Impartiendo clase de sistemas operativos con GNU/Linux en ciclos de grado superior de informática

Manuel Sergio Jaime Rodríguez

Autor: Manuel Sergio Jaime Rodríguez
ISBN: 978-1-84799-289-5
Editorial: www.lulu.com
Enlace al libro: http://www.lulu.com/content/1499929

Este libro nace de la necesidad de preparar las clases para un ciclo de grado superior de informática, concretamente el de Administración de Sistemas Informáticos.

He agrupado todos los ficheros de apuntes en formato de presentaciones que he creado para que estén disponibles para cualquiera.

A lo largo de los años que llevo impartiendo este módulo mis alumnos me han ayudado a corregir los errores que me he ido encontrando, pero estoy seguro que usted estimado lector encontrará algunos más. En tal caso ruego que me perdone y que a través del la página de comentarios del libro pueda hacérmelos llegar para que sean corregidos.

Espero que te sea de ayuda.

Índice de contenido

Parte primera: Presentación de GNU/Linux.

Esta parte debería durar unas 2 horas y en ella se debe hacer una presentación de las características de GNU/Linux, GPL y el software libre.

GNU/Linux	MS Windows
No sólo es comandos. Existen entornos gráficos. ¿De informáticos para informáticos? Gratis y estable -> Popular. Grandes empresas. PYMES. Gobiernos. Usuarios: ¿para el informático? ¿para papá y mamá? ¿colegios? Ayuda -> Manuales, Internet,...	Entorno gráfico. No hay posibilidad de cambiarlo. ¿Para informático? ¿tan fácil es? Coste. Grandes empresas. PYMES. Gobiernos. Usuarios: ¿trabajo fiable en profesionales? para casa. colegios. Ayuda: pringao howto.

Tratar de explicar la direfencia entre la filosofía Linux junto con la del Código Abierto frente a la de MicroSoft Windows.

Ventajas para profesionales, estudiantes, empresas, etc...	
Linux.	Windows y otros sistemas.
Código abierto. Posibilidad de estudio del funcionamiento. Genera conocimiento/riqueza local. Modificable. Bajo coste. Se puede copiar -> no es delito = GPL. Seguro. Actualizable rápidamente. Grandes empresas te respaldan. Innovación lenta. Desarrollo dependiente de voluntarios.	Código cerrado -> desconocimiento/incertidumbre. Imposible aprender de su código. No genera conocimiento/riqueza a nivel local y apenas nacional/estatal. No es modificable salvo por los creadores. Coste alto, gran innovación. No se permite copiar, necesidad de comprar más licencias. La instalación de copias sin permiso es delito. Actualizaciones más lentas y no fiables. Desarrollo dinámico -> lucrativo. Muy difundido.

Historia
Unix nace en 1969. Minix y Andrew Tanenbaum. Era se que se era un estudiante que quería crear su propio sistema operativo: Linus Torvalds. Octubre 1991 : primera versión publicada 0.02 con bash, gcc, gnu-make, compress... Gran aceptación. 1992: Linus añade Linux al proyecto GNU -> GPL. Abril 1992: ¡versión 0.96! Aquí vieneeeen las X-Windows (entorno gráfico). Abril 1994: Versión 1.0. Abril 1996: Versión 2.0 y... ¡TUX!

El núcleo

Se compone de 3 números: x.y.z

- X: es el indica la versión principal del núcleo, por lo tanto es el más importante. Desde 1996 está en la versión 2.
- Y: indica mejoras en el núcleo dentro de la actual versión, es un avance pero sin ser lo suficientemente importante. Si es par es versión estable, si es impar inestable o de prueba. La actual es la 2.6. Se le llama: revisión. Por ejemplo, en la 2.2 los ficheros de sonido mp3 en ordenadores lentos podían escucharse con saltos cuando se iniciaba una nueva aplicación, esto fue corregido y en la 2.4 no sucedía.
- Z: indica una mejora de la revisión actual, es decir, de la revisión Y.
- También se usan otras nomenclaturas: pre-U, donde U es una versión no finalizada para la próxima Z; o bien rc (release candidate, versión candidata a estable).

- La versión actual a 11 de diciembre es la 2.6.15.
- La última versión estable (altamente comprobada) es la 2.4.33.
- www.kernel.org
- Mejoras en el 2.6.10: precompilado para 32 microprocesadores y 64 GB de RAM; reparto del microprocesador más equilibrado en servidores; mejora y ampliación de buses firewire, usb e inalámbricas; mejoras en el control de la energía; incorporación de HAL para hotplug en desktop.

Explicar como se identifican las versiones del núcleo y donde encontrarlo.

Posibilidades del sistema
• Multitarea: RR. • Multiusuario, y además sin necesidad de adquirir nuevas licencias para nuevos usuarios... sin comentarios :-). • Multiplataforma. Podemos encontrar GNU/Linux en videoconsolas GP2X (www.gbax.com) , terminales teléfonicos (Nokia y Symbian), compatibles X86, Sparc, Motorola, ... y sigue como el conejito de duracell. • Shell programable lo que hace su línea de comandos muy versatil y flexible. • Independencia del dispositivo lo que permite la conexión de cualquier dispositivo y en cualquier número. • Linux y su hermano mayor Unix son los SS.OO. que soportan la mayor cantidad de tipos distintas de conexiones y sin colapsarse. • Permite la ejecución de hebras, desde siempre. Ejecuta los programas usando memoria compartida y virtual de forma que un programa no tiene porqué estar cargado por completo en memoria para se ejecutado. • Soporta un gran número de sistemas de ficheros, lo cual le posibilita usar el que más convenga en cada momento: los de MS, el de las carísimas estaciones de procesamiento de gráficos de Silicon Graphics, el de IBM para los grandes sistemas de datos.

Información y docuemntación
• www.debian.org/international/Spanish.es.html • Toda la documentación de Linux: www.tldp.org , y es.tdlp.org para hispanohablantes. • www.insflug.org traducción de documentos breves. • Howto y minihowto. • www.escomposlinux.org • www.linux-es.com • www.hispalinux.es • www.linux.org • Revistas: • www.linuxgazette.com ---> mejor en cristiano: www.gacetadelinux.com • Linux focus: www.tldp.org/linuxfocus/castellano • mundolinux, dlinux, todolinux... • La web es grande... muy grande, pero está Google :-) FIN de la teoría, ahora tocan... ¡las curvas!

¿Qué vamos a aprender durante el curso con: Debian/Ubuntu/Guadalinex?	
• Instalación del sistema. • Entorno gráfico: • Entorno gráfico, menús: aplicaciones, lugares y sistema. • Configuración de las X windows. • Configuración de la impresora. • Primeros pasos en Debian. • Inicio del sistema: GRUB e inittab. • Sistema de ficheros. • Permisos y Gestión de usuarios. • Instalación/desinstalación de paquetes y actualización del sistema. • La shell Bash y configuración de red de Linux. • Shell bash.	• Comandos. • Configuración de la red TCP/IP. • Administración avanzada. • Copias de Seguridad. • Logs del sistema. • Tareas Programadas. • Administración remota con Webmin. • Configuración del disco duro. • Seguridad en sistemas Linux. • Critptografía del sistema de ficheros. • Blindaje del sistema. • Vulnerabilidades del sistema. • Introducción al análisis forense. • Antivirus.

Parte segunda: Configuración y uso del sistema desde el entorno gráfico.

Esta sesión debería durar unas 2 horas y en ella se debe hacer una presentación de las características de GNU/Linux, GPL y el software libre.

Configuración del entorno gráfico de Debian/Ubuntu/Guadalinex

- ➔ Antes de empezar: ¿donde puedo encontrar más información?
  http://www.guia-ubuntu.org/hoary/doku.php
  http://es.tldp.org/
  Entorno gráfico Gnome en Guadalinex/Ubuntu:
- ➔ Menús: Aplicaciones, Lugares y Sistema. (ver a fondo).
- ➔ Para pedir la contraseña de superusuario: sudo passwd root.
- ➔ Añadiendo más aplicaciones: Sistema -> Administración -> Añadir programas.
- ➔ Configuración de las X windows
- ➔ ¿Soporta X windows mi tarjeta gráfica? http://www.xfree86.org/current/manindex4.html
- ➔ Ficheros de configuración en: /usr/X11R6 y /etc/X11.
- ➔ Configurando las Xs a pelo : xf86config (para expertos) o dpkg-reconfigure xserver-xfree86, o xf86cfg.
- ➔ Fichero de configuración de las Xs: /etc/X11/XF86Config-4 (saca copia de seguridad antes si vas a modificarlo).
- ➔ Secciones del XF86Config-4: Files (fuentes), ServerFlags (casi sin uso, configuración de opciones globales), InputDevice (teclado, ratón,... todos los dispositivos de entrada), Monitor, Device (tarjeta gráfica) y Screen (resolución y profundidad de color del monitor).
- ➔ Ccnfiguración de la aceleración 3D: ati y nvidia en sus páginas respectivas.
- ➔ Efectos 3D: Compiz Fusión. Esto si son efectos y no los de Windows.

Configuración de la impresora.
• Tres tipos de impresoras según el lenguaje que usan para imprimir. • Lenguaje Postscript (PS): ¿cómo imprimir la página?; o su propio lenguaje. • Winprinters: difíciles de configurar en Linux o imposible. • Herramientas para configurar la impresora. • foomatic-gui (no disponible en GLX-V3). • gnome-cups-manager (también en el entorno gráfico). • CUPS: una gran base de datos de impresora y un entorno web para su gestión. • Local o en red. • Unix Printer (lpd): impresora en una máquina Linux/Unix que usa, y está disponible el demonio lpd. • Impresoras en Windows (SMB): para poder usar las impresoras de una máquina que usa Windows, el ordenador cliente debe tener instalado SAMBA (www.samba.org). Se recomienda la lectura de: http://es.tldp.org/Manuales-LuCAS/USANDO-SAMBA/usando-samba-html/ • CUPS: impresora en una máquina Linux/Unix que está ejecutando el demonio de uso y configuración de impresoras CUPS. • Ficheros. • En DOS el puerto paralelo era lpt1, en Linux es el fichero lp0, lp1,... en el directorio /dev • En /etc/cups/printers.conf se puede encontrar la información relativa a la impresoras instaladas en el sistema, ya sean en red o locales. • Ordenes básicas de impresion. • /etc/init.d/cupsd star, stop, restart

Parte tercera: El inicio del sistema.

En esta parte vamos a estudiar cómo se inicia el sistema, desde la carga del gestor de arranque, la decisión de qué opción y cómo comienza la carga del núcleo.

El inicio del sistema en GNU/Linux

Los ordenadores utilizan, independientemente del SSOO que tengan, el mismo método para iniciar el SSOO:
a) Control del hardware.
b) Cargador de inicialización, búsqueda del SSOO. en el MBR.
c) Inicio de carga del SO. c) Comienzo de los procesos propios del SO.

Pero los SSOO no son iguales y Linux es muy diferente a lo que conocéis y eso se pone de manifiesto en la gran cantidad de opciones de inicio.

Conceptos:
a) MBR (Master Boot Record).
b) GRUB: gestor de arranque. Opciones: e para modificar parámetros y c para acceder a la interfaz de comandos..
c) Tratamiento de los discos y las particiones en Linux:

c1) Linux. Primer disco hdXY, donde X es una letra que comienza en a e Y es el número de la partición. Así hda1 corresponde al primer disco, maestro del primer canal IDE, primera partición; hdb3 corresponde al disco esclavo del primer canal IDE y la tercera partición; hdc5 corresponde a... ¿? Si el disco es SCSII: sdXY.

c2) GRUB: (hdX,Y), teniendo en cuenta que tanto X como Y son un número >= 0. 0 corresponde al primer disco o primera partición, 1 corresponde al segundo disco o segunda partición.
d) El fichero de configuración de GRUB se encuentra en: /boot/grub/menu.lst
e) El MBR se compone de: 446 Bytes donde se guarda el programa gestor de arranque.
64 Bytes para la tabla de particiones.
- 2 Bytes para la cifra mágica: AA55h. Si no está almacenado ese valor en el MBR se ejecuta la interrupción 18h para informar del error.

Configuración de GRUB: el fichero **menu.lst**.

/boot/grub/menu.lst
- Opción por defecto comenzando en 0: default=0.
- Tiempo de espera en segundos: timeout=10.
- Si falla la opción por defecto prueba con otra: fallback=1 2 3
- Gráfico de fondo imagen 14 colores. splashimage=(hd0,0)/boot/grub/splash.xpm.gz
- Opción: Title Guadalinex.
- Partición de arranque: root (hd0,5).
- Núcleo a cargar y opciones: kernel /boot/vmlinuz-2.6.12-9-k7 root=/dev/hda9 ro auto quiet splash (splash=silent en núcleos anteriores al 2.6)
- Por si el núcleo necesita módulos especiales de arranque: /boot/initrd.img-2.6.12-9-k7
- Orden de arrancar: boot.
- Opción por defecto: savedefault

Ejemplo de grub para cargar Windows:
- La línea también comienza igual que para elegir un Linux: Title Windows XP.
- Establecer la partición de arranque sin montarla: rootnoverify (hd0,0)
 o bien montándola: root (hd0,0).
- Orden de carga del primer sector de la partición donde hemos indicado que está Windows: chainloader +1

GRUB en disquetes

Pon un disquete y como root :

dd if=/boot/grub/stage1 of=/dev/fd0 bs=512 count=1

dd if=/boot/grub/stage2 of=/dev/fd0 bs=512 seek=1

Anota en papel los ficheros que influyen en el arranque de Linux: root, kernel, initrd

Reinicia el ordenador con el disquete dentro y...

grub> root (hd1,0)

grub> kernel /boot/vmlinuz-2.6.9-12-k7 root=/dev/hda9 ro quiet splash

grub> initrd /boot/initrd.img-2.6.8.1-3-386

grub> boot

Creación de un disco de arranque de emergencia con el gestor de arranque completo (sólo para el mismo ord.) (debes hacerlo como **root**).

Formatear el disco en ext2 : mke2fs /dev/fd0

Creación del directorio del disquete **si no existe:** mkdir /media/floppy

Se monta el disquete: mount -t ext2 /dev/fd0 /media/floppy

Ahora se instala en el disquete: grub-install --root-directory=/media/floppy fd0

Creación de la estructura de directorios (dentro del floppy en /boot/grub) y copia de los ficheros ***si no se ha efectuado***

cp /boot/grub/stage* /media/floppy/boot/grub

cp /boot/grub/menu.lst /media/floppy/boot/grub

Modifica una entrada del menu.lst (por ejemplo el de Windows) para comprobar que lo hemos hecho bien: gedit /media/floppy/boot/grub/menu.lst

Y se desmonta el disquete : umount /media/floppy

Recuperación del gestor de arranque.

Supongamos que hemos perdido GRUB tras la instalación de Windows... ¿cómo lo recuperamos? Si GRUB no se instaló bien, podemos instalarlo GRUB como sigue:

*** otras formas: disco inst SUSE, grub4win.

1. Iniciamos el ordenador desde el lector de CDs con el CD de la distribución.
2. Cuando se haya cargado el SO, hay que abrir un terminal de línea de comandos
3. En esa terminal debemos ejecutar los comandos siguientes:

a) Vamos a trabajar como administrador

$su

password: xxxxxx

o bien sudo antes de los comandos siguientes...

b) Vamos a crear un directorio en el que montar el Linux de nuestro disco duro y lo montamos:

#mkdir sysimage (por ejemplo hazlo dentro de /media)

#mount -t ext3 /dev/hda2 sysimage

OJO: Suponemos que el disco con el que estás trabajando es el maestro del primer canal IDE y que Linux está instalado en la segunda partición primaria (hda2). Si no es así tendremos que ajustarlo a nuestro sistema.

#chroot sysimage

Nos permite trabajar como si el árbol de ficheros partiera de ese directorio, se trata del directorio en donde se monta nuestro sistema Linux.

4. Con el editor gedit podemos modificar el fichero /boot/grub/menu.lst hasta ajustarlo a nuestra máquina, una vez que se adecúe a nuestro sistema ejecutaremos:

#grub-install /dev/hda

5. Para terminar sólo tenemos que ejecutar el comando exit (un par de veces).

GRUB: seguridad y otras opciones.

Contraseña para GRUB.
- Localiza la línea: #password topsecret
donde topsecret es la contraseña por defecto
- Codificando la contraseña de GRUB:
a) Desde la consola de GRUB:
grub> md5crypt que devuelve una cadena encriptada
password: **********
'1ddTCc1$8v3fWFR4m5kDfuRG5LUHo/'
grub> quit nos devuelve a la línea de comandos.
que debemos copiar a la línea de menu.lst quedando:
password --md5 1ddTCc1$8v3fWFR4m5kDfuRG5LUHo/
b) Desde la línea de comandos:
/sbin/grub-md5-crypt que devuelve la cadena encriptada
A partir de ahora si quisieras modificar grub durante el arranque tendrás que introducir p para cambiar los parámetros.
Si deseas restringir una opción del menú basta con poner la línea de la contraseña bajo la del título.
Edición desde grub: c para acceder a la interfaz de línea de comandos.
grub> help
grub> displaymem
grub> cat (hd0,0)/autoexec.bat
grub> cat (hd0,1)/boot/grub/menu.lst
grub> geometry (hd0)
grub> color blue/light-gray yellow/red
Elimina GRUB desde DOS/Windows con: fdisk /mbr

GRUB: configuración de la imagen de fondo de menú

Vamos a ver cómo poner una imagen de fondo en el menú de inicio de GRUB. Escoge una imagen y comienza.

a) Situémonos en la ruta en donde está la imagen

```
# cd /usr/share/pixmaps/guadalinex
```

b) Convirtamos la imagen al formato deseado (formato xpm de dimensiones 640x480 y con una profundidad de 14 colores)

```
# convert -resize 640x480 -colors 14 guadalinex-background.png fondo.xpm
```

c) La imagen la comprimiremos en formato gz.

```
# gzip fondo.xpm
```

d) Movamos el fichero obtenido a donde debe estar

```
# mv fondo.xpm.gz /boot/grub
```

e) Modifiquemos el fichero de configuración de grub de forma que cargue la imagen en el inicio. Pero cuidado que la partición del disco en que está la imagen habrá que adecuarla a nuestro sistema:

```
# gráfico de fondo por defecto
splashimage=(hd0,1)/boot/grub/fondo.xpm.gz
```

Ejercicios de GRUB

1) ¿Dónde se instala GRUB? Dí todo lo que sepas. ¿Qué es: stage1, stage2 y AA55?
2) Quita las opciones de arranque silent y vga ¿qué ocurre?
3) Mira las opciones default y savedefault. ¿Están relacionadas? Prueba a que default no apunte a savedefault. Explica lo que pasa.
4) Prepara tu GRUB para que si falla el arranque de un kernel lo intente desde otro.
5) Aparte de vga=791 ... ¿qué más opciones tiene vga? Haz un cuadro de referencia.
6) Haz dos discos de instalación de GRUB, uno desde Linux y otro desde ¿Windows? Explica lo que pasa.
7) Bájate de Internet dos imágenes de fondo de GRUB y cambia la imagen que aparece en el gestor de arranque.
8) Crea una nueva entrada al GRUB y ponle una contraseña.
9) Haz una recopilación de diferentes sitios webs donde encontrar información sobre cómo utilizar GRUB.
10) Baja grub4win. Instálalo y prueba. ¿Te hace falta una copia de la configuración de GRUB?
11)

El fichero initt (I)

Inicio del sistema: /etc/inittab
a) # Nivel de ejecución por defecto.
id:2:initdefault:
b) # Script de configuración/inicialización en el momento de arranque. Esto se ejecuta primero excepto cuando se inicia en modo de emergencia (-b).
si::sysinit:/etc/init.d/rcS
c) # Qué hacer en modo de single-user.
~~:S:wait:/sbin/sulogin
d) # /etc/init.d ejecuta los scripts S y K una vez ha cambiado el nivel de ejecución.
e) # Nivel de ejecución 0 es parar.
Nivel de ejecución 1 es monousuario.
Nivel de ejecución 2-5 son multiusuario.
Nivel de ejecución 6 es reinicio.
l0:0:wait:/etc/init.d/rc 0..6 (7 en total)
g) Reiniciar el login si se pierde
z6:6:respawn:/sbin/sulogin
h) ¿Qué hacer cuando se presiona CNTRL+ALT+DEL?
ca:12345:ctrlaltdel:/sbin/shutdown -t1 -a -r now
i) ¿Qué hacer cuando hay la energía falla/regresa?
pf::powerwait:/etc/init.d/powerfail start
pn::powerfailnow:/etc/init.d/powerfail now
po::powerokwait:/etc/init.d/powerfail stop

El fichero initt (I)

j) Consolas disponibles (CNTRL+ALT+F1 a F7) La F7 está destinada a la consola gráfica si usas Xs.

```
1:2345:respawn:/sbin/getty 38400 tty1
2:23:respawn:/sbin/getty 38400 tty2
```

k) Terminal en el puerto serie:

```
T0:23:respawn:/sbin/getty -L ttyS0 9600 vt100
```

l) Cómo configurar una línea de modem

```
T3:23:respawn:/sbin/mgetty -x0 -s 57600 ttyS3
```

- Todos los scrips ejecutados, o no, por init están en: /etc/init.d
- El primer script en ejecutarse es: /etc/init.d/rcS
- Nomenclatura de los ficheros: SXXnombre o KYYnombre. 'S' indica que es un servicio que se ejecutará al encender el ordenador (Start). 'K' indica que es un servicio que se ejecutará al dar la orden de apagado (Kill). XX e YY son números de dos cifras que indican el orden de ejecución. Se inician de menor a mayor.

- Cambiando el nivel de ejecución:

telinit n o bien # init n

- ¿Qué hacen #init 0 o #init 6 ?
- ¿Puede GRUB cambiar el nivel de arranque?

Aceleración del arranque (I)

Linux no fue pensado como sistema de escritorio.

1º /etc/inittab.
No son necesarias todas las consolas. La única imprescindible: 7.
2º Guadalinex=Ubuntu. (sólo probado para estas dos distribuciones)
/etc/readahead contiene una lista de los archivos que el sistema va a usar durante el arranque. Si cambia algo en el GRUB tocar la línea del kernel y añadir sólo por una vez: **profile**. La primera vez tardará un poco más, pero si no hay más cambios el arranque será algo más rápido en los siguientes arranques.
Pero la carga de los ficheros se realiza en paralelo y en sí misma podría relentizar el arranque. Así que se hace necesario cambiar la forma en la que funciona el script de carga de **readahead**. Para ello entramos en el fichero **/etc/rcS.d/S01readahead** y buscamos la línea donde indica que carga en background:
if /sbin/start-stop-daemon --start --quiet background
y le borramos ***--background*** .
3º Entrar con un usuario por defecto ***¡cuidado!***: **gdmsetup**.
Y varias cosas más...

Aceleración del arranque (II)

Todas las distribuciones, salvo Gentoo, introducen muchos servicios que no son necesarios. Pero el sistema nos proporciona diversas formas de solucionarlo.

4º Con el programa **services-admin** podemos desactivar y activar algunos servicios al arranque.
También se puede encontrar en el menú de Gnome: Sistema->Administración->Servicios. Ubuntu también trae una herramienta parecida: bum.
O podemos hacerlo a mano...
5º Pero el punto 4º no te permite des/habilitar todo, así que habrá que hacerlo a mano.
Debian, y por extensión sus hijos (Ubuntu, Linex, Guadalinex, etc...) tienen un comando que permite habilitar o deshabilitar el arranque de algunos programas. ¿Cómo funciona? míralo en **man update-rc.d**.
Ejemplo: **/etc/init.d/update-rc.d -f eagle-usb remove**.
desactiva la carga y sin el parámetro -f hace que se cargue.
Alguno otro programa/script que quizás te convenga deshabilitar: powernowd (útil sólo para portátiles), lmsensors (útil para ver la temperatura del ordenador), pcmcia (portátiles), etc... Si tienes dudas: man nombre-servicio.
Esto mismo lo puedes hacer a mano renombrando un servicio que empieza SXXnombre por KXXnombre, es decir, cambiando la **S** por la **K**.
6º También puedes cambiar la configuración en ***/etc/default*** ,
pero la opción 5 es más segura**.**

Ejercicio

1) Localiza sitios en Internet que sean útiles para aprender más sobre inittab.
2) ¿De cuantas formas puedes iniciar el sistema (nivel de ejecución)?
3) Apaga el sistema modificando el nivel de ejecución.
4) ¿Qué pasa si pulsas CONTROL+ALT+DEL? ¿Puedes modificar el comportamiento del sistema para que cuando pulses esa combinación de teclas la ejecución del comando se lleve a cabo en 10 minutos? Si no sabes cómo hacerlo busca información con info, man o por Internet.
5) ¿Cuantos consolas hay disponibles? ¿Las necesitas todas? ¿Qué pasaría si quitas varias o todas? ¿Existe alguna imprescindible?
6) ¿Cómo se diferencian los scripts de arranque y los de parada? ¿Cómo sabes en que orden y nivel de ejecución de inician/paran los servicios (programas)? ¿Se pueden modificar? ¿Cómo?
7) ¿Cómo cambiamos de un nivel de ejecución a otro? ¿Para qué puede servir?
8) Modifica GRUB y crea una entrada nueva para entrar al sistema en modo mono-usuario. Hazlo de las distintas formas que sabes. ¿Qué ocurre?
9) ¿Qué debería ocurrir si dejas permanentemente profile como parámetro del kernel en el GRUB?
10) Desactiva el servicio **networking**. Reinicia y comprueba si tienes red. Vuelve a activar el servicio. Haz lo mismo con **netapplet** y **hotplug**, pero antes mira para que sirven cada uno.
11) ¿Cómo puedes cambiar el orden de arranque de un programa? ¿Para qué sirve el programa *ksysv* ?
12) ¿Qué es y cómo activas XDMCP?

Parte cuarta: El sistema de ficheros.

Estudiaremos en esta parte los distintos sistemas de ficheros existentes en Linux así como sus diferencias con los de otros sistemas operativos..

Características (I)

1) Para usar el sistema de ficheros primero necesitamos particionar el disco.
2) Crear el sistema de ficheros: crear las estructuras de datos que el sistema necesita para contener los ficheros y directorios.
3) ¿Pueden existir dos o más tipos de sistemas de ficheros diferentes en el mismo disco duro?
4) Los sistemas de ficheros de Windows y Linux son diferentes. Cuidado con las tildes.
5) Características de los sistemas de ficheros Linux/Unix:
    - Linux es "case sensitive" es decir que diferencia entre mayúsculas y minúsculas.
    - La barra que delimita los caminos de los directorios es / , en los sistemas de MS es \ .
    - Admiten hasta 255 caracteres en el nombre de los ficheros.
    - Estructura del sistema de ficheros en modo árbol.
6) En Linux existen cuatro tipo de ficheros diferentes:
    - Ficheros normales.
    - Directorios.
    - Enlaces.
7) Archivos especiales: los ficheros de dispositivos que se encuentran en el directorio /dev o ficheros especiales fifo o tuberías de nombre que no veremos.
8) La mayoría de los sistemas de ficheros de los *ix se organizan de forma parecida, varían en la implementación que se afectará a la eficiencia.
9) Sistemas de ficheros de Linux:
    - Ext → ext2 → ext3.
    - MSDOS → vfat → ntfs. umsdos (uso de nombres largos desde el sistema de ficheros Linux).
    - ISO9660.
    - hpfs: el sistema de ficheros de ya extinto OS/2.h
    - nfs: sistema de ficheros en red. SUN.
    - minix: padre de ext.
    - Reiserfs.

Los directorios Unix/Linux
/bin : contiene los comandos básicos del sistema. /sbin : contiene comnandos esenciales para la administración del sistema. /boot : todo lo necesario para el arranque del sistema: binarios, ficheros de configuración del gestor de arranque, imagen del kernel... /dev : ficheros de dispositivos. /etc : ficheros de configuración programas del sistema. /home : directorios de trabajo de los usuarios. /root : directorio de trabajo del usuario root. /lib : librerías básicas para trabajar en Linux. /mnt : directorio donde se montan los sistemas de ficheros: discos windows, floppys, unidades de red. En Ubuntu el montaje se realiza en /media y en Guadalinex en el directorio principal los extraibles. /proc : no es en realidad un directorio físico. Su información se crea en tiempo de ejecución. Permite acceder a información del kernel. En /proc/filesystems encontramos los distintos sistemas de ficheros soportados por el kernel en ejecución. Los procesos en ejecución pueden verse a través del comando ls -l /proc. /tmp : información temporal. /usr : programas que no forman parte del sistema básico. /usr/bin tiene programas pero no son básicos. /usr/sbin tiene programas que no son tan esenciales para el funcionamiento del sistema. /var : ficheros de datos variables, información (logs) del sistema, datos administrativos... lost+found: www.pathname.com/fhs Destacar que según que distribución se use el número de directorios en la raiz del sistema de ficheros puede variar.

Ficheros de configuración del sistema (I)

Los directorios de usuario “/home/manolo” contienen los ficheros de configuración de cada uno de ellos. Estos empiezan por . . Los básicos los crea el sistema en el momento de creación del usuario, otros (de programas) se crearán más adela

En /etc y sus subdirectorios se encuentran la mayoría de los ficheros de configuración del sistema, veamos de qué se encargan algunos:

- bash.bashrc : valores por defecto para todo el sistema para la shell bash.
- exports : lista de directorios locales a ser compartidos por el sistema NFS (Network File System).
- fstab : contiene los diveros sistemas de ficheros a los que se puede acceder y las ubicaciones dentro del sistema de fichero Linux donde se montan.
- mtab : lista de sistema de ficheros que están montandos en ese momento.
- group : almacena los nombres y los id de los grupos existentes en el sistema.
- host.conf : contiene las direcciones donde se buscan las localizaciones de dominio.
- hosts: lista de direcciones IP y sus correspondientes nombres de dominio.
- hosts.allow / host.deny : contiene los nombres de los ordenadores autorizados (allow) / no autorizados (deny) a acceder al sistema.
- hostname: contiene el nombre netbios del sistema.

Ficheros de configuración del sistema (I)

- inittab : este fichero permite ajustar el arranque del sistema para que ejecute según nuestros intereses.
- manpath.config : fichero de configuración del comando man.
- modules.conf : alias y opciones para los módulos que puede cargar el kernel.
- passwd : contiene la información relativa a cada usuario: directorio home, grupo de referencia al que pertenece. Clave encriptada...
- printcap : contiene las impresoras configuradas en el sistema.
- profile : configura el entorno de trabajo, tiene validez para todo el sistema, programas, entorno y usuarios.
- resolv.conf : contiene las DNS de nuestro proveedor de internet.
- shadow : contiene las contraseñas encriptadas de los usuarios con permiso de acceso al sistema.
- sudoers : fichero de configuración del comando sudo.
- syslog.conf : recoge qué mensajes de inicio de sesión guarda el demonio syslogd y qué ficheros se almacenan en él.

/etc
• X11: configuración de las Xs y de los diferentes gestores de ventanas instalados. • apt: contiene la configuración de Synaptic y el sources.list. Único de los sistemas Debian. • cron.*: contiene distintos directorios referidos a la forma de ejecutar el demonio “crond” de ejecución temporizada. • default: contiene ficheros que definen valores por defecto para distintas aplicaciones, por ejemplo useradd: directorio de inicio, número del grupo, fecha de caducidad de la contraseña, shell, directorio esqueleto... • init.d: contiene los guiones de los distintos niveles de ejecución. Vinculado a /etc/rc?.d. • ppp: ficheros de configuración para la conexión a internet a través de modem. • rc?.d: 7 directorios que determinan los scripts, y el orden de estos para ejecutarse en los distintos niveles de ejecución. • security: condiciones de seguridad por defecto. • skel: contiene el esquema de directorios que se crean por defecto cuando se añade un nuevo usuario. • network: contiene la información relativa a la configuración de las tarjetas de red. • Cualquier aplicación instalada debe tener aquí sus datos de configuración.

/usr
Es el mayor sistema de datos de la jerarquía. Es compartido en modo de sólo lectura . - Cómo mínimo debe tener estos subdirectorios: a) bin: la mayoría de los comandos de usuario. b) include: cabeceras de las librerías C. c) lib: librerías. d) local: jerarquía local (vacía después de la instalación principal). e) sbin: programas del sistema que no son vitales para el funcionamiento. f) share: datos independientes de la arquitectura. Almacena programas accesibles por todos los usuarios, Temas de escritorio, iconos... g) src: almacena los ficheros fuente de los programas descargados, por ejemplo, el núcleo.

Creación de un sistema de ficheros

- Para crear un sistema de ficheros se usa el comando *mkfs.*

mkfs -V -t tipofs dispositivo -c -v

-V : produce una salida detallada. Útil para encontrar errores.

-t tipofs: indica el tipo de sistema de ficheros a crear: ext2, ext3, minix, msdos, vfat, xfs, xiafs.

-c : hace una comprobación de posibles bloques defectuosos antes de la construcción del sistema de ficheros.

-v=-V.

dispositivo es un fichero asociado a una unidad física de disco: /dev/hda, /dev/fd0,...

- mkfs.ext2, mkfs.ext3 (mke2fs), mkfs.minix, mkfs.msdos, mkfs.vfat, mkfs.xfs, mkfs.xiafs, mkdosfs.

**** Históricamente los discos, locales o en red, se han montado en el directorio */mnt*, si bien, últimamente podemos encontrarnos distribuciones Linux que cambian ese directorio y lo sitúan en */media* o el directorio raiz en un intento quizás de parecerse más a otro sistema operativo.

Otros comandos relacionados con la manipulación del sistema de ficheros:

- mount /dev/*dispositivo* dir-punto-de-montaje: monta un disco en un directorio.
- umount dir-punto-de-montaje: desmonta un disco.

* tanto en mount como umount si el dispositivo está en el fichero */etc/fstab* se puede obviar su inclusión.

- fdisk /dev/*dispositivo*: permite realizar acciones variadas sobre el dispositivo de disco.
- fsck /dev/*dispositivo*: permite comprobar y opcionalmente recuperar errores de un sistema de ficheros.
- ln: crea un enlace (acceso directo) a un directorio.

No se puede desmontar un sistema de ficheros si está en uso o se está situado dentro de él.

/etc/fstab (I)

El fichero *fstab* tiene información descriptiva sobre los distintos sistemas de ficheros que se encuentran disponibles en el equipo. El orden es importante porque mount, umount y fsck actuan secuencialmente sobre él. Su formato es el siguiente:

dispositivo directorio tipo opciones frecuencia secuencia

- dispositivo: dispositivo local o remoto de sistema del ficheros a montar.
- directorio: directorio donde se montará. Para particiones *swap* este campo estará a : ***none***.
- tipo: es el formato del sistema de ficheros. Con la opción *auto* se le indica que lo intente autodetectar.
- opciones:
    a) auto/noauto: la partición se monta/no se monta al arrancar.
    b) user/nousers: permite/no permite a los usuarios montar la partición. El usuario root puede desmontar y montar a placer.
    c) users: permite a los usuarios desmontar la partición aunque no la hayan montado ellos.
    d) ro/rw: montaje en solo lectura o en lectura-escritura.
    e) exec: permite ejecución de programas.
    f) async: el sistema seguirá trabajando sin esperar la confirmación de escritura.
    g) defaults = rw, exec, auto, nousers, async.

/etc/fstab (II)

- frecuencia: es un número que determina la periodicidad de las copias de seguridad con el comando dump. Las copias de seguridad las veremos en unos días. 0 (cero) indica ninguna.
- secuencia: es un número que determina el orden en que *fsck* comprueba la integridad del sistema. Los discos físicos distintos deberían tener números distintos. Los discos contenidos dentro de un mismo disco físico (unidade lógicas=particiones) deberían tener números consecutivos según la importancia. El sistema de ficheros raiz (/) debería ser 1. 0 indica que no hay comprobación.

*** Los sistemas de ficheros que se encuentran montados en este momento se anotan en el fichero: */etc/mtab.*
En los enlaces simbólicos, los permisos de los ficheros que prevalecen son los del fichero apuntado .

Enlaces: accesos directos

El sistema de ficheros de Unix permite hacer enlaces, en otros sistemas se conocen como Accesos Directos. El comando es: *ln*.

Existen dos tipos de enlaces: ***duros*** y ***blandos***.

a) Los "duros" son como una copia, ya que de la existencia del fichero enlazado no depende la existencia el enlace. Por ejemplo supongamos que queremos hacer un enlace duro al fichero pepe y llamarlo pepe-enlazado . Si borramos pepe , podemos seguir accediendo al fichero a través de pepe-enlazadas ya que lo que se hace es una referencia. No se pueden realizar enlaces ***duros*** entre ficheros de dos sistemas distintos, y tampoco entre directorios. La forma de crearlo es:

ln /caminocompleto/pepe /caminocompletodestino/pepe-enlazado

b) Los "blandos" o simbólicos son simplemente un puntero al verdadero fichero. Es posible establecer enlaces simbólicos entre ficheros de distintos sistemas de ficheros, directorios, e incluso entre ficheros que no existen. Se crean con la opción *-s*, ejemplo:

ln -s /caminocompleto/pepe /caminocompletodestino/pepe-enlazado

Ejercicios
1) Enuncia las diferencias entre los sistemas de ficheros Linux y los de MS. 2) Respecto a los directorios que cuelgan de la raíz: a) ¿Dónde puedes encontrar ficheros de sistema? b) ¿Dónde están los programas? c) ¿Dónde están los ficheros de configuración de las Xs? d) ¿Qué puedes encontrar en /var? e) ¿Puede variar la organización de los directorios dependiendo de la distribución que uses? ¿Cuáles son los mínimos comunes? f) ¿Para qué puede servirte el directorio /proc y su contenido? g) ¿Qué utilidad puedes darle a lost+found? 3) Explica brevemente para qué sirven y donde puedes encontrar cada uno de los siguientes ficheros de configuración: exports, fstab, mtab, group, hostname, host.deny, host.allow, passwd, inittab, profile, resolv.conf, shadow, modules.conf. 4) Explica para qué sirven los directorios: cron.*, apt, X11, security. 5) Busca información sobre las distintas opciones que tiene el comando que comprueba la integridad del sistema de ficheros. Describe los códigos de salida que aparecen en la ayuda. Usa el comando con la opción de comprobación interactiva, es decir, que pregunte antes de llevar a cabo una reparación. 6) Escribe la información de la tabla de particiones referente al disco duro, físico, no la partición, donde estás.

Ejercicios (continuación)

7) ¿En qué situaciones no se pueden desmontar los sistemas de ficheros? ¿Qué comando se utiliza para montar/desmontar un sistema de ficheros? ¿Cómo puedo saber qué procesos están usando un sistema de ficheros determinado?
8) Crea un sistema de ficheros ext2 en un disquete desde línea de comandos. Monta el disquete y copia en él algo. Comprueba que todo ha salido bien montándolo de nuevo y visualizando su contenido.
9) Lee la página del manual referente a *fsck*. Comprueba la integridad del sistema de ficheros del disquete del punto anterior añadiendo una opción en la cual no te pida confirmación. Utiliza también *fsck* para comprobar los sistemas de ficheros existentes en /etc/*fstab,* hay una opción para ello, lee el manual. ¿Qué hace la opción -y?
10) Cambia el nombre y directorio con el cual el sistema se refiere a los discos de Windows por un nombre que describa más su sistema de ficheros y su contenido. Modifica el orden en el cual se comprueba la integridad del sistema de ficheros.
11) Crea un enlace en el escritorio a: cada unidad de windows y al pendrive. Como root crea otro enlace a el fichero /etc/fstab de forma que si se borrase el fichero origen el fichero no se pierda. ¿De qué tipo de enlaces estamos hablando en cada caso? Justifica tu respuesta. ¿Puedes hacer un enlace al directorio Mis Documentos de Windows.

Parte quinta: permisos y gestión de usuarios.

Durante este tema llevaremos a cabo la iniciación a la gestión de los usuarios del sistema y de los permisos de los ficheros.

Introducción

Linux es multitarea y multiusuario, lo cual indica que varios usuarios pueden estar a la vez en el sistema ejecutando tareas simultáneamente, por lo tanto es necesario identificar los usuarios y las acciones que pueden llevar a cabo, y esto se hace a través de las cuentas de usuarios y los grupos. La identidad del usuario junto al grupo/s al/a los que pertenece determina los derechos de acceso a los ficheros y los privilegios.

En los sistemas Unix, un fichero tiene un dueño (normalmente el usuario que lo creó) y un grupo al que pertenece. Partiendo de esta estructura el sistema asigna **permisos** para: **el propietario, el grupo al que pertenece, y a resto de usuarios**. Además, para cada uno de los tres anteriores existen otros tres tipos de permisos básicos: **lectura (r), escritura (w) y ejecución (x).**
Para poder ver estos permisos, el propietario y el grupo del fichero, desde la consola: ls -l .
Ejercicio: ¿Qué información proporciona la opción -l (ele)?
¿Cómo podemos cambiar los permisos? ¿Y el grupo y/o usuario al que pertenece un fichero? ¿Cómo añadimos/borramos/cambiamos usuarios y grupos? Veámoslo con más detalle...

Permisos (I)

El comando: ls -l /etc/fstab produce la siguiente salida:

-rw-r--r-- 1 root root 840 2006-01-31 11:32 /etc/fstab

El primer guión por la izquierda indica el que el fichero es normal. Posibilidades para ese primer caracter:

d : indica que es un directorio.

l : un enlace simbólico.

c : un dispositivo de caracteres.

b : un dispositivo de bloques.

p : canalización con nombre (lo vemos más adelante).

¿rwxr--r-- ? Estos permisos se agrupan de tres en tres: tres para el usuario propietario, tres para el grupo al que pertenece el propietario y otros tres para el resto. En este caso:

rwx : indica que el dueño tiene permiso para leer, escribir y ejecutar el archivo.

r-- : indica que el grupo del dueño tiene permiso para leer, pero no puede ni modificar ni ejecutar el archivo.

r-- : indica que el resto de los usuarios del sistema tiene permiso para leer, pero no puede ni modificar ni ejecutar el archivo.

Permisos (II)

Dando un poco más de detalle sobre los permisos diremos que **cuando se refieren a un directorio**:

r : permite leer el contenido del directorio, pero no garantiza la lectura de los ficheros. Permite ls , pero no ls -l .

w : permite escribir/borrar en el directorio ficheros y otros subdirectorios.

x : permite entrar en el directorio y usar su información: ejecutar, ls -l,..

******* Los permisos de un fichero/directorio están condicionados por los permisos del directorio donde reside. Ejemplo. Sea el directorio .../restringido en algún lugar del disco duro con los permisos **rw-rw-rw-** , y sea el fichero ejecutamesipuedes con los permisos **rwxrwxrwx**. **El fichero no se podrá ejecutar** porque **el directorio no lo acepta**. Con esto se consigue un nivel más de seguridad.

Comandos para la manipulación de permisos (I)

- chown : cambio de propietario de un fichero/directorio. Formato: chown usuario fichero.
- chgrp : cambio de grupo de un fichero/directorio. Formato: chgrp grupo fichero.
- chmod: cambia los permisos de un fichero/directorio. Formato: chmod modo fichero.

modo es el permiso que se le asigna a cada fichero y responde a esta tabla:

OCTAL	BINARIO	PERMISOS
0	000	ninguno
1	001	- - x
2	010	- w -
3	011	- w x
4	100	r - -
5	101	r – x
6	110	r w -

Comandos para la manipulación de permisos (II)

Si queremos dar permisos a un fichero podemos hacerlo de las tres formas: octal, binario y nemotécnico, teniendo en cuenta que hay que poner los permisos para: usuario propietario, grupo y otros.

La forma más fácil de dar permisos es en octal. Ejemplo:

chmod 644 mifichero daría permisos de lectura y escritura al dueño, y de lectura a su grupo y al resto.

chmod 760 mifichero el dueño tiene todos los permisos, los que pertenezcan al mismo grupo tendrán de lectura y escritura pero **no** de ejecución, el resto de usuarios no puede realizar ninguna acción sobre el fichero.

La forma de dar permisos de forma nemónica es: chmod [usuario] [operador] [permiso].

- En lugar de usuario podemos poner: u (user, propietario), g (group), o (other, el resto), a (all, todos).
- Operardores: + para añadir, - para quitar y = para fijar el permiso.
- Los permisos: r de lectura, w de escritura, x de ejecución, t de stickybit (siguente punto a ver), s establecer uid o gid.

Ejemplos:

chmod o=r fichero permiso de lectura fijado para el resto de usuarios.

chmod u+x fichero activación del permiso de ejecución para el usuario dueño del fichero.

chmod o=r,u+x fichero mezcla de los dos ejemplos anteriores.

Más sobre los permisos

- **MASCARAS**. Los ficheros se crean con una serie de permisos por defecto que dependen del programa que los crea y del valor de la variable user mask . Su valor para todos los usuarios se define en **/etc/profile**. Con el comando ***umask*** se puede visualizar el valor de la máscara y también cambiar.

Si el sistema crea los ficheros con los permisos 777 y la máscara es de 002, el permiso resultaría ser: 777 - 002 = 775.

Si para otro usuario la máscara es 033, el resultado será: 777 - 033 = 744.

Los permisos en **modo gráfico** se cambian seleccionando el archivo y accediendo a sus **Propiedades** .

Además de los permisos anteriores, existen otros tres: sticky bit, SUID y SGID.

- El ***sticky bit***, o de permanencia, le dice al sistema que es un fichero ejecutado frecuentemente así que debería ser guardado en el área de ***swap*** para ser ejecutado más rápido, aún cuando no esté siendo ejecutado en este momento. Se muestra como un *t* (te) en el lugar donde debería estar el permiso de ejecución para los otros usuarios.

Cuando este bit está activo en un directorio (un directorio es un tipo especial de fichero), no se puede borrar nada dentro de él (salvo que los permisos lo permitan), aunque si crear nuevos ficheros y modificarlos pero sólo por su dueño. Forma de activarlo/desactivarlo: # chmod 1nnn fichero.

- SUID. Este bit le dice al sistema que cuando ejecute el fichero tome la identidad del usuario que lo ejecuta. Por ejemplo si queremos cambiar nuestra contraseña de acceso al sistema. Se identifica por una s donde debería estar el permiso de ejecución del propietario. Ejemplo, como usuario normal: ls -l /usr/bin/passwd

-SGID. Igual que SUID pero para grupos. Si un directorio tiene el bit s activado, todos los ficheros que se creen en él pertenezcan al mismo grupo que el del directorio, y el resto de usuarios del grupo podrá modificarlos.

Atributos del sistema de ficheros ext2/ext3

El sistema de ficheros de GNU/Linux trae consigo una serie de ***atributos*** que, siendo root, permiten incrementar la seguridad de un sistema. Existen dos comandos: ***chattr*** y ***lsattr.*** El primero cambia los attributos y el segundo los muestra. Los siguientes attributos son aplicables a cualquier tipo de fichero presente en el sistema, sólo veremos algunos de ellos, son particulares al sistema de ficheros ext2/3.

A NO Actualiza en el momento.
S Actualiza de forma síncrona.
a sólo permite añadir.
c comprime el fichero.
i fichero inmutable, intocable, tampoco puede ser enlazado.
d el programa dump no lo incluirá en la copia de seguridad.
s borrado seguro.
u si está establecido este atributo, cuando se borre permitirá recuperarlo.

http://es.tldp.org/Manuales-LuCAS/SEGUNIX/unixsec-2.1-html/node57.html

Ejercicio: Con la información anterior haz lo siguiente: a) Ve a la página de ayuda de chattr y lsattr y aprende sobre su funcionamiento. b) Crea un directorio y protege su contenido contra borrado, es decir, se podrá escribir en él, pero nunca borrar. c) Crea un fichero y dale un contenido extenso de texto (copia y pega) y posteriormente activa su atributo de compresión. d) Da a un archivo el permiso i e intenta hacer operaciones sobre él.

Comandos para la minipulación de usuarios y grupos

Para cada **usuario** es necesario almacenar cierta información: nombre, identificación de usuario y grupo, su clave, nombre real del usuario (completo), directorio *home* e intérprete de comandos. Toda esta información se almacena en el fichero ***passwd*** que está en el directorio ***/etc***. En el fichero ***adduser.conf*** en ***/etc*** está la información sobre como se organiza cómo se añade un nuevo usuario al sistema. Comandos:
- adduser y useradd. - passwd. - deluser y userdel (para el directorio de trabajo).
El fichero ***passwd*** tiene la siguiente forma:
usuario:clave-codificada:user-id:group-id:nombre-completo-usuario, , ,:directorio-usuario:shell
Para **deshabilitar** la cuenta de un usuario temporalmente, basta con poner un asterisco en el apartado de la clave:
usuario:*:user-id:group-id:nombre-completo-usuario, , ,:directorio-usuario:shell
- Algunas órdenes para la manipulación de grupos:
- groups. - addgroup. - delgroup.

Ejercicio: ¿Puede un usuario pertenecer a más de un grupo? Crea dos grupos (A y B) y haz pruebas con 4 usuarios. Todos pertenecen a un mismo grupo (A) y dos de ellos además han creado un grupo (B) para ellos sólos, de forma que el resto sólo puedan ver sus ficheros pero éstos podrán tendrán acceso como el resto al grupo no restringido (A).

En modo texto podemos hacerlo añadiéndolo a la línea correspondiente al grupo deseado en el fichero ***group*** situado en ***/etc,*** tras los dos punto la identificación del grupo. De forma gráfica haciendo doble click sobre el grupo deseado y añadiendo al usuario.

/etc/shadow contiene las contraseña encriptadas de los usuarios del sistema y otra serie de datos interesantes para la administración. El formato del fichero es:
usuario : contraseña_cifrada : d1 : d2 : d3 : d4 : d5 : d6 : reservado
donde cada campo corresponde con:
usuario: es el login o nombre de usuario (el mismo que en /etc/passwd)
x: contraseña: aparece una x; la contraseña se encuentra cifrada en /etc/shadow.
d1: nº de días desde el 01/01/1970 hasta último cambio de la contraseña.
d2: nº de días que deben pasar hasta que se pueda cambiar la contraseña.
d3: nº de días que deben pasar para que caduque la contraseña y deba ser cambiada.
d4: nº de días de antelación con los que avisará el sistema de la caducidad de la contraseña.
d5: nº de días con contraseña caducada antes de deshabilitar la cuenta.
d6: nº de días desde el 01/01/1970 y el día en que se deshabilitó la cuenta.
reservado: campo reservado

Para deshabilitar una cuenta de usuario basta con poner ! al principio de la línea, y para no permitir a un usuario entrar al sistema basta con poner ! antes de la contraseña.

Ejercicio: Entra en man y lee la información que aparece sobre el fichero. A uno de los usuarios que has añadido de comodín modifica su línea en el fichero shadow para que : le pida que cambien la clave cada semana , le avise todos los días cuanto tiempo queda para cambiar la clave, y que al segundo día que no haya cambiado la clave se deshabilite la cuenta. Cambia la fecha del sistema para comprobar que las modificaciones anteriores tienen efecto.

Control de grupos

Los grupos a veces pueden necesitar un administrador, y debería tener miembros y claves de acceso.
Comandos:
- addgroup : permite agregar un grupo con el nombre dado. Con la opción gid ID puede especificarse el número que identificará al grupo, número acorde con el archivo /etc/adduser.conf.
- passwd : para cambiar la clave de un grupo. Para quitar la clave de un grupo se emplea: passwd -g -r grupo . Los programas newgrp y sg no permiten cambiarse a un grupo sin clave.
- groups: un usuario puede ver los grupos a los que pertenece con este programa.
- newgrp : para cambiar el grupo de referencia durante la sesión en curso. Si no se específica grupo alguno se cambiará al grupo principal del usuario.
- gpasswd : para administrar grupos, puede ser usado por el administrador del sistema y por el administrador de un grupo. Con la opción -A usuario el administrador del sistema (root) puede agregar un administrador, un usuario normal, a un grupo. Con la opción -M usuario puede retirarse la administración de un grupo a un usuario. Con la opción -r puede quitarse la clave a un grupo con clave y con la opción -R puede inhibir el acceso con newgrp a un grupo con clave. Un administrador de grupo puede agregar y eliminar usuarios del grupo con las opciones -a usuario y -d usuario respectivamente.
- groupdel : permite eliminar un grupo. Sólo pueden eliminarse grupos que no sean el grupo principal de algún usuario.
- chgrp: cambia la propiedad de un grupo

Ejercicios: Crea dos grupos: G1 y G2 y asígnales un administrador y una contraseña a cada uno. ¿A qué grupos pertenece el usuario administrador de G1? Añade un usuario a cada uno de los dos grupos creados. Crea otro usuario, haz login como él e intenta añadirlo a un grupo con clave.

Control avanzado del acceso al sistema por parte de usuarios y grupos
Denegación de acceso a grupos completos de usuarios en tres pasos: 1º) Es necesario la instalación de libpam-modules. 2º) Modifica el archivo /etc/pam.d/login y donde pone: # Uncomment and edit /etc/security/access.conf if you need to # set access limits. # (Replaces /etc/login.access file) account required pam_access.so la última línea debe quedar descomentada. 3º) El archivo /etc/security/access.conf permite indentificar que usuarios y grupos tienen acceso garantizado o a cuales se les impide la entrada al sistema. El formato es: +|- : usuario|grupo : lugardeacceso ¡ten la precaución de no dejar espacios en blanco ni delante ni detrás de ningún campo! Ejemplo de denegación de la entrada al usuario kuku desde la consola tty1: -:kuku:tty1 Ejemplo de denegación de acceso al sistema al grupo guest desde cualquier sitio: -:guest:ALL Ejemplo de garantizar el acceso al sistema al usuario lolo desde este ordenador: +:lolo:LOCAL Ejercicios. Consulta el fichero access.conf para obtener más información sobre como denegar el acceso a todos los usuarios exceptuando a unos elegidos. Crea un grupo llamado A . Toma dos usuarios de prueba de tu sistema añádelos a A y deshabilita el acceso a ese grupo y comprueba si pueden entrar. Ahora permite entrar sólo a uno de los dos usuarios del grupo A, compruébalo.

Ejercicios (II)
1) Crea tres ficheros: numerouno, numerodos y numerotres. El primero deberá tener permiso de escritura para todo el mundo, el segundo sólo para el propietario y el tercero para el propietario y para el grupo. Hazlo igual para los permisos de lectura y ejecución. Hazlo de todas las formas posibles. 2) Cambia la máscara de creación de ficheros para que por defecto los usuarios pertenecientes a otros grupos no tengan la posibilidad de usar dichos ficheros. 3) Crea un directorio y dentro de él varios ficheros. Ahora cambia los permisos de lectura y escritura de forma que los usuarios que están en tu grupo puedan leer su contenido y sólo tú puedas escribir en él. Añade un nuevo usuario al sistema, añádelo al grupo al que tu perteneces; entra como él (su nuevousuario); intenta escribir, es decir crear un nuevo fichero, en ese directorio. ¿Puedes? 4) Crea tres usuarios: uno, dos y tres. El uno tiene acceso a los otros, el dos sólo al directorio del tres, y el tres sólo al suyo. 5) Modifica los permisos de un directorio para que aunque un usuario tenga permiso total sobre él, sólo pueda modificar sus ficheros, es decir los de su propiedad.

Ejercicio (II)
6) Deshabilita, no elimines, el usuario creado en el punto 3. 7) Disponemos de una clase con 4 alumnos, 2 grupos y un profesor. Necesitamos crear los alumnos necesarios (alumnoX), los grupos necesarios (grupoX) y el profesor, de forma que: cada alumno sólo tenga acceso a sus ficheros, cada grupo tendrá su directorio /home/grupoX en el que sólo podrán escribir los miembros del grupo pudiendo y el profesor ver los ficheros pero no modificarlos, habrá un directorio /home/clase donde los alumnos podrán ver su contenido, pero no modificarlo. 8) Deshabilita la entrada de uno de los grupos del punto anterior y de uno de los alumnos del otro grupo. Garantiza la entrada al sistema del profesor. 9) En un gran sistema el administrador no puede encargarse de atender las peticiones de todos los usuarios. Si el sistema está estructurado en departamentos. ¿cual es la mejor forma de hacerlo? Haz pruebas en tu ordenador con 13 usuarios, 2 grupos (AB). Cada grupo tendrá un administrador que podrá añadir usuarios al grupo y borrarlos. Cada grupo contiene una carpeta que no puede ser accedida por los miembros del otro grupo y otra que sí podrá serlo. Cada grupo contiene una carpeta en la que se podrá escribir, pero nunca borrar (salvo el root). Haz que los administradores de los grupos tengan como grupo de referencia el grupo que administran. Elimina el administrador de un grupo ¿qué ocurre? 10) Deniega el acceso a todo el grupo A y B excepto a sus administradores.

Parte sexta: instalación de programas desde la línea de comandos.

Toca ahora aprender sobre cómo instalar programas en un sistema derivado de Debian. Para mi mucho amantes de Linux, el sistema de paquetes de Debian es el mejor por su forma de tratar las descargas y dependencias de los programas. Veámoslo.

APT

- ¿Qué es ***apt***? Advanced Packaging Tool es una herramienta para la gestión de paquetes, permitiendo la instalación/desinstalación de grupos de programas y resolviendo las dependencias automáticamente.
- Palabras clave: .deb , i386 , repositorios, sources.list, deb y deb-src,
- Los paquetes descargados son almacenados en: /var/cache/apt/archives por si los necesitamos en otro momento.
- Uso:

apt-get (opciones: --reinstall -purge) [install | remove | update | upgrade | dist-upgrade | clean | autoclean] nom-paquete

Diferencia entre upgrade y dist-upgrade radica en que el primero sólo actualiza paquetes existentes, y el segundo actualiza todo el sistema, instalando y/o eliminando lo necesario para dejar el sistema como instalado desde la nueva versión.

www.debian.org/doc/manuals/apt-howto/index.es.html

Ejercicio: Instala el paquete Xnest y doc-debian-es y doc-linux-es.

Comprueba si está instalado el paquete xapi .Actualiza el sistema.

DPKG

El sistema nativo de Debian para instalar paquete es el DPKG y data de 1993.
dpkg (-i | -r | -purge | -l | -L | -s | -S)

a) -i :instala b) -r : elimina c) -purge : elimina completamente d) -l (ele) : lista información sobre el paq.

e) -L : muestra los directorios donde se ha instalado el paquete. f) -s : dependencias, paquetes inst y estado.

g) -S : busca un nombre de fichero correspondiente al patrón de búsqueda indicado.
http://es.tldp.org/Manuales-LuCAS/LIPP2/lipp-2.0-beta-html/node117.html
http://es.wikipedia.org/wiki/Dpkg
Existen otros programas para la instalación de paquetes: dselect y aptitude.
La instalación gráfica de paquetes se lleva a cabo en Debian y derivados con Synaptic.

Ejercicio: http://www.debianitas.net/docbook/locale/locale.html

Ejercicios

1) ¿Cuantas formas conoces de entrar al sistema como otro usuario a la vez que tienes el tuyo activo? ¿Para qué sirve el programa Xnest? Instálalo mediante ***apt*** en la línea de comandos y luego ejecuta: ***gdmflexiserver xnest***. Create una entrada en el menú Herramientas del Sistema para ejercutar la aplicación anterior.

2) Busca la aplicación **superkaramba e** instálala. Descarga un plugin para esta aplicación llamado **Stylus Sysinfo** que la puedes encontrar en **http://kde-look.org** . Descomprímela con tar -xvf sysinfo.tar.gz y obtendrás una carpeta llamada sysinfo . Ejecuta ***superkaramba&*** y añade el tema que te has descargado.

3) Usa dpkg para encontrar los directorios donde se ha instalado superkaramba.

4) Muestra las dependencias del paquete gnome-applets.

5) Elimina completamente superkaramba .

Parte septima: BASH.

El intérprete de comandos es quien posibilita al administrador tomar el control del sistema desde el punto más bajo posible. Una de las ventajas de Linux es su independencia respecto del sistema gráfico que permite que tenga un rendimiento muy superior a sus rivales.

Introducción
Shell: intérprete de comandos. ¿Y qué es BASH? Bourne Again Shell, un entorno de comandos programable. Existen variables de entorno, aunque el usuario puede crear las suyas. Ejemplos: HOME, PATH, PS1, PWD. Para usarlas se debe poner el símbolo *$* delante. Ejemplo: echo $HOME . Ejercicio: Usa variables de entorno para almacenar tus datos personales: nombre, direccion, telefono, directorio home. Visualízalas en una misma línea de forma que construyas una frase: echo mi nombre es... . Para crearlas: NOM-VARIABLE='loquesea' . ***NOTA: las variables de entorno por convención están en mayúsculas. El comando *set* permite ver el contenido de las variables. El comando *export* permite exportar el contenido de las variables a todos los procesos hijos de esta shell. Ejercicio: ejecuta el comando *set* | *less* . ¿Qué ves?

Los ficheros de configuración e inicio de BASH

Durante el arranque del sistema deben cargarse ciertos aspectos de configuración del sistema en general y los del entorno de trabajo de cada usuario en particular. Cada shell tiene sus ficheros de configuración, a nosotros nos interesa los que respecta a la Shell BASH.

- /etc/profile : algunas variables de entorno y otros parámetros para el resto de usuarios del sistema : prompt, path, tamaño máximo de ficheros que se pueden crear, permisos por defecto, tamaño de los ficheros del historial, ...

- ~/.bash_profile : configura opciones de cada usuario; se encuentra en los directorios *home*. Este fichero llama a *.bashrc* que se encuentra también en *home,* pero está comentada.

- .bashrc : información/configuración específica de cada usuario. Sobreescribe información que se definió para el conjunto de usuarios del sistema. Se lee cada vez que el usuario entra en el sistema y cada vez que se crea un terminal shell bash.

- Ejercicio: localiza las líneas en .bashrc donde se establece el prompt y busca una dirección de internet donde te explique cómo personalizarlo con más opciones.

Castellanizando Linux

A veces nos podemos encontrar con que el sistema no viene preparado para ver bien todos los caracteres característicos del Español. Un problema típico es que la tilde no aparece encima de una vocal y pero sí aparecen caracteres extraños que impiden una correcta lectura.

Como ejercicio vamos a configurar nuestro sistema para la correcta visualización de los caracteres en Español y el símbolo del Euro ¬

En Debian existe un fichero que se encarga de decirle al sistema qué tabla de códigos cargar: /etc/enviroment. Sólo hay que poner estas líneas, eliminando claro está las ya existentes:

LC_ALL=es_ES@euro
LANGUAGE=es_ES@euro
LANG=es_ES@euro

Aunque hay que hacer alguna cosa más. Veámoslo con un ejercicio.

Ejercicio: http://www.debianitas.net/docbook/locale/locale.html

Prueba ahora a abrir una página de ayuda con man y comprueba que esté en español.

Redireccionamientos
En Unix y derivados, la línea de comandos permite el redireccionamiento. Veamos sus diferentes formas: - Redireccionamiento de entrada: (<). Hace que el comando de la izquierda de < tome como entrada el comando de la derecha. Por ejemplo: # cat < /etc/passwd - Redireccionamiento de salida: (>). Hace que el comando a la izquierda del > vuelque su salida en vez de por pantalla, al comando o fichero que se encuentra a lado derecho del > . Por ejemplo: cat > fichero. #ls ~/ > fichhome - Redireccionamiento para añadir (>>). Igual que el redireccionamiento de salida, pero en vez de sobreescribir, añade al final. Útil para escribir al final de ficheros: # df -h >> historialdedisco. -Tuberías: (|). Este símbolo se forma con ALTgr+1. Las tuberias sirven para enlazar distintos comandos de forma que la salida de uno será la entrada del siguiente (izq a dch). Ejemplo: dmesg | less y prueba también dmesg | grep PCI Ejercicios. 1) Crea un fichero con el comando cat > fichero. (pulsa control+z para finalizar de escribir). Compruebalo. 2) Vuelca el contenido que produce dmesg en un fichero. 3) Ejecuta: tail -4 < /etc/passwd ¿Qué obtienes? y prueba también: cat /etc/passwd > tail -4 ¿se puede? 4) Vas a usar en tubería las órdenes cat, tail -10 y wc sobre /etc/passwd. ¿Qué obtienes? 5) La orden sort ordena un texto. Accede a /etc/passwd, órdenalo y escríbelo en tu escritorio como contraseñas .

Listas de comandos
En Unix y derivados existe la posibilidad de ejecutar comandos de forma secuencia, paralela o condicionada. - comando& hace que el comando se ejecute en segundo plano. - comando1 ; comando2 ; comando3 hace que los comandos se ejecuten en orden de izquierda a derecha pero esperando la finalización del precedente. - comando1 && comando2 hace que se ejecute comando2 si y solo si comando 1 lo hace correctamente. - comando 1 ‖ comando2 hace que ejecute comando 2 si no ha terminado correctamente comando1. Ejercicios: 1) Desde el directorio principal ejecuta el comando ls sobre tu directorio home , luego si todo ha ido bien, muévete a /home/tuusuario. 2) Como usuario normal ejecuta e intenta borrar un fichero de /etc y si no es posible que se ejecute posteriormente un comando informando NO HA SIDO POSIBLE PORQUE NO ERES ROOOOOT . 3) Muestra, como usuario normal, el contenido de /etc/group y si se ha ejecutado de forma correcta otro que muestre el mensaje OOOOLEEEE . 4) Ejecuta en una misma línea: acceso a tu directorio home, muestra el .bashrc, copia .bashrc tu escritorio como *micopia*.

Alias
En Unix y derivados existe la posibilidad de ejecutar comandos de forma secuencia, paralela o condicionada. - comando& hace que el comando se ejecute en segundo plano. - comando1 ; comando2 ; comando3 hace que los comandos se ejecuten en orden de izquierda a derecha pero esperando la finalización del precedente. - comando1 && comando2 hace que se ejecute comando2 si y solo si comando 1 lo hace correctamente. - comando 1 ‖ comando2 hace que ejecute comando 2 si no ha terminado correctamente comando1. Ejercicios: 1) Desde el directorio principal ejecuta el comando ls sobre tu directorio home , luego si todo ha ido bien, muévete a /home/tuusuario. 2) Como usuario normal ejecuta e intenta borrar un fichero de /etc y si no es posible que se ejecute posteriormente un comando informando NO HA SIDO POSIBLE PORQUE NO ERES ROOOOOT . 3) Muestra, como usuario normal, el contenido de /etc/group y si se ha ejecutado de forma correcta otro que muestre el mensaje OOOOLEEEE . 4) Ejecuta en una misma línea: acceso a tu directorio home, muestra el .bashrc, copia .bashrc tu escritorio como micopia.

Historial de comandos
Los comandos que ejecutas se almacenan en una memoria. El número de órdenes se almacena en una variable de sistema denominada HISTSIZE. La variable HISTFILE contiene el nombre del fichero que por defecto almacena el historial, que como mucho podrá tener HISTFILESIZE de tamaño. El fichero que almacena las órdenes es ~/.bash_history. Ejercicio: muestra el historial de órdenes con: # history | less. Muestra el valor de las variables relacionadas. Ejecuta !e , !1 , !5 y !10 y dí que hace. Crea un fichero cuya parte primera sea el contenido del historial de tus comandos, como parte segunda un texto en el que pongas que lo has creado como prueba. Visualízalo.

Búsqueda de ficheros
En Linux tenemos distintas formas de localizar ficheros: a) locate nombre-comando. Si la base de datos estuviese antigua habría que ejecutar: locatedb. b) find busca ficheros según un patrón de búsqueda. Es bastante potente y con muchas opciones. Ejemplos: 1) find . -name *.txt busca todos los ficheros con extensión txt desde el directorio actual. 2) find . -size +50k busca todos los ficheros que ocupen más de 50 kilobytes desde el directorio actual. 3) find /home/usuario -empty busca todos los ficheros vacios en /home/usuario. 4) find . -atime 1 busca todos los ficheros que han sido modificados en 1x24 horas. c) whereis comando. Ejercicios. 1) Accede a la página de ayuda de find. Busca a) todos los ficheros menores de 1 MB; b) todos los ficheros creados antes de las última semana (7x24). c) todos los ficheros que pertenezcan al grupo al cual tú perteneces; d) todos los ficheros que tengan como segundo carácter una a . 2) Utiliza locate para buscar: find, amule, Xnest. 3) Accede a la página de ayuda de whereis y mira las posibilidades que te ofrece. Localiza la situación de los comandos del ejercicio anterior con el comando whereis. Busca también si los comandos anteriores tienen página de manual. Ejecuta el ejemplo que te pone en la página de man.

Procesamiento de ficheros

Unix y sus derivados tienen multitud de órdenes para tratar los ficheros: trocearlos, comprimir, mover, ver...

- split es un comando que divide un fichero en trozos. Útil por ejemplo para luego enviarlos por email.

split [opciones] fichero-origen fichero-destino

split -b 1m fichero.zip fichero.zip. <-termínalo en punto

Daría como resultado fichero.zip.XX donde XX son letras consecutivas.

Para **unir** los trozos nada más fácil que: cat fichero.zip.* > fichero.zip y como si nada.

- tar: es un comando usado para empaquetar ficheros y directorios con un nombre común.

tar [opciones] nom-fichero-final.tar [fichero1 fichero2 fichero3]

tar -cvf fichero.tar ficherosacomprimir

tar -xvf ficherocomprimido.tar

- gzip: permite comprimir ficheros. Una uso típico de Unix es usar tar para empaquetar y luego usar gzip para aumentar la capacidad de compresión. El comando gunzip descomprime.

gzip [-d -1..9 -t] archivo-a-comprimir (-d para descomprimir, ¿opciones al final?)

También cabe la posibilidad de comprimir y descomprimir con tar con la opcion -z antes.

- bzip2 (comprime) y bunzip2 (descomprime).

Ejercicios.

1) Comprime todo tu directorio home: primero con tar, y luego el fichero tar lo comprimes con gzip y con bzip2. ¿Qué comprime más gzip o bzip2? Extrae los dos ficheros comprimidos con tar, gzip y bunzip2.

2) Divide el fichero .gz del apartado anterior usando el comando split en ficheros de 500k; posteriormente los unes con cat.

Control de procesos (I)
Un proceso es un programa en ejecución. Unix trae utilidades para actuar sobre los procesos. - ps [opciones]. Lista los programas en ejecución. Ejecuta: ps -U tunombreusuario. Para ver una lista de opciones: ps -help. - Para ejecutar un comando en segundo plano: comando&. - Cuando se usa **Control+Z** sobre un proceso, esto pone en modo **pausa** un programa pudiendo continuarlo despues. Y si queremos que continue su ejecución bastará con poner **bg** (background) para que se ejecute en **segundo plano**, o bien, **fg** (foreground) si queremos que lo haga en **primer plano**. No confundir Control+Z con Control+C que interrumpe su ejecución sin posibilidad de continuar su ejecución por donde se paró. Si hubiese más de una tarea suspendida, podría seleccionarse cual queremos activar con: bg %numerotarea. - Para interrumpir/matar procesos: kill [-9 | -15] PID.

Control de procesos (I)

- Ejecución diferida. El comando **at** permite temporizar la ejecución de comandos. Por ejemplo: ejecutar la compresión dentro de 20 minutos: at +20 minutes (pulsamos ENTER para introducir el comando)

> tar -cvf comprimido /media/datos (este es el comando, para terminar CONTROL+D)

- El comando atq muestra un listado de los comandos en espera de ser ejecutados.
- El comando atrm elimina trabajos en espera.

Ejercicios:

1) Ejecuta el comando ps con las opciones l, u, a, x, r y t. Primero por separado y luego juntas. Mira en la página de ayuda que hace cada una.

2) Ejecuta el comando top . Páralo temporalmente. Continúalo después en primer plano, páralo de nuevo y continúalo pero en segundo plano. ¿Sirve de algo tener este comando en segundo plano?

3) Vuelve a ejercutar top , busca su PID y mátalo usando -15 y -9.

4) Ejecuta tres comandos en modo diferido. Los comandos son: echo le vamos a regalar una XBox a Manolo ; shutdown apagándose en 30 minutos; por último una compresión de todo tu directorio. Comprueba que están en espera y elimina dos.

Parte septima:
SCRIPTS.

La versatilidad de la línea de comandos de Unix/Linux hace posible la encadenación de comandos en ficheros de textos que permiten realizar tareas de forma automatizada y repetidamente. En este tema veremos como hacerlo usando el/la shell BASH.

Introducción
Una vez hemos visto los conceptos básicos de la línea de comandos vamos a entrar de lleno en la programación de scripts en Linux con el shell BASH: - Creación del ficheros de scripts. - Escritura en la salida estándar / Lectura de la entrada estándar. - Operaciones aritméticas. - if-then-else-if, operadores lógicos, y evaluación test, de enteros, archivos. - case. - Bucles: while, until y for. - funciones. ***NOTA: guarda todos los scripts que vamos a realizar en ficheros diferentes para luego practicar en tu casa.

Ficheros de scripts
Todos los scripts son archivos de texto y deben comenzar de la siguiente forma: #!/bin/bash En esa línea le indica al shell que el intérprete que va usar para este script está en /bin/bash, es decir, el shell bash. Además el archivo de texto debe tener los permisos de ejecución: chmod +x nombrescript.sh Ejercicios: 1) crea un script que muestre un fichero por pantalla. 2) crea un script que muestre por pantalla el contenido de una variable. 3) crea un script que muestre la fecha actual (busca el comando).

Los parámetros

Unas cosas más sobre las variables:

$$ es el numero identificador del proceso del shell.

$? Resultado de la ejecución del comando anterior.

$0 Nombre del script que se esta ejecutando

$1-$9 Primer a noveno argumento con los que se invoca.

$* Todos los argumentos como única palabra separador _.

$@ Array con todos los argumentos pero con posición.

$# Numero de argumentos recibidos.

La variable de entorno $? puede tener dos posibles valores: 0 o 1. 1 significa fracaso en la ejecución del comando/script anterior, y 0 éxito.

Ejercicios
Crea un script que te proporcione el número de identificación de proceso que tiene. También deberá mostrar el contenido del directorio donde se está ejecutando y el contenido del propio script. Crea un script al que le vas a pasar tu nombre, apellidos, dirección y teléfono y te los va a mostrar uno en cada línea. Al final deberá poner el número de parámetros que has introducido. Añade al script del ejercicio 1) la comprobación de si cada comando se ha ejecutado correctamente. Añade al script del ejercicio 2) un comando que imprima en una sola línea todos los parámetros que le has pasado.

Parámetros del comando echo
\a : alerta. \b : retroceso. \c : no pasa a la siguiente línea. \f : nueva hoja (impresoras). \n : nueva línea. \r : retorno de carro. \t : tabulación horizontal. \v : tabulación vertical. \\ : escribe \ con la opción -e. \XXX : imprime el carácter ASCII en octal. -e : hace que se interpreten los símbolos anteriores. echo -e beeep\a echo -e "alumno\ba" echo -e hola\c ; echo -e chica echo -e "hoja 1\fhoja2" | lpr echo -e linea 1\nlinea2 echo -e "andromeda\rA" echo -e sin tabular\ttabulada echo -e sin tabular\vtabulada echo -e la barra es \\ echo -e "\033[1;31mRojo\033[0;0m"

Ejercicios
Realiza un script que muestre un fichero que le pasan por teclado. Antes de mostrarlo deberá imprimir un mensaje advirtiendo de ello junto con el nombre del fichero y al finalizar la impresión por pantalla deberá sacar otro mensaje advirtiéndolo junto con el nombre del fichero. Antes de empezar a imprimir y justo después debe imprimirse una bocina. Realiza un script al que le pasen como parámetros cinco palabras que deberán ser mostradas tabuladas primero horizontalmente y luego en una línea distinta verticalmente. Realiza un script al que le pasen como parámetros tres palabras que serán mostradas en tres colores diferentes. Realiza un script que reciba un nombre de fichero. Éste será imprimido por pantalla siguiendo las siguientes órdenes: a) de la línea 1 a la 10 se imprimirán en rojo, de la 20 a 25 en azul y del a 26 a la 30 en amarillo, por último un mensaje de despedida en el color habitual. Busca la forma de incluir una pausa entre una impresión y la siguiente o el final. Pista. órdenes tail y head.

Capturando datos de comandos
La salida de un comando puede ser almacenada en una variable: a) de la salida de un comando: quiensoy=`whoami` (con el acento francés) b) de la salida de la unión de varios comandos espacioescritorio=$(ls -l /home/$USER | grep Desktop) c) de una operación aritmética: suma=$[4+5] d) de una cadena: variable= uno;dos;tres;cuatro echo $variable | cut -d";" -f1 esto muestra uno

Ejercicios
Crea un script que va a recibir los siguientes parámetros: 1º el directorio de usuario, 2º un fichero del directorio, 3º y 4º dos números. La salida del script debe propocionar dos líneas diferentes: 1ª) la salida con la opción -la /directorio/fichero y que sólo muestre la línea donde está ese fichero; 2º) la multiplicación de los números. Crea un script que reciba dos comandos por parámetros y los ejecute mostrando su salida por pantalla. Ejemplos de comandos podrían ser ls sobre el directorio actual, un cat que se realizaría sobre el mismo script, averiguar quien es el usuario que está ejecutando el comando, etc... Crea un script que dado un número de línea como parámetro obtenga del fichero passwd el usuario al que pertenece. Pista: head, tail, y cut. Crea un script que dado un fichero como parámetro te muestre el número de líneas total que tiene y en la línea siguiente te muestre justo la de en medio.

Entrada de datos interactiva desde la línea de comandos en un script
Existe un comando que puede ser muy útil: read. Éste te permite introducir datos al script desde el teclado. Ejemplo: read variable1 variable2 echo la variable 1 tiene como valor $variable1 Además read admite opciones: a) -p texto : indica que ponga un texto como prompt para pedir el dato. b) -r : le quita el significado a las barras invertidas \ . c) -s : modo silencio, no muestra lo que escribe. d) -t : pone un tiempo límite en segundos antes de dar por errónea la entrada de datos. Ejemplo: read -s -t 10 -p por favor introduzca su nombre nombre echo $nombre

Operadores
Operadores aritméticos: + - * ** / % ++ -- Operadores de comparación: == != < <= > >= -eq nt lt -le gt ge Operadores lógicos: ! && ‖ Operadores binarios: & | ^ << >> Operadores de asignación: = *= /= %= += -= <<= >>= &= ^= |= Operadores de tipos de ficheros: -e b c d f -h -L -p -S t Operadores de permisos: -r -w -x -g -u -k -O -G N Operadores de fechas: -nt ot et Operadores de cadenas: -z -n Los veremos según toque en cada momento.

Operadores aritméticos

Para llevar a cabo operaciones aritméticas: comando let. (buscad en Internet cómo se define una variable numèrica).

```
let a=5  # Asignación a=5.
let b=$a+3*9  # b=a+(3*9)=32.
echo "a=$a, b=$b"      #  a=5, b=32
let c=$b/($a+3)          # c=b/(a+3)=4.
let a+=c--        # a=a+c=9, c=c-1=3.
echo "a=$a, c=$c"      # a=9, c=3
let CTE=$b/$c; RESTO=$b%$c        # CTE=b/c; RESTO=resto(b/c).
echo "Cociente=$CTE, Resto=$RESTO"      # Cociente=10, Resto=2
```

$((Expresión)) donde Expresión es un calculo aritmético

$[Expresión]

Especificación del entrecomillado
Los 3 tipos básicos de entrecomillado definidos en BASH son [2]: · Carácter de escape (\Carácter): mantiene el valor literal del carácter que lo precede; la secuencia \\ equivale a presentar el carácter \. Cuando aparece como último carácter de la línea, sirve para continuar la ejecución de una orden en la línea siguiente. · Comillas simples ('Cadena'): siempre conserva el valor literal de cada uno de los caracteres de la cadena. · Comillas dobles ("Cadena"): conserva el valor de literal de la cadena, excepto para los caracteres dólar ($), comilla simple (') y de escape (\$, \\, \', \", ante el fin de línea y secuencia de escape del tipo ANSI-C). Ejemplos: echo "Sólo con permiso \"root\" " # da como salida: Sólo con permiso "root" echo 'Sólo con permiso \"root\" ' # da como salida: Sólo con permiso \"root\" echo Soy $USER # da como salida: Soy *usuario* echo 'Soy $USER # da como saldia: Soy $USER

Ejercicios
Crea un script que lea con read dos números y haga la multiplicación de ellos. Crea un script que lea de teclado un usuario que sepas que existe en tu sistema. Busca dicho usuario en el fichero /etc/passwd. Usa el comando grep . Crea un script que lea con read dos números y realice una operación aritmética. El read esperará 3 segundos a que lo introduzcas en caso contrario el script debe terminar. Crea un script que simula la entrada de una contraseña pero de forma que lo introducido no se vea. Crea un script que simula la creación de un usuario introducido por teclado con el comando read.

Ejercicios

1) Introduce 3 números por teclado y muestra la media.
2) Calcula el área de un rectángulo cuyos datos se introducirán por teclado.
3) Dado el año actual con 'date +%Y' (ojo acento francés) calcula la edad que tiene una persona cuya edad se introduce por teclado.
4) Dado un número del 1 al 9, escribe su tabla de multiplicar.
5) Introduce dos números por teclado almacenándolos en variables, y sólo usando la multiplicación y la división intercambia los valores.
6) Resuelve la siguiente ecuación donde x es un número introducido por teclado:

f(x)=2x+(10/2)+100*x

Condicional: if

En los scripts también se puede controlar el flujo del programa con sentencias de control. Hay que respetar escrupulosamente el formato de if-then-else-if para que sea correcto. Formato:

```
if  expresion-a-evaluar;  # este comando siempre se ejecuta
   then
bloque de comandos
a ejecutar
[ elif expresión-a-evaluar
bloque de comandos
a ejecutar]
[else
   comandos]
fi          #fin del if
```

Ejemplo de if:

```
#!/bin/bash
echo   comprobación de si un usuario existe en el sistema
read -p "por favor introduzca su nombre: " nombre
if grep -L  -l  ^$nombre /etc/passwd
  then
       echo existes
echo
  else
       echo no
fi
```

*NOTA: los parámetros -L y -l son para que no muestre la ejecución de grep

Test

El shell usa un comando llamado test para evaluar expresiones condicionales. Pueden ir dentro de las expresiones de if entre corchetes: if [-e /etc/passw]. O bien if test -e /etc/passwd
test devuelve 0 (verdadero) o 1 (falso), opciones :
a) -b fichero : Verdadero si fichero existe y es un fichero de bloques.
b) -c fichero : Verdadero si fichero existe y es un fichero de caracteres.
c) -d fichero : Verdadero si fichero existe y es un directorio.
d) -e fichero : Verdadero si fichero existe.
e) -f fichero : Verdadero si fichero existe y es un fichero normal (regular).
f) -g fichero : Verdadero si fichero existe y tiene activo el bit de grupos.
g) -k fichero : Verdadero si fichero tiene su sticky bit activo.

Test (II)

a) O fichero : Verdadero si fichero existe y es propiedad del usuario activo.
b) G fichero : Verdadero si fichero existe y es propiedad del grupo al que pertenece el usuario.
c) fichero1 -nt fichero2 - Verdadero si fichero1 es mas nuevo, según la fecha de modificación, que fichero2.
d) fichero1 -ot fichero2 - Verdadero si fichero1 is mas viejo que fichero2, según fecha de modificación.
e) fichero1 -ef fichero2 - Verdadero si fichero1 y fichero2 tienen el mismo numero de dispositivo y de inode.

a) -z cadena : Verdadero si la longitud de cadena es 0.
b) -n cadena : Verdadero si la longitud de cadena no es 0.
c) -l cadena : devuelve la longitud de la cadena. Otra forma de evaluarla es: ${#cadena}
d) cadena1 = cadena2 : Verdadero si las cadenas son iguales

e) cadena1 != cadena2 : Verdadero si las cadenas no son iguales.
f) ! expr : Verdadero si expr es falso.
g) expr1 -a expr2 : Verdadero si expr1 y expr2 son verdaderos.
h) expr1 -o expr2 : Verdadero si expr1 o expr2 es verdadero.
i) arg1 OP arg2 : OP es uno de -eq, -ne, -lt, -le, -gt, or -ge.

Ejercicios

1) Crea un script en el cual se introduzcan los nombres y edades de dos personas. Comprueba si esos datos son iguales e imprime el resultado.
2) Crea un script al que le pases por parámetros dos nombres de ficheros. Comprueba que existan y del tipo que son mostrando el mensaje oportuno.
3) Crea un script al que le introduzcan tres números. Comprueba si los números han sido introducidos como parámetros porque en tal caso no hará falta leerlos. Ordena los números de menor a mayor.
4) Crea un script que reciba tres nombres de ficheros y que haga una compresión sólo de los ficheros que pertenezcan al usuario que ejecuta el script.
5) Crea un script que reciba un nombre de fichero y lo busque e indique si existe o no.
6) Haz un script que centre una cadena de texto en la pantalla. La fórmula es:
espacios=$(((ancho-${#cadena})/2))

CASE

En los scripts también existe un *switch* como en C o *case* como en otros lenguajes.

```
case Variable in
[(]Patrón1) Bloque1 ;;
...
esac
```

Ejemplo:

```
case $1 in
    (yes|true) echo   has introducido yes/true  ;
   echo   imagino que quieres continuar...
  ;;

echo   imagino que querrás salir...  ;
exit;
  ;;
esac
```

Bucles

Los bucles deben tener siempre una condición de salida, de lo contrario sería infinitos y crearían inestabilidad.

Para romper la secuencia de un bucle se pueden utilizar los siguientes comandos:

a) break: rompe el bucle saltando al comando inmediatamente posterior. Debe utilizarse con cuidado ya que la depuración del código se vuelve más complicada.

b) continue: rompe el bucle volviendo a la condición de éste y dejando el resto del código del bucle sin

ejecutar. También debe usarse con cuidado.

For

Empecemos con las distintas formas de realizar un bucle for . El funcionamiento es el mismo que en C y otros lenguajes de programación, únicamente habrá que adaptarse a este nuevo lenguaje. Veamos al primera forma que sirve para tratar el resultado de una lista de uno en uno:

```
 for variable [ in Listadeloquesea] ; do
Bloque de comandos
 done
```

- Ejemplo:

```
for variable in `ls /home/$USER`; do
echo $variable  # esto va a mostrar lo que hay en el directorio
echo `ls /home/$USER/$variable` # muestra el contenido de $variable
done
```

Ejercicio: adapta el bucle for anterior para mostrar los nombres de usuarios que hay en el fichero /etc/passwd. Para complicarlo más, muestra junto al nombre de usuario su contraseña encriptada que está en /etc/shadow.

La siguiente forma de hacer el bucle for es como la del lenguaje C. Veámoslo:

```
For ((Exp1; Exp2; Exp3)); do
Bloque de comandos
done
```

Ejemplo:

```
for (( i=1; i<20; i++ )); do
echo    2 x $1 = $[2*$i]
done
```

Ejercicios: Busca los ejercicios que tienes de la asignatura de programación y adáptalos para usar este bucle for.

While

El uso de estos bucles son muy parecidos a los de C. Veamos primero el formato del While:
while expresión; do
bloque de comandos
done
Ejemplo: Mostrar las líneas del fichero /etc/passwd pero una a una
typeset -i nlineas=`wc -l /etc/passwd`
typeset -i i=1;
while [$i -le $nlineas]; do
echo mostrando la línea $i
echo `cat /etc/passwd | head -${i} | tail -1`
let i++
done

Until
El formato de este bucle es: Until [expresión cierta]; do bloque de comandos done Para poner un ejemplo de este bucle vamos a basarnos en el anterior del while pero obteniendo el nombre de usuario: typeset -i nlineas=`wc -l /etc/passwd | cut -d -f1`

```
typeset -i i=1;
until [ $i -gt $nlineas ]; do
usuario= `cat /etc/passwd | head -${i} | tail -1 | cut -d : -f1 `
echo mostrando el usuario $i: $usuario
let i++
done
# ¡¡¡Cuidado con las comillas al copiar y pegar al terminal!!!!!
```

While y Until

El uso de estos bucles son muy parecidos a los de C. Veamos primero el formato del While:

```
while expresión; do
bloque de comandos
done
```

Ejemplo: Mostrar las líneas del fichero /etc/passwd pero una a una

```
typeset -i nlineas=`wc -l /etc/passwd`
typeset -i i=1;
```

```
while [ $i -le $nlineas ]; do
echo   mostrando la línea $i
echo `cat /etc/passwd | head -${i} | tail -1`
let i++
done
```