea gratuito y en vez de proporcionar el programa ya compilado, se ofrezca el código fuente para que cada usuario lo modifique y compile en función de su sistema operativo. Es posible que para compilar el código fuente necesite tener instaladas algunas librerías por lo que se recomienda leer detenidamente la documentación incluida con el código fuente.

Normalmente el código fuente viene comprimido en un archivo tar.gz o zip y los comandos para descomprimirlos suelen ser:

```
tar xvzf fichero.tar.gz   /o bien/   unzip fichero.zip

cd directorio_codigo_fuente
```

Dentro del código fuente suele existir información de cómo realizar la compilación e instalación. Para ello debemos buscar y leer ficheros con nombres tales como: README, configure, Makefile, INSTALL, etc.

Los comandos más habituales que habría que ejecutar para compilar e instalar son:

```
./configure  [--prefix=/usr]
make
make install
```

5. Funcionamiento en modo texto: otros comandos útiles (55 minutos)

5.1 Comandos relacionados con ficheros de texto (35 minutos)

Además de los editores de texto habituales (`emacs, mcedit, nedit, nano, vi, ...`), Linux dispone de múltiples comandos para operar con los ficheros de texto, tales como:

Básicos	`touch, echo, diff, cmp, more, less, cat, tac, head, tail, dos2unix, unix2dos, sort, od, tee, uniq, wc, cut, grep, sed`

A continuación se resume la funcionalidad de estos comandos:

Comando	Función
`emacs, nano, mcedit`	Ejemplos de editores texto en modo texto
`emacs, nedit`	Ejemplos de editores texto en modo gráfico
`cat fichero` `tac fichero`	Muestra el contenido del fichero de texto indicado (“tac” orden inverso)
`more fichero` `less fichero`	Muestra la salida en partes, permitiendo avance (“espacio” o “AvPág”) y retroceso (“b” o “RePág”) por pantallas
`tail [-n xx] fichero`	Muestra las últimas "xx" líneas de un fichero ("10" por omisión)
`head [-n xx] fichero`	Muestra las primeras "xx" líneas de un fichero ("10" por omisión)
`touch fichero_nuevo`	Actualiza la fecha de un archivo. Si no existe lo crea.
`echo texto`	Imprime texto en la salida estándar
`diff  fich1  fich2`	Muestra las diferencias entre dos ficheros

Comando	Función
`cmp fich1 fich2`	Compara dos ficheros, indicando si son iguales o no
`dos2unix fichero` `unix2dos fichero`	Adapta ficheros de texto de Windows a Linux (elimina los retornos carro, dejando sólo los avances de línea) o viceversa
`sort [-n/R/...] fichero`	Ordena las líneas de fichero: alfabéticamente (por omisión), numéricamente (-n), aleatoriamente (-R), ...
`od [-t c/x/f...] fichero`	Imprime el resultado de convertir el contenido del fichero a octal (por defecto), ASCII (c), hexadecimal (x), flotante (f), ...
`tee [fichero1] [fichero2] [...]`	Copia la entrada estándar tanto en los ficheros indicados como en la salida estándar.
`uniq fichero`	Elimina líneas repetidas.
`wc [-l] [-w] [-c] fichero`	Imprime el número de líneas (-l), palabras (-w) y bytes o caracteres ("-c" o "-m") del contenido del fichero
`cut [-bx,... | -cx,... | -fx,... -d "c"] fichero`	Extrae los bytes, caracteres o campos (separados por el carácter delimitador "c") ubicados en la posición (o posiciones) "x". Usa "tabulación" como delimitador por defecto.
`grep [-v] [-q] [-E] [-P] "reg_exp" fichero`	Imprime en pantalla las líneas de "fichero" (o, como siempre, de la salida estándar de un "comando" mediante tuberías) que contienen la cadena "reg_exp" (escribir el patrón entre comillas sólo es necesario si contiene caracteres de espaciado). "-v" invierte la búsqueda (imprime todo lo que no coincida con el patrón). "reg_exp" admite expresiones regulares básicas (BRE); con "-E" admite expresiones regulares expandidas ERE, y con "-P" admite expresiones PCRE. "-q" no imprime en pantalla (sólo comprueba).
`sed [-i destino] -e "reg_exp" fichero`	Manipulador de textos (editor de flujo): imprime en pantalla el contenido de "fichero" tras aplicarle las operaciones (de sustitución...) indicadas en la expresión regular "reg_exp" básica BRE (añadir "-r" para soportar extendidads ERE). Imprime en la salida estándar (por omisión) o en el fichero "destino".

Para la escritura de la cadena "reg_exp" de los comandos "grep" y "sed" (find y pkill) pueden usarse expresiones regulares. Las expresiones regulares son cadenas de texto con caracteres que tienen un significado especial (e.g. "^" principio de la línea, "$" final de la línea, ...). Existe tres sintaxis posibles para la escrituras de las expresiones regulares:

- POSIX Básica (BRE) y Extendida (ERE): pueden consultarse con "`man grep`" (apartado "REGULAR EXPRESSIONS"), estando normalizadas en el estándar POSIX IEEE Standard 1003.1 2004/ISO 9945.2:

  http://pubs.opengroup.org/onlinepubs/009695399/basedefs/xbd_chap09.html

- Compatibles con Perl (PCRE): usan la misma sintaxis de expresiones del lenguaje Perl. Pueden consultarse con "`man pcrepattern`", o en la definición de Perl:

  http://perldoc.perl.org/perlre.html

Aunque depende de la implementación usada, las versiones GNU de "grep" suelen soportar las tres sintaxis, y las de "sed", al menos la sintaxis básica BRE. Los comando "find" y "pkill" vistos anteriormente suelen soportar en su patrón la sintaxis extendida ERE de expresiones regulares.

Los principales caracteres especiales que pueden usarse para las expresiones regulares básicas (BRE) son:

^	Comienzo de la línea
$	Final de la línea
.	Carácter único (cualquier carácter)
*	***Cero*** o más ocurrencias del carácter previo
[...]	Uno o varios de los caracteres indicados entre los corchetes
[^...]	Ninguno de los caracteres indicados entre corchetes

Las expresiones extendidas (requieren "grep -E") ofrecen muchos más operadores. Por ejemplo: "?" (Cero o una ocurrencia del carácter previo), "+" (Una o más ocurrencias del carácter previo), "{n, m}" (entre "n" y "m" ocurrencias del carácter previo), "|" (expresión regular con varias posibles cadenas de búsqueda, "cadena1|cadena2|...").

Para escribir una expresión regular en sed ésta debe escribirse entre barras: "/expresion/".

Para Profundizar	Las expresiones regulares ofrecen una funcionalidad muy potente, documentada en las referencias antes citadas para que pueda profundizar. En las evaluaciones de la Asignatura no se exigirá un conocimiento de las expresiones regulares adicional al explicado en esta práctica.

TAREAS

Realice las siguientes operaciones:

1º Consulte el manual de ayuda de los distintos comandos mencionados para conocer su funcionalidad.

2º Usando "`echo`", cree el fichero "`/tmp/texto1`" con el contenido "cadena1". Usando un editor de textos, cree el fichero "`/tmp/texto2`" con el contenido "cadena2".

3º Compare ambos ficheros con los comandos "`diff`" y "`cmp`", y analice que tipo de información aporta cada uno de ambos comandos.

4º Cree el fichero vacío "`/tmp/vacio`" con el comando "`touch`" e imprima su contenido en pantalla para comprobar que efectivamente no contiene nada.

5º Visualice el fichero "`/etc/passwd`" a través de los filtros "`more`" y "`less`", y compare la diferencia entre ambos.

6º Visualice "5" líneas del fichero "`/etc/fstab`" mediante los comandos "`tail`" y "`head`", y analice la diferencia entre ambos.

7º Utilice el comando adecuado para obtener el número de líneas y palabras que contiene el fichero "`/etc/fstab`".

8º Haga uso de los comandos "`cat`" y "`tac`" sobre el fichero "`/etc/fstab`". Determine cual es la diferencia de funcionalidad entre ambos comandos.

9º Ejecute el comando "wc /etc/fstab" para obtener el número de líneas, palabras y bytes, seguidos del nombre del fichero "/etc/fstab". Ejecute ahora "cat /etc/fstab | wc -l" o "wc -l /etc/fstab" para que sólo imprima el número de líneas.

5.1.1 Ejemplos prácticos de uso de Filtros de Texto

Los comandos de manipulación de textos presentados en este apartado resultan especialmente útiles dentro de los shell-scripts (que se estudiarán en un tema posteriores). Por ejemplo, son muy habituales los shell-scripts que usan estos comandos para procesar la información de salida de otros comandos o para procesar los datos de los formularios de páginas web CGI basadas en shell-scripts. Consecuentemente, **resulta importante que aprenda a usar estos comandos con cierta soltura**. A continuación se le plantean algunos ejercicios resueltos de usos prácticos de estos comandos para procesar información, **analice con detenimiento cómo están funcionando estos comandos**:

TAREAS

1º Comando que imprima, de todos los procesos del sistema, aquellos asociados al intérprete “bash”:

```
ps ax | grep bash
```

2º Comando que imprima todas las líneas del fichero “/etc/passwd” que contengan la cadena “dit” y otro para las líneas que contengan la cadena “/bin/bash”:

```
cat /etc/passwd | grep dit
```

```
grep /bin/bash /etc/passwd
```

3º Comando que imprima la línea de “/etc/passwd” del usuario “dit:

```
grep -e "^dit:" /etc/passwd
```

Adviértase que esta línea no equivale al comando “cat /etc/passwd | grep dit”, dado que éste busca cualquier línea que contenga la cadena “dit”, mientras que la expresión regular anterior obliga que “dit” sea la primera palabra de la línea y que vaya seguida del separador “:”.

4º Comando que imprima la línea de "/etc/passwd" de aquellos usuarios que tengan configurado el shell “/bin/bash”:

```
grep ":/bin/bash$" /etc/passwd
```

5º Comando que imprima la línea de “/etc/passwd” de todos los usuarios que tengan configurado el shell “/bin/bash”, y no sean el usuario “dit”

```
grep ":/bin/bash$" /etc/passwd | grep -v -e "^dit:"
```

6º Comando que imprima, únicamente, el número de sesiones actualmente abiertas por el usuario “dit”:

```
who | grep "^dit " | wc -l
```

7º Comando que imprima la línea de “/etc/passwd” del usuario con UID “0”:

```
grep ".*:.*:0:.*" /etc/passwd
```

8º Comando que imprima la línea de “/etc/passwd” del usuario con GID “0”:

```
grep ".*:.*:.*:0:.*" /etc/passwd
```

9º Comando que imprima una lista con los usuarios definidos en el sistema:

```
cut -f1 -d ":" /etc/passwd
```

TAREAS

10º Comando que imprima una lista con los usuarios definidos en el sistema y su shell asociado:

```
cut -f1,7 -d ":" /etc/passwd
```

11º Comando que imprima una lista de los grupos definidos en el sistema y sus miembros:

```
cut -f1,4 -d ":" /etc/group
```

12º Comando que imprima el identificador de los dispositivos recogidos en "/etc/fstab":

```
grep -v "^#" /etc/fstab | cut -f1 -d " "
```

13º Comando que imprima los directorios donde, según el comando "mount", están actualmente montados los dispositivos físicos de almacenamiento:

```
mount | grep "^/" | cut -f3 -d " "
```

Adviértase que "cut" está usando un espacio como delimitador por defecto, dado que muchos comandos (como por ejemplo "mount", "route" o "arp") "formatean" la información en la salida estándar usando espacios en lugar de tabulaciones.

14º Comando que imprima la información detallada de los procesos que pertenecen al usuario efectivo "dit":

```
ps aux | grep "^dit "
```

15º Comando que imprima la dirección IP de la pasarela mostrada por el comando "route" (asumiendo que sólo hay una pasarela y está asociada a la red "default" 0.0.0.0):

```
/sbin/route -n | grep "^0.0.0.0" | cut -f10 -d " "
```

Adviértase cómo, al usar espacio como delimitador, ha sido necesario calcular el número de espacios para conocer el número del campo que contiene la dirección IP buscada.

16º Comando que imprima el número de entradas asociadas a la interfaz eth0 que posee la tabla de encaminamiento impresa por el comando "route -n":

```
/sbin/route -n | grep eth0 | wc -l
```

17º Comando que sólo imprima las"2" primeras entradas de la tabla de encaminamiento impresa por el comando "route -n":

```
/sbin/route -n | head -n 4 | tail -n 2
```

TAREAS

18º Comando que imprima el fichero "/etc/passwd" pero sustituyendo los separadores ":" por "#" ("s" se usa para indicar "reemplazo, y "g" para que sea global y no sólo en la primera aparición):

```
sed -e "s/:/#/g" /etc/passwd
```

19º Comando que imprima la línea de "/etc/passwd" del usuario "dit" pero sustituyendo los separadores ":" por espacios:

```
grep "^dit:" /etc/passwd | sed -e "s/:/ /g"
```

20º "cut" permite usar rangos para indicar los campos a imprimir. Por ejemplo, comando que, para el usuario dit, imprima:

- Su campo de clave en /etc/passwd, UID y GID (campos 2 a 4):

```
grep "^dit:" /etc/passwd | cut -d ":" -f2-4
```

- Su campo de login, clave en /etc/passwd, UID y GID (campos 1 a 4):

```
grep "^dit:" /etc/passwd | cut -d ":" -f-4
```

- Su carpeta personal y shell en el fichero /etc/passwd (campos 6 a 7, siendo "7" el último):

```
grep "^dit:" /etc/passwd | cut -d ":" -f6-
```

Además de los comandos aquí mencionados, para la manipulación de cadenas de caracteres existen herramientas específicas muy utilizadas dentro de los shell-scripts, tales como el lenguaje "awk", para el que su sistema ofrece el intérprete "/usr/bin/mawk" (man awk).

5.2 Comandos relacionados con el empaquetado y compresión (10 minutos)

Linux dispone de múltiples comandos para empaquetar y comprimir la información, entre ellos:

Básicos	`tar, 7z`
Para Profundizar	`gzip, gunzip, zcat, bzip2, bunzip2, zip, unzip, 7za, afio, compress, uncompress`

El comando "tar" puede ser usado sólo para empaquetar (ficheros ".tar", o para empaquetar y comprimir, admitiendo compresión en varios algoritmos (entre otros, gzip, bzip2 y xz). Por su frecuente aplicación, se indica a continuación la sintaxis básica de uso de este comando:

Empaquetado y compresión en gzip (.tgz)	`tar cpfvz fichero.tar.gz dir1 file2 ...`
Empaquetado y compresión en bzip2 (.tbz)	`tar cpfvj fichero.tar.bz2 dir1 file2 ...`
Empaquetado y compresión en xz (.xfz)	`tar cpfvJ fichero.tar.xz dir1 file2 ...`
Desempaquetado y descompresión de gzip	`tar xpfvz fichero.tar.gz [-C dir_destino]`
Desempaquetado y descompresión de bzip2	`tar xpfvj fichero.tar.bz2 [-C dir_destino]`
Desempaquetado y descompresión de xz	`tar xpfvJ fichero.tar.xz [-C dir_destino]`

con el siguiente significado de parámetros (si no se indica ni "z" ni "j" sólo se empaqueta, no se comprime):

- "c": empaquetar o comprimir (Compress).
- "x": desempaquetar o descomprimir (eXtract).
- "p": conservar los permisos de los ficheros.
- "f": trabajar con ficheros (si no se pusiese, se leerían bloques de información del disco, no distinguiendo ficheros).
- "v": modo verbose (imprimir información por pantalla).
- "z": des/comprimir bajo el algoritmo "gzip".
- "j": des/comprimir bajo el algoritmo "bzip2".
- "J": des/comprimir bajo el algoritmo "xz".

Para el uso del comando 7z (algoritmo 7-zip) se emplearía la siguiente sintaxis:

Compresión ("-r" recursivo)	`7z a [-r] fichero.7z dir1 file2 ...`
Descompresión	`7z x fichero.7z [-oDir_Destino]`
Ver listado del contenido de un 7z	`7z l fichero.7z`

Debe advertirse que “7z” no guarda el usuario/grupo propietario de los ficheros, por lo que no resulta adecuado para realizar backups de directorios Unix, siendo más adecuado usar en su lugar el comando “tar”.

TAREAS

1º Busque información con el comando man de cómo utilizar las herramientas indicas.

2º Comprima la carpeta "/etc" en el fichero "/tmp/conf.tar.gz". Luego descomprima dicho fichero en el directorio "/tmp/desc".

3º Repita la tarea 2º pero usando ahora el comando 7z.

5.3 Comandos relacionados con la conexión a otras máquinas (10 minutos)

Los comandos más habituales son:

Básicos	`ssh, scp, sftp, telnet, ftp`

Los comandos ssh/telnet permiten abrir un intérprete de comandos (e.g. bash) en un equipo remoto (al igual que con los intérpretes locales, para cerrar ese intérprete remoto se usa el comando “exit”). Los comandos ftp/sftp/scp se emplean para la transferencia de ficheros.

La sintaxis básica de estos comandos es la siguiente:

Comandos inseguros	`telnet IP [puerto]` `ftp IP [puerto]`

Comandos seguros (si no se indica el usuario, se usa el actual en la consola de comandos)	`ssh [–oPort=puerto] [user@]IP`
	`ssh [–oPort=puerto] [user@]IP comando`
	`sftp [–oPort=puerto] [user@]IP`
	`scp [–oPort=puerto] fichero [user@]IP`

El uso básico de ftp/sftp se realiza con los siguientes comandos (algunos de los comandos internos de ftp y sftp no coinciden):

Comando	Función
`help`	Lista de comandos disponibles
`cd ruta_remota`	Cambiar de directorio en el servidor remoto
`lcd [ruta_local]`	Cambiar de directorio en el equipo local

Comando	Función
`ls [ruta_remota]`	Lista contenido de directorio remoto (=dir)
`binary`	Transferir archivos en binario (por defecto)
`ascii`	Transferir archivos en ASCII
`put [ruta_local]fichero`	Enviar archivo al servidor remoto (=send)
`mput [ruta_local]patrón`	Enviar varios archivos al servidor remoto
`get [ruta_remota]fichero`	Traer un archivo del servidor remoto (=recv)
`mget [ruta_remota]patrón`	Traer varios archivos del servidor remoto
`rm [ruta_remota]fichero`	Borra un archivo en el servidor remoto
`rmdir [ruta_rem]`	Borra directorio en el servidor remoto
`mkdir [rura_remota]dir`	Crear directorio en el servidor remoto
`rename [ruta_remota]fichero`	Renombra un fichero en el servidor remoto
`quit`	Cierra la sesión (=bye)

Los distintos equipos de la sala están dotados de un servidor SSH (por omisión parados), configurado para que escuche en el puerto estándar "22".

TAREAS

Realice las siguientes tareas:

1º Acceda vía "sftp" al equipo cuya IP obtuvo anteriormente y envíele el fichero "/etc/fstab", copiándolo en el directorio "/tmp/" del equipo remoto.

2º Acceda vía SSH a ese equipo remoto y ejecute en él el comando "ifconfig" para corroborar que efectivamente está trabajando en dicha máquina. Asimismo, compruebe que el fichero "fstab" que anteriormente transfirió vía "sftp" efectivamente se encuentra en "/tmp/fstab".

6. Anexo

[1] Durante el proceso de arranque del sistema, las variables de shell son configuradas en múltiples ficheros, destacando: "`/etc/profile`" (común para todos los usuarios, salvo para el usuario "root", para el que aplica una configuración particular) y "`$HOME/.bashrc`", "`$HOME/.bash_profile`" (particulares de cada usuario y dependientes del intérprete de comandos, pudiendo existir o no de forma opcional; los ficheros indicados son los empleados por el intérprete Bash). La lectura de dichos ficheros (que corresponden a scripts del shell) por parte del sistema se realiza en el mismo orden en que se han mencionado.

[2] Nombre de usuario: Aunque para el campo "login" resulta posible usar un número, no resulta recomendable, aconsejándose emplear cadenas que comiencen por letras. Por omisión, los comandos "adduser" (y "addgroup") sólo admiten como nombre de usuario (o de grupo) cadenas que se ajusten al formato definido por el parámetro "NAME_REGEX" en el fichero "`/etc/adduser.conf`" (habitualmente, cadenas alfanuméricas y con el carácter "_", sólo minúsculas, que comiencen con un carácter alfabético).

[3] Por defecto, el control del acceso en los sistemas Linux es gestionado por el sistema PAM (Pluggable Authentication Module); en concreto, en Debian, las características de las claves permitidas para los usuarios son configuradas en el fichero "`/etc/pam.d/common-password`" ("man pam.conf", "man pam_unix") indicándose, por ejemplo, si se permite o no el acceso de un usuario sin clave, la longitud mínima que debe tener una clave o el algoritmo usado para la encriptación de las claves (MD5, SHA, DEC, ...).

[4] El valor para la variable "LS_COLORS" está constituido por un conjunto de tipos de ficheros y un código de color para cada uno. El valor de la variable lo configura el comando "dircolors" (man dircolors) a partir del contenido de los ficheros " /etc/dir_colors" (global) y "$HOME/dir_colors" (por usuario), con sintaxis definida en "man dir_colors". Si no existen, se usa la configuración predeterminada que puede obtenerse con el comando "dircolors --print-database".

[5] Secciones de manual (man):

Sección (x)	Tipo
1	Programas ejecutables o comandos del interprete de ordenes
2	Llamadas de sistema
3	Llamadas de bibliotecas
4	Archivos especiales (alojados normalmente en /dev)

Sección (x)	Tipo
5	Formatos de archivo y convenciones
6	Juegos
7	Miscelánea
8	Comandos de administración del sistema
9	Rutinas no estándar del núcleo
n	Documentación nueva, que podría ser trasladada posteriormente
l	Documentación local especifica de su sistema

[6] Para ello se emplean los siguientes ficheros (usados por "/bin/login"):

Fichero regular	Funcionalidad
`/etc/securetty`	Contiene las consolas virtuales en modo texto locales "tty/X" (no las emuladas "pts/X") y gráficas ("X") para las que se permite acceso como usuario "root"
`/etc/nologin`[9]	Si existe, sólo se permite el acceso al usuario "root" (afecta a las consolas virtuales en modo texto locales "tty/X" y gráficas "X"). El contenido de este archivo (si lo tiene), será mostrado a los usuarios normales cuando intentan entrar al sistema y se les deniega el acceso.
`/etc/pam.d/login`	Controla la activación de los ficheros anteriores: securetty (requiere la directiva "auth requisite pam_securetty.so"), nologin (requiere la directiva "auth requisite pam_nologin.so").

En cuanto a las sesiones remotas, el control de acceso dependerá del servidor. Por ejemplo, el servidor "telnet" aplica el fichero "/etc/securetty", mientras que el servidor SSH controla directamente si el usuario "root" tiene permiso de acceso (en la directiva "PermiRootLogin" de su fichero de configuración "/etc/ssh/sshd_config").

[7] Otros directorios del sistema de archivos

Directorio	Contenido
`/boot`	[Obligatorio] Contiene todos los archivos que se necesitan para arrancar el sistema a excepción de los archivos de configuración.

[9] Si en lugar de crear el archivo regular "`/etc/nologin`", se crea el directorio "`/etc/nologin/`", se bloqueará el acceso a todos los usuarios del sistema, incluyendo al usuario "root" (tenga precaución con ello).

Directorio	Contenido
`/lib`	[Obligatorio] Contiene librerías compartidas que se utilizan en el momento de arranque del sistema o que usan algunos comandos de alto nivel que están ubicados en el directorio "`/bin`". Por su parte, las librerías que se utilizan para dar soporte a los comandos contenidos en el directorio /usr están contenidas en el directorio "`/usr/lib`".
`/opt`	[Opcional] Está pensado para contener todos los datos requeridos para dar soporte a los paquetes software que se añadan al sistema original.
`/root`	[Opcional] Carpeta personal del usuario root. Contiene información y archivos de configuración necesarios para la cuenta del usuario root.
`/srv`	[Obligatorio] Debe contener los datos de los servicios que este sistema proporciona.
`/sys`	[Opcional] Sistema de archivos virtual ("`sysfs`") que contiene información sobre el hardware del equipo.
`/usr`	[Obligatorio] Contiene datos y aplicaciones de utilidad diversa.

[8] Algunos casos habituales de búsqueda de ficheros:

Sintaxis: `find [ruta] [expresión_de_búsqueda] [acción]`		
Operación	Sensitive	Comandos/Ejemplos
Buscar ficheros por nombre	Sí	`find /rutas -name cadena` `find /rutas -name "[a-m]*.txt"`
	No	`find /rutas -iname cadena`
Buscar ficheros por contenido	Sí	`grep -R "cadena" /ruta/` `find /ruta -type f | xargs grep "cadena"`
	No	`grep -i -R "cadena" /ruta/` `find /ruta -type f | xargs grep -i "cad"`
Buscar ficheros por nombre y contenido	Sí	`grep -R --include="*.conf" "cadena" /ruta` `find /ruta -type f -name "*.conf"	xargs grep "cadena"`
	En cadena búsqueda	`find /ruta -type f -name "*.conf"	xargs grep -i "cadena"`

Sintaxis: `find [ruta] [expresión_de_búsqueda] [acción]`		
Operación	Sensitive	Comandos/Ejemplos
	En cadena y nombre	`grep -i -R --include="*.conf" "cad" /ruta` `find /ruta -type f -iname "*.conf"	xargs grep -i "cadena"`
Buscar archivos vacíos		`find /ruta -size 0c`
Eliminar (acción) los ficheros con tamaño mayor a "1 MB" en /tmp		`find /tmp -size +1000k -exec rm -f {} \;`

[9] El proceso shell de la sesión de comandos actual va guardando en memoria los comandos ejecutados en dicha sesión, no volcándolos en el fichero ".bash_history" (o similar para otros shells) hasta que se cierre ese proceso del intérprete (con el comando exit, kill, cierre de la ventana, ...). Por su parte, el comando "history" muestra el histórico completo, imprimiendo el contenido del fichero ".bash_history" seguido de ese conjunto de comandos de la sesión actual almacenado en memoria. Si se desea cerrar la sesión actual sin añadir ningún comando al fichero ".bash_history", puede ejecutarse el comando "history -c" (limpia la lista de comandos de la sesión actual guardada en memoria) y a continuación pulsar "Ctrl-D" (carácter de fin de fichero, provocando el cierre de la sesión).

[10] umask (user mask, máscara de usuario): Cuando un usuario crea un nuevo fichero, los permisos iniciales con los que se crea están determinados por el valor de la propiedad "umask" del siguiente modo (valores en octal, siendo "& = AND" y "~ = NOT" operadores binarios):

- Ficheros: `Permisos_nuevos_Ficheros = 0666 & ~ (umask)`
- Directorios: `Permisos_nuevos_Directorios = 0777 & ~ (umask)`

La máscara (propiedad umask) *representa así los permisos que **no** deben poseer los nuevos ficheros*. Debe observarse cómo, por cuestiones de seguridad, no se permite que los archivos regulares sean creados por omisión con permisos de ejecución, mientras que los directorios sí.

Ejemplos: si "umask=022":

- Permisos_nuevos_Ficheros = `666 & ~ 022 =` ***644*** # u=rw-,g=r--,o=r--
- Permisos_nuevos_Directorios = `777 & ~ 022 =` ***755*** # u=rwx,g=r-x,o=r-x

Aclaraciones adicionales:

- Se trata de una propiedad (POSIX) del proceso shell actual (no es una variable, por lo que ***no*** es visualizable con "`echo $umask`" o "`set | grep ^umask=`"), siendo heredada por todos los procesos abiertos por dicho shell (scripts hijos, ...). Consecuentemente, puede definirse un valor distinto de "umask" para cada shell.

- Para obtener el valor actual de la propiedad "umask" puede ejecutarse el comando interno "umask". Para modificar su valor, debe usarse dicho comando (consulte su uso en el manual de shell que esté usando).