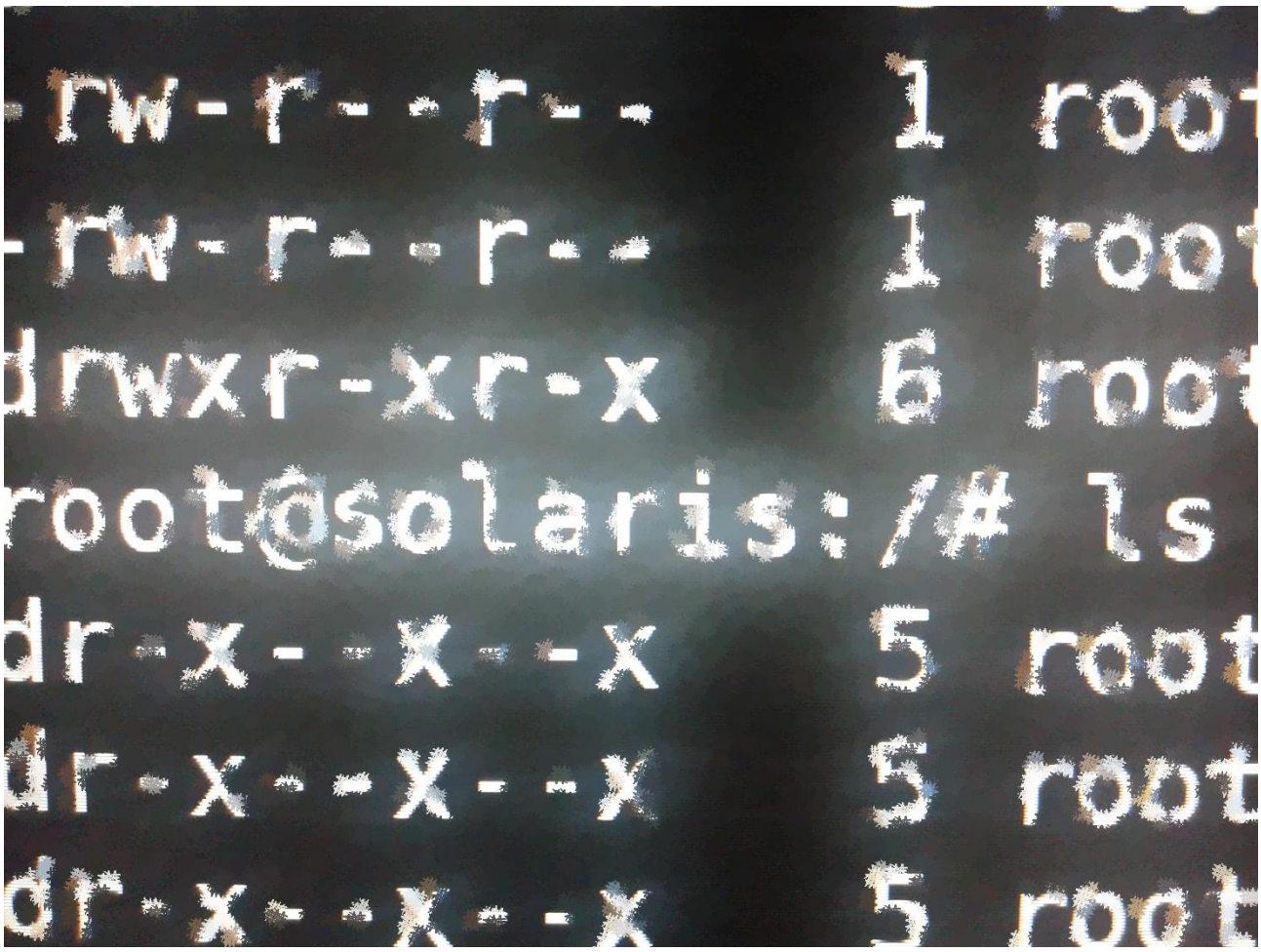

Oracle Solaris 10: 1Z0-876

Gustavo Tejerina Santos

ORACLE Certified Associate
Oracle Solaris 10 Operating System

V. 1.0 agosto 2016

Sobre UNIX Solaris...

Solaris es uno de los sistemas UNIX más populares y empleados a nivel empresarial en la actualidad.
Fue originalmente desarrollado por **Sun Microsystems** en 1992 y mantenido y evolucionado por la compañía hasta la compra de la misma por parte de **Oracle Corporation** en 2010. En ese momento, Solaris se encontraba en la versión 10, liberada inicialmente en enero de 2005.
Solaris es una evolución del sistema **SunOS** que estaba basado en el sistema **BSD**.
Solaris está disponible tanto para arquitecturas SPARC como x86.
En cuanto a las novedades de la versión 10 respecto a sus predecesoras, destaca la inclusión del sistema de ficheros **ZFS**, **GRUB** como gestor de arranque y el entorno de escritorio **Java Desktop System**, por defecto.

Sobre el examen de certificación 1Z0-876...

El examen **1Z0-876** es la vía para la obtención del título **Oracle Certified Associate, Oracle Solaris 10 Operating System**.
La prueba consiste en un test de 52 cuestiones, a resolver en un tiempo máximo de 90 minutos, y en las que existirán una o varias posibles respuestas correctas.
Las preguntas planteadas, a pesar del formato de test, se pueden considerar fundamentalmente prácticas.
En muchas de ellas se mostrará una situación (*exhibit*) que puede constar de una serie de comandos, del resultado de la ejecución de los mismos, del contenido de un fichero de configuración, etc. sobre la que se planteará una pregunta.

Sobre el presente manual...

Esta obra desarrolla el temario oficial del examen **1Z0-876** que establece **Oracle University**.
Se incluyen los conceptos fundamentales sobre el sistema Solaris que debe asimilar el candidato a la prueba.
La teoría contenida intenta ser suficiente y concisa y se centra fundamentalmente en el uso de comandos, incluyendo las opciones más relevantes y ejemplos ilustrativos de los mismos.
Por otra parte, la información aquí contenida requiere de unos conocimientos básicos sobre sistemas UNIX y, de cara a la obtención del certificado, se recomienda al candidato profundizar en el aprendizaje práctico de los conceptos desarrollados.
Igualmente, para la profundización en los conocimientos, el manual en línea del propio sistema aportará datos y opciones de comandos que no se han incluido por considerarse menos relevantes.

I. Ver y utilizar los componentes de un sistema Solaris.

I.I. Inicio de sesión en línea de comandos y cambio de contraseña de usuario.

Una sesión es la colección de aplicaciones, configuraciones y recursos asignados al usuario dentro del sistema. El fichero **/etc/default/login** permite establecer las políticas de login en el sistema. Los parámetros que contiene son:

TIMEZONE	Establece el valor de zona horaria para la fecha y hora del sistema.
ULIMIT	Tamaño máximo del fichero de login.
CONSOLE	Si la línea está comentada, el usuario **root** puede acceder al sistema a través de los protocolos **telnet** o **rlogin**.
PASSREQ	Determina si el login requiere contraseña.
ALTSHELL	Determina si la variable de entorno SHELL es establecida.
PATH	Establece el valor inicial de la variable de entorno PATH.
SUPATH	Establece el valor inicial de la variable de entorno PATH para el usuario **root**.
TIMEOUT	Nº de segundos de espera antes de abandonar la sesión de login.
UMASK	Permite establecer el valor inicial de la máscara de permisos.
SYSLOG	Establece si se registran en el log del sistema los intentos de login de **root**.
SLEEPTIME	Nº de segundos de espera para mostrar que el login es incorrecto.
DISABLETIME	Nº de segundos de espera para reintentos al introducir la contraseña.
RETRIES	Nº de reintentos posibles para introducir la contraseña del usuario.
SYSLOG_FAILED_LOGINS	Nº de intentos fallidos para mostrar el mensaje de login erróneo.

Comando **login**

- ✓ Establece una nueva sesión en el sistema.
- ✓ Sólo puede ser ejecutado como comando por el usuario **root**.
- ✓ Es posible deshabilitar el login de los usuarios en el sistema con la creación del fichero **/etc/nologin**. El contenido del mismo será el texto mostrado a los usuarios al intentar iniciar sesión.

-f	Sin autenticación. La autenticación es previa.
-h	Permite indicar el nombre del host.
-p	Mantiene las variables de entorno actuales.

Comando **logins**

- ✓ Muestra información sobre los inicios de sesión de los usuarios en el sistema.
- ✓ Sin modificadores, las columnas en que se muestra la información corresponden a:
  - Nombre de usuario.
  - ID de usuario.
  - Nombre de grupo al que pertenece el usuario.
  - ID de grupo.
  - Descripción del usuario contenida en el fichero **/etc/passwd**.

```
$ logins -x -l gustavo
gustavo         100     gustavo         100
                        /export/home/gustavo
                /bin/bash
                PS 020816 -1 -1 -1
```

✓ Los datos de la última línea, relativos a la contraseña, son, de izquierda a derecha:
- Estado.
- Última fecha en que la contraseña fue modificada.
- Nº de días requeridos entre cambios de contraseña.
- Nº de días requeridos para poder realizar un cambio.
- Nº de días de preaviso por caducidad de contraseña.

-l	Permite indicar un nombre de usuario concreto sobre el que mostrar los datos.
-p	Muestra los datos de los usuarios sin contraseña para login en el sistema.
-x	Amplía los datos de salida añadiendo el directorio **home** del usuario, **shell** por defecto y estado de la contraseña.

Los posibles estados de la contraseña se detallarán en el apartado del comando **passwd**.

Comando **su**

✓ Permite pasar a ser otro usuario del sistema sin abandonar la sesión actual.
✓ Si no se especifica nombre de usuario como parámetro, se cambia a usuario **root**.
✓ El argumento "**-**" hace que los datos del entorno pasen a ser los del usuario destino. En caso contrario, se mantienen los datos de entorno del usuario origen.

```
$ su - root
```

➔ Cambio a usuario **root** adquiriendo su configuración de entorno.

Comando **passwd**

✓ Cambia la contraseña de usuario o muestra información relativa al estado de la misma.

-l	Bloquea cuenta de usuario.
-u	Desbloquea cuenta de usuario bloqueada (LK).
-s	Muestra los atributos de contraseña del usuario:

LK	Cuenta bloqueada por el sistema.
NL	Es una cuenta que no permite login.
NP	Cuenta sin contraseña.
PS	Cuenta que probablemente tiene una contraseña válida.

	UN	Los datos en el campo contraseña son desconocidos.
	UP	La cuenta no se encuentra activa.
-as	Muestra los atributos de contraseña de todos los usuarios.	

```
# passwd gustavo
Nueva contraseña:
Vuelva a escribir la nueva contraseña:
passwd: la contraseña se ha cambiado para gustavo satisfactoriamente
```

➔ En el ejemplo se modifica la contraseña del usuario "gustavo" desde la cuenta de **root**.

Las directivas que deben cumplir las contraseñas de los usuarios se definen en el fichero **/etc/default/passwd**:

MAXWEEKS	Número máximo de semanas de vigencia.
MINWEEKS	Número de semanas que deben pasar antes de que la contraseña pueda ser modificada.
PASSLENGTH	Número mínimo de caracteres de la contraseña.
NAMECHECKNO	Limitación de que la contraseña pueda ser igual al nombre de usuario.
HISTORY	Establece el número de contraseñas previas a mantener y comprobar en un cambio.
MINDIFF	Número mínimo de caracteres distintos respecto a contraseña anterior.
MINALPHA	Número mínimo de caracteres alfabéticos que debe contener.
MINNONALPHA	Número mínimo de caracteres no alfabéticos que debe contener.
MINUPPER	Número mínimo de caracteres alfabéticos en mayúscula que debe contener.
MINLOWER	Número mínimo de caracteres alfabéticos en minúscula que debe contener.
MAXREPEATS	Número máximo de repeticiones permitido para un carácter.
MINSPECIAL	Mínimo de caracteres especiales que debe contener.
MINDIGIT	Mínimo de caracteres numéricos que debe contener.
WHITESPACE	Limitación a usar o no espacios en blanco.
DICTIONLIST	Nombre de ficheros, separados por coma, con cuyos nombres se crea una base de datos para determinar si una contraseña está basada en una palabra de diccionario.
DICTIONDBDIR	Directorio en el que se encuentra la base de datos de comprobación de diccionario.

Las contraseñas de los usuarios se almacenan, encriptadas, en el fichero **/etc/shadow**. Este fichero también contiene datos sobre el tiempo de vigencia de las contraseñas.
Los datos contenidos en el fichero **/etc/passwd** comprenden los siguientes campos separados por ":" y de izquierda a derecha:

```
$ grep gustavo /etc/passwd
gustavo:x:102:100:Gustavo Tejerina:/export/home/gustavo:/bin/bash
```

- Nombre del usuario.
- Si se muestra el carácter "**x**", la contraseña encriptada está almacenada en **/etc/shadow**.
- ID de usuario.
- ID de grupo.
- Información del usuario (comentario).
- Directorio home del usuario.
- Shell por defecto del usuario.

Los registros del fichero **/etc/group** contienen los datos:

```
$ grep jboss /etc/group
jboss::102:gustavo,root
```

- Nombre del grupo.
- Contraseña de grupo. Suele estar vacío.
- ID de grupo.
- Lista de usuarios que pertenecen al grupo.

I.2. Comandos básicos en la línea de comandos.

Comando **pwd**

✓ Muestra el path o ruta absoluta al directorio activo.

```
$ pwd
/export/home/gustavo
```

➔ Salida por pantalla de la ruta absoluta al directorio activo o actual.

Comando **cd**

✓ Cambia el directorio de trabajo.
✓ Sin parámetros cambia al directorio home del usuario.
✓ La opción "**..**" cambia el directorio activo al directorio inmediatamente anterior, jerárquicamente hablando.
✓ Ejecutado con el modificador "**-**" cambia al anterior directorio activo.

```
$ cd /
$ cd /export/home
$ cd -
$pwd
/
```

➔ El primer comando cambia el directorio activo al directorio raíz.
El segundo pasa al directorio **/export/home** como activo.
Posteriormente, con la ejecución de cd - se cambia al directorio activo anterior que, como demuestra la salida del comando **pwd** es de nuevo el directorio raíz.

Comando **ls**

✓ Para cada nombre de directorio, lista el contenido del mismo; para cada nombre de fichero, indica su nombre y datos.
✓ Por defecto la salida está ordenada alfabéticamente.

-a	Lista todos los archivos incluso los no visibles.
-F	Indica el tipo de elemento: directorio (**/**), ejecutable (*****), puerta (**>**), socket (**=**), fifo (**|**).
-d	Lista sólo directorios como ficheros sin mostrar su contenido. Por ejemplo, para listar los directorios existentes en el directorio activo: **ls -d */**
-k	Indica el tamaño en kilobytes.
-l	Lista en formato largo.
-r	Invierte el ordenamiento.
-i	Muestra el número de inodo.
-h	Muestra el tamaño de los ficheros y directorios con su unidad. Sin esta opción, la unidad por defecto es bytes.
-R	Lista recursivamente los directorios.
-t	Ordena por fecha de última modificación: de más reciente a más antiguo.

```
$ ls -dFtr /*
/lost+found/  /mnt/         /sbin/       /platform/   /kernel/     /var/
/Desktop/     /bin/         /net/        /vol/        /dev/        /etc/
/export/      /system/      /usr/        /lib/        /boot/       /Documents/
/opt/         /devices/     /home/       /cdrom/      /tmp/        /proc/
```

➔ Se listan los directorios del directorio raíz "/" ordenados por última fecha de modificación de más antiguo a más actual. Además se muestra, junto a los nombres, el carácter "/" identificativo del tipo de elemento directorio.

Comando **hostname**

✓ Sin argumento muestra el nombre de la máquina.
✓ Con argumento fija el nombre pasado como nombre de la máquina.

```
$ hostname
Solaris10host
```

➔ Salida por pantalla del nombre del host,

✓ Para que la modificación del nombre del host sea permanente será necesario además añadir el nombre del host en el fichero **/etc/hosts** y **/etc/nodename**.

Comando **uname**

✓ Proporciona el nombre del sistema y otros datos.

-a	Muestra toda la información disponible.
-n	Muestra el nombre del equipo (nodo) en la red.

-m	Arquitectura de la máquina (i86pc en el ejemplo).
-s	Nombre del sistema operativo ("SunOS" en el ejemplo).
-r	Número de versión del sistema operativo ("5.10" en el ejemplo).
-v	Versión del kernel ("Generic_147148-26" en el ejemplo).

```
$ uname -a
SunOS Solaris10host 5.10 Generic_147148-26 i86pc i386 i86pc
```

➔ Salida completa del comando: nombres de sistema, del nodo en la red, de versión del sistema y del kernel y de la arquitectura del equipo anfitrión.

Comando **swap**

✓ Administración de la memoria de intercambio o **swap**.

-l	Lista las áreas de memoria swap y sus datos.
-s	Tamaño total y ocupación de memoria swap.
-a	Crea un área swap sobre la ruta de dispositivo indicada.
-d	Elimina un área swap.

```
# swap -l
swapfile             dev  swaplo bloques   libre
/dev/dsk/c0t0d0s1   33,1       8 1076344 1076344
# swap -d /dev/dsk/c0t0d0s1
/dev/dsk/c0t0d0s1 era un dispositivo de volcado --
invocando dumpadm(1M) -d swap para seleccionar un nuevo dispositivo de volcado
dumpadm: no hay ningún dispositivo de intercambio disponible
# swap -a /dev/dsk/c0t0d0s1
anteriormente se había inhabilitado el volcado por caída del sistema operativo --
invocando dumpadm(1M) -d swap para seleccionar un nuevo dispositivo de volcado
```

➔ Secuencialmente, se listan las particiones de memoria de intercambio o swap en el sistema, se elimina la partición swap **/dev/dsk/c0t0d0sI** y, finalmente, vuelve a crearse el área eliminada. En el caso del ejemplo, sólo existe un área swap, por lo que el comando devuelve warnings relativos a esta situación.

Comando **df**

✓ Muestra la cantidad de espacio ocupado y disponible en los sistemas de ficheros del disco.
✓ Por defecto, proporciona los valores en kilobytes.

-a	Incluye los sistemas de ficheros virtuales.
-h	Muestra las ocupaciones en unidades fácilmente legibles.
-l	Se omiten los sistemas de ficheros en red.
-F tipo_filesystem	Para limitar la información a un determinado tipo de sistema de ficheros.

```
$ df -h -F ufs
Sistema de archivos  tamaño usados aprovechar capacidad Montado en
/dev/dsk/c0t0d0s0      8,8G   4,1G   4,6G     48%    /
/dev/dsk/c0t0d0s7      6,4G   6,5M   6,4G      1%    /export/home
```

➔ Se muestran las particiones de discos del sistema que tengan sistema de ficheros UFS. Las medidas de ocupación se representan en unidades fácilmente legibles.

Comando **du**

- ✓ Sirve para monitorizar el uso de disco por directorios.
- ✓ Busca de manera recursiva los directorios especificados e informa del espacio que ocupa cada uno.
- ✓ Por defecto, proporciona los valores en kilobytes.

-a	Muestra, además del espacio ocupado por los directorios, el de los ficheros individuales.
-h	Muestra las ocupaciones en unidades fácilmente legibles.
-s	Muestra la ocupación total de los ficheros o directorios especificados como parámetro.

```
$ du -sh .
 146M   .
```

➔ Ocupación de memoria en el directorio activo y en unidad fácilmente legible.

Comando **uptime**

- ✓ Indica, entre otros datos, cuánto tiempo lleva el sistema corriendo.
- ✓ Por ejemplo, para la siguiente salida del comando:

```
$ uptime
7:47pm  en funcionamiento  1:14,  2 usuarios,  promedio de carga: 0,00, 0,00,
0,00
```

➔ Los datos proporcionados son, de izquierda a derecha:
- Hora actual.
- Tiempo que ha permanecido el equipo conectado ininterrumpidamente.
- Número de usuarios conectados en el momento.
- Carga del sistema. Carga promedio del sistema en los últimos 1, 5 y 15 minutos.

1.3. Acceder a la documentación en línea.

Los ficheros de la documentación en línea del sistema se localizan, habitualmente, en el directorio **/usr/share/man**.
El manual en línea estructura su contenido en las siguientes secciones según la temática tratada:

1	Comandos generales.
2	Llamadas al sistema.
3	Biblioteca C de funciones.
4	Ficheros especiales y drivers.
5	Formatos de fichero y convenciones.
6	Juegos y salvapantallas.
7	Miscelánea.
8	Comandos de administración del sistema y demonios.

El diseño de los artículos presenta una estructura común conteniendo los apartados:

NAME	Nombre del comando y una descripción breve y genérica de su función.
SYNOPSIS	Sintaxis para la ejecución y enumeración de las posibles opciones o modificadores.
DESCRIPTION	Detalle de la funcionabilidad del comando y de cada una de sus opciones.
EXAMPLES	Ejemplos de uso.
SEE ALSO	Referencia a otros comandos o funciones relacionados con el consultado.

En algunos artículos aparecen, además, otros apartados como OPTIONS, EXIT STATUS, ENVIRONMENT, BUGS, FILES, AUTHOR, REPORTING BUGS, HISTORY o COPYRIGHT

Comando **man**

- ✓ Muestra las páginas del manual en línea.
- ✓ Seguido de un nombre de comando muestra las páginas del manual sobre el mismo.

-a patrón	Muestra todas las páginas del manual en las que existe alguna coincidencia con el patrón pasado como argumento.
-k patrón	Lista las páginas del manual que contienen entradas relacionadas con el texto. Equivalente al comando **apropos**.
-f fichero	El paso de argumentos para el comando se hace a través de uno o varios ficheros.
-l patrón	Lista las páginas del manual en las que existen coincidencias con el patrón proporcionado.
-s N	Permite indicar la sección del artículo a la que se quiere acceder.

- ✓ Para salir de la ejecución de **man** se puede emplear la tecla de escape "**q**" o enviar señal de terminación al proceso con la combinación **Control+c**.

2. Ver ficheros y directorios.

2.1. Usar comandos para crear, mover, copiar, renombrar y eliminar ficheros y directorios.

Comando **cp**

- ✓ Copia fichero de origen como fichero de destino.
- ✓ Se puede indicar únicamente el directorio de destino. En este caso, el fichero copiado tendrá el mismo nombre que el fichero de origen.

-f	Sobrescribe directorios y ficheros destino si existen, sin mostrar advertencia.
-i	Muestra aviso antes de sobrescribir directorios o ficheros existentes.
-p	Mantiene en destino el usuario y grupo propietarios, permisos y fecha de creación/modificación.
-r -R	Recursivo: copia directorios y su contenido.

```
# ls -l /export/home/gustavo/prueba
-rw-r--r--   1 gustavo  gustavo        0 jul 28 10:44 /export/home/gustavo/prueba
# cp -p /export/home/gustavo/prueba /export/home
# ls -l /export/home/prueba
-rw-r--r--   1 gustavo  gustavo        0 jul 28 10:44 /export/home/prueba
```

➔ Copia del fichero "prueba", conservando datos de propiedad, fechas y permisos, contenido en el directorio **home** del usuario "gustavo" al directorio /home.

Comando **mv**

- ✓ Permite cambiar el nombre de un fichero o mover ficheros desde una ubicación origen a otra ubicación destino.

-f	Sobrescribe ficheros destino con el mismo nombre si existen.
-i	Avisa antes de sobrescribir ficheros con el mismo nombre.

```
$ mv /home/gustavo/prueba .
$ ls -l prueba
-rw-r--r--   1 gustavo  gustavo        0 jul 28 10:46 /export/home/gustavo/prueba
```

➔ En el ejemplo se mueve el fichero "prueba" del directorio home del usuario "gustavo" al directorio activo. La ejecución posterior del comando **ls** demuestra la nueva ubicación de dicho fichero en el directorio activo.

Comando **rm**

✓ Elimina los ficheros o directorios indicados como parámetro.

-f	Ignora ficheros inexistentes pasados como parámetro y nunca pide confirmación.
-i	Muestra aviso de confirmación antes de eliminar cada fichero.
-r	Recursivo: borra directorios y su contenido.

```
$ rm prueba
$ ls -l prueba
prueba: No existe tal archivo o directorio
```

➔ Borrado del fichero "prueba" del directorio activo. Tras el borrado el comando **ls** reporta error por no encontrase el elemento.

Comando **touch**

✓ Por defecto, actualiza al momento presente la fecha y hora de creación/modificación de los ficheros indicados.
✓ Si los nombres proporcionados no corresponden a ficheros existentes en la ubicación, el comando touch los creará.

```
$ touch -t 198105070630 /export/home/gustavo/prueba
$ ls -l /export/home/gustavo/prueba
-rw-r--r--   1 gustavo  gustavo        0 may  7  1981 /export/home/gustavo/prueba
```

➔ Creación del fichero "prueba" en el directorio home del usuario "gustavo" con fecha de última modificación del 07/05/1981 06:30.

-c	Si el fichero no existe, no lo crea.
-t YYYYMMDDhhmm	Crea el fichero o actualiza uno existente con la fecha de última modificación indicada.

✓ La máscara "YYYYMMDDhhmm" para la fecha y hora se desglosa como:
- YYYY. Año en cuatro dígitos.
- MM. Mes con dos dígitos. Se antepone el valor cero para los meses anteriores a octubre.
- DD. Día del mes con dos dígitos. Se antepone el valor cero para los días anteriores al 10.
- hh. Hora con dos dígitos y en formato 24h.
- mm. Minuto con dos dígitos. Igualmente, se antepondrá el valor cero en los casos en que proceda.

Comando **mkdir**

✓ Crea directorios con los nombres indicados como parámetros.

-p	Crea el directorio y todos los directorios padre anteriores indicados en la ruta.

```
$ ls -d /export/home/gustavo/directorio/inexistente
/export/home/gustavo/directorio/inexistente: No existe tal archivo o directorio
$ mkdir -p /export/home/gustavo/directorio/inexistente
$ ls -d /export/home/gustavo/directorio/inexistente
/export/home/gustavo/directorio/inexistente
```

➔ El primer comando muestra que la ruta pasada como argumento no existe en el sistema. Con la opción **-p** del comando **mkdir** se crea en el segundo comando, como demuestra la ejecución del tercero.

Comando `rmdir`

✓ Elimina los directorios, que no contengan ningún elemento, pasados como parámetros al comando.

```
$ ls -l /export/home/gustavo/directorio
total 0
$ rmdir /export/home/gustavo/directorio
$
```

➔ Se muestra que el directorio "/export/home/gustavo/directorio" no contiene elementos y, a continuación, se elimina dicho directorio con el comando **rmdir**.

2.2. Edición de ficheros con el editor Vi y personalización de sesión.

En la apertura de ficheros desde el editor, se pueden indicar las opciones:

-R	El editor abre el fichero en modo solo lectura.
-r	Abre la edición del fichero después de un fallo del editor o del sistema: recupera la versión del fichero almacenada en el buffer.

Es posible crear una sesión personalizada de Vi. Para ello es necesario crear un fichero con extensión **.exrc** en el directorio **home** del usuario con las distintas variables de la sesión.

Vi presenta tres modos de ejecución posibles:
- Modo comando. Es el modo por defecto. Se entra en modo comando con la tecla Esc.
- Modo de inserción. Se emplea para introducir texto en el documento. Se entra en modo de inserción con las teclas "i" o "a".
- Modo línea de comandos. En este modo, una línea en la parte inferior de la pantalla del editor permite insertar una serie de comandos especiales (para realizar búsquedas, guardar, salir, modificar configuraciones de Vi, etc.) Se entra en modo comando con la tecla ":".

Comandos de inserción de texto.

i	Permite insertar texto desde la posición actual del cursor.
a	Permite insertar texto después de la posición actual del cursor.
I	Permite insertar texto desde el principio de la línea en la que se encuentra el cursor.
A	Permite insertar texto desde el final de la línea en la que se encuentra el cursor.
o	Agrega una línea en blanco debajo de la línea en que se encuentra el cursor y entra en modo inserción en ella.
O	Agrega una línea en blanco encima de la línea en que se encuentra el cursor y entra en modo inserción en ella.

Comandos de borrado de texto.

x	Elimina el carácter sobre el que se encuentra el cursor.
X	Elimina el carácter previo a la posición en que se encuentra el cursor.
dd	Elimina la línea sobre la que se encuentra el cursor.
Ndd	Elimina las N líneas posteriores a la posición actual del cursor.
dw	Borra todos los caracteres desde la posición actual del cursor hasta el final de la palabra sobre la que se encuentra el mismo.
D d$	Borra todos los caracteres desde la posición actual del cursor hasta el final de la línea.
d0	Borra todos los caracteres desde el principio de la línea actual hasta la posición del cursor.
dG	Borra desde la línea actual hasta el final del documento.
d1G	Borra desde el principio del documento hasta la línea actual.
u	Deshace la última acción realizada.

Comandos de sustitución de texto.

r+carácter	Sustituye el carácter sobre el que se encuentra el cursor por el carácter pasado.
c$	Sustituye el texto a partir de la posición del cursor, y hacia la derecha, por el nuevo texto que se inserte.
C0	Sustituye el texto a partir de la posición del cursor, y hacia la izquierda, por el nuevo texto que se inserte.
~	Cambia de mayúscula a minúscula (y viceversa) el carácter sobre el que se encuentra el cursor.
:%s/original/nuevo	Sustituye todas las ocurrencias de la cadena “original” por “nuevo”.

Comandos de desplazamiento del cursor.

h	Mueve el cursor a la izquierda.
j	Mueve el cursor hacia abajo.

k	Mueve el cursor hacia arriba.
l	Mueve el cursor a la derecha.
$	Mueve el cursor al último carácter de la línea actual.
0	Mueve el cursor al primer carácter de la línea actual.
w	Mueve el cursor al primer carácter de la siguiente palabra.
G	Mueve el cursor a la última línea del documento.
:$	
:N	Desplaza el cursor a la línea número N del documento.

Comandos para copiar texto.

yy	Copia el texto de la línea en que se encuentra el cursor al buffer.
Nyy	Copia las N líneas posteriores a la posición del cursor.
p	Pega el texto almacenado en el buffer a la derecha de la posición actual del cursor.
P	Pega el texto almacenado en el buffer a la izquierda de la posición actual del cursor.

Ejecución de comandos externos.

:!comando	Ejecuta el comando indicado desde el editor. Por ejemplo: `:!ls -lrt`

```
~
~
~
~
~
~
:!ls -lrt
total 0
-rw-r--r--   1 gustavo  gustavo        0 may  7  1981 prueba
[Presionar Intro para continuar]
```

➔ Desde el editor Vi se ejecuta el comando **ls -lrt**.

Comandos para búsqueda de texto.

/texto	Busca desde la derecha del cursor y hacia el final del documento la cadena "texto".
?texto	Busca desde la derecha del cursor y hacia el principio del documento la cadena "texto".
n	Se traslada a la siguiente ocurrencia de la cadena buscada hacia el final del documento.
N	Se traslada a la siguiente ocurrencia de la cadena buscada hacia el principio del documento.

Comandos de salida y salvado del documento.

:q :quit	Sale del editor si no se han realizado cambios.
:q! :quit!	Sale del editor sin guardar los cambios en el documento editado o, sin guardar el nuevo texto en un nuevo fichero.
:w	Guarda los cambios realizados en un fichero existente.
:wq	Guarda los cambios realizados en un fichero existente y sale del editor.
:w nombre	Guarda el contenido en un fichero con el nombre indicado.
:wq nombre :x nombre	Guarda el contenido en un fichero con el nombre indicado y sale del editor.
:ZQ	Sale del editor sin guardar.
:ZZ	Sale del editor guardando los cambios sobre un fichero ya existente.

Otros comandos.

:set all	Muestra todas las opciones de set posibles.
:set nu	Muestra los números de línea del texto.
:set nonu	Oculta los números de línea.
:set ic	Las búsquedas no diferenciarán entre mayúsculas y minúsculas.
:set noic	Las búsquedas diferencian entre mayúsculas y minúsculas.
:set list	Muestra caracteres especiales o no imprimibles: tabulaciones (^I) y finales de línea ($).
:set nolist	Revierte la visualización de los caracteres espaciales de tabulación y fin de línea.
:set ro	Cambia el fichero a de solo lectura.
:set noro	Deshace el cambio a fichero de solo lectura.
:set showmode	Indica, en la parte inferior de la pantalla, si se emplea el modo "insertar", "cambiar" o "anexar".
:N1, N2 co N3	Copia el párrafo comprendido entre las líneas N1 y N2 y lo copia tras la línea N3.
:., N2 co N3	Copia el párrafo comprendido entre la línea en la que se encuentra el cursor y la línea N2 y lo copia tras la línea N3.
:N1, $ co N3	Copia el párrafo comprendido entre la línea N1 y el final del fichero y lo copia tras la línea N3.
N1, N2 m N3	Corta el párrafo comprendido entre las líneas N1 y N2 y lo pega tras la línea N3.
N1, N2 d	Elimina las líneas comprendidas entre las líneas N1 y N2.
:N r fichero	Inserta, a partir de la línea N, el contenido del fichero indicado.

```
~
~
~
~
~
~
~
~
~
:set all
noautoindent            nomodelines                 noshowmode
autoprint               nonumber                    noslowopen
noautowrite             nonovice                    tabstop=8
nobeautify              nooptimize                  taglength=0
directory=/var/tmp      paragraphs=IPLPPPQPP LIpplpipnpptags=tags /usr/lib/tags
noedcompatible          prompt                      tagstack
noerrorbells            noreadonly                  term=xterm
noexrc                  redraw                      noterse
flash                   remap                       timeout
hardtabs=8              report=5                    ttytype=xterm
noignorecase            scroll=20                   warn
nolisp                  sections=NHSHH HUuhsh+c     window=40
nolist                  shell=/sbin/sh              wrapscan
magic                   shiftwidth=8                wrapmargin=0
mesg                    noshowmatch                 nowriteany
[Presionar Intro para continuar]
```

➔ Listado de las opciones posibles de **set** con la ejecución del comando **:set all**.

```
~
~
~
~
~
~
:1 r /etc/release
```

```
                    Oracle Solaris 10 1/13 s10x_u11wos_24a X86
  copyright (c) 1983, 2013, Oracle and/or its affiliates. All rights reserved.
                          aSSEMBLed 17 January 2013
~
~
~
~
```

➔ Inserción del contenido del fichero **/etc/release** en el editor mediante la ejecución del comando de la primera captura.

2.3. Comandos de la shell, expresiones regulares, variables de la shell de Korn y ficheros de inicialización de usuario.

Operadores de redirección.

La ejecución de los comandos en la shell, comprenden tres tipos de flujo:

- Entrada estándar (**stdin**). En la mayoría de los casos, la información que reciben los programas de la entrada estándar llega desde el teclado.
- Salida estándar (**stdout**). La salida de los programas se suele mostrar a los usuarios en modo texto, a pantalla completa, o bien por una ventana GUI.
- Error estándar (**stderr**). Flujo de salida con información de alta prioridad como los mensajes de error.

Estos flujos pueden redirigirse empleando los siguientes operadores:

comando > fichero	Crea un nuevo fichero con la salida estándar. Si el fichero existe lo sobrescribe.
comando >> fichero	Añade la salida estándar al fichero indicado.
comando 2> fichero	Crea un nuevo fichero con la salida de error. Si el fichero existe lo sobrescribe.
comando 2>> fichero	Añade el error estándar al fichero indicado.
comando &> fichero	Crea un nuevo fichero con la salida estándar y de error. Si el fichero existe lo sobrescribe.
comando< fichero	Envía el contenido del fichero indicado para utilizarlo como entrada estándar.
comando << fichero	Utiliza el texto de las siguientes líneas que se inserten como entrada estándar.
comando1 | comando2	Envía el resultado de un comando a otro para que procese el mismo. El orden en la ejecución de los comandos sigue el sentido de izquierda a derecha.

Existe en UNIX un *device* con fichero de dispositivo **/devices/pseudo/mm@0:null** (enlazado desde **/dev/null**) que permite la redirección de la salida de comando a nulo.

```
$ cat /etc/release >/dev/null
$
```

➔ El operador de redirección provocará que no se muestre por pantalla el contenido del fichero indicado.

Es posible redirigir de un flujo a otro. Para ello se indica el flujo origen seguido del operador de redirección y el flujo destino precedido de "**&**".

```
$ ls -z / > salida
ls: opción no permitida -- z
sintaxis: ls -1RaAdCxmnlhogrtuvVcpFbqisfHLeE@ [archivos]
$ ls -z / > salida 2>&1
$ cat salida
ls: opción no permitida -- z
sintaxis: ls -1RaAdCxmnlhogrtuvVcpFbqisfHLeE@ [archivos]
```

➔ El primer comando emplea una opción no admitida, por lo que arroja error por pantalla (salida estándar de error) y no redirige ningún contenido al fichero "salida". En el segundo comando, se redirige la salida de error a la salida que en este caso es el fichero pasado como parámetro. Por este motivo, el contenido del fichero "salida", tras la ejecución del segundo comando, será el texto de error.

Expresiones regulares.

Una expresión regular es un patrón que define a un conjunto de cadenas de caracteres.

Elemento	Función	Ejemplo
[]	Cualquiera de los caracteres contenidos.	`a[abcd]z`
-	Cualquiera de los caracteres comprendidos en el rango.	`a[a-d]z`
.	Cualquier carácter único.	`a.z`
^	Comienzo de línea.	`^a`
$	Final de línea.	`z$`
*	Cero o más ocurrencias de cualquier carácter.	`ast*sco`
?	Cero o una ocurrencia de cualquier carácter.	`interr?gacion`
+	Una o más ocurrencias del patrón indicado.	`m[a]+s`
|	Separa varias posibles coincidencias (or).	`cad1|cad2|cad3`
\	Convierte caracteres especiales en literales.	`fichero\.txt`

{N,M}	El carácter se repite entre N y M veces.	`h[o]{1,3}la` (podrá tomar los valores "hola", "hoola" u "hooola")
{N}	El carácter se repite N veces.	`h[o]{1}la` (toma el valor "hola")
{N,}	El carácter se repite N o más veces.	`h[o]{3,}la` (podrá tomar los valores "hooola", "hoooola"...)

Variables de entorno.

Las variables de entorno contienen información a la que se accede a través de su nombre. Se encuentran disponibles por cualquier programa que se ejecute en la sesión.
Para definirlas se utiliza el operador de asignación "=".
Con el comando **export** la variable pasa de ser local a variable de entorno.

Las variables de entorno más relevantes son:

$PATH	Contiene una lista separada por dos puntos de directorios en los cuales el intérprete de comandos buscará los archivos ejecutables que no se invocan con una ruta.
$HOME	Contiene la ubicación del directorio de usuario.
$DISPLAY	Contiene el identificador del display que los programas de X11 deben usar por defecto.
$LANG	Estándar de idioma.
$LC_ALL	Estándar de valores locales (moneda, hora, etc.)
$LOGNAME	Muestra el nombre del usuario con el que se ha logado en el sistema.
$TZ	Zona horaria.
$RANDOM	Es una variable de entorno especial que, almacena un valor numérico aleatorio.
$PWD	Contiene la ruta del directorio actual.
$OLDPWD	Contiene el directorio, que fue activo, anterior al actual.
$SHELL	Contiene el nombre de la shell que está corriendo.
$TERM	Contiene el nombre de la terminal que está corriendo.
$EDITOR	Contiene la ruta al editor de texto, usualmente un editor liviano.
$VISUAL	Contiene la ruta al editor de texto, usualmente un editor como Vi o Emacs.
$MAIL	Contiene la ubicación del correo electrónico entrante.
$HISTFILE	El nombre del archivo donde se guarda el historial de comandos.
$HISTFILESIZE	El número máximo de líneas que puede contener el historial de comandos.
$HOSTNAME	Contiene el nombre de host del sistema.
$PS1	El prompt por defecto.
$USER	Nombre de usuario actual en el sistema.
$MANPATH	Cadena de texto separada por comas con las ubicaciones de las páginas del manual en línea del sistema.

Los comandos **env**, **set** y **declare**, ejecutados sin argumentos, muestran la lista de variables de entorno de la sesión.

Comando `set`

- ✓ Ejecutados sin argumentos, muestran la lista de las variables de entorno.
- ✓ También permite eliminar, añadir o modificar variables de entorno. Para ello se emplea con la sintaxis **`set var=value`**
- ✓ Además, puede activar o desactivar diversos comportamientos de la sesión de usuario. Las opciones más destacables son:

allexport	Todas las variables que se definan se exportarán automáticamente.
bgnice	Ejecuta todas las tareas en segunda plano con la mínima prioridad.
emacs	Establece el editor Emacs en la línea de comandos.
errexit	Si un comando tiene una salida distinta de cero, se finaliza la ejecución.
gemacs	Establece el editor Gemacs en la línea de comandos.
monitor	Indica si el control de tareas (jobs) está habilitado. De no estarlo, los comandos **fg** y **bg** no tendrán efecto.
noclobber	Evita sobrescribir de forma accidental un fichero al emplear el operador de redirección ">".
noexec	Lee los comandos e indica los errores de sintaxis pero no llega a ejecutarlos.
nounset	Finaliza la ejecución de una secuencia de comandos si se intenta emplear una variable sin inicializar.
history	Permite establecer si se almacenan los comandos ejecutados.
ignoreeof	La shell no interpretará los finales de línea como final de comando, se debe emplear **exit**.
noglob	Deshabilita **globbing** (uso de expresiones regulares).
vi	Establece el editor Vi en la línea de comandos.
xtrace	Se muestra por pantalla cómo se ejecutan los comandos.

- El comando **set -o** muestra las opciones de **set** y su estado (on u off).
- Para pasar una opción a on, se emplea el comando **`set -o`** opción.
- Para pasar una opción a off, se emplea el comando **`set +o`** opción.

```
$ set -o
allexport       off
braceexpand     on
emacs           on
errexit         off
errtrace        off
functrace       off
hashall         on
histexpand      on
history         on
ignoreeof       off
interactive-comments    on
keyword         off
monitor         on
noclobber       off
noexec          off
noglob          off
nolog           off
notify          off
nounset         off
onecmd          off
physical        off
pipefail        off
posix           off
privileged      off
verbose         off
vi              off
xtrace          off
```

➔ Opciones de **set** establecidas en la sesión actual.

```
$ var=1
$ export var
```

➔ Declaración y asignación de la variable local "var". Posteriormente se exporta y pasa a ser variable global.

```
$ set +o emacs
$ set -o | grep emacs
emacs           off
```

➔ Cambio de la opción de edición en línea de comandos con el editor **Emacs** a **off**.

Comando **unset**

✓ Permite vaciar de contenido una variable, ya sea local o de entorno.

```
$ var=123
$ echo $var
123
$ unset var
$ echo var

$
```

➔ Declaración y asignación de la variable local "var" y salida de su contenido por pantalla. A continuación se vacía el contenido de "var" y se observa que el comando **echo** no devuelve valor.

Ficheros de inicialización del usuario.

El fichero de perfil, contenido en directorio home del usuario, se ejecuta en el inicio de sesión y contiene variables de entorno y otros parámetros para la configuración de la misma.
En la shell de Korn, **ksh**, el fichero de perfil se denomina como **.profile**.
En el caso de la shell **bash**, el fichero es **.bash_profile**.

2.4. Permisos de ficheros y directorios. Permisos básicos y ACL.

Existen tres niveles de permisos sobre un fichero o directorio:
- Permisos para el propietario del objeto.
- Permisos para el grupo al que pertenece el usuario.
- Permisos para el resto de los usuarios del sistema.

Los permisos básicos que se aplican en los tres niveles son:
- Lectura (**r**).
- Escritura (**w**).
- Ejecución (**x**).

Los permisos aplicados a ficheros posibilitan las acciones:
- Lectura. El usuario puede leer el contenido del fichero.
- Escritura. El usuario puede modificar el contenido del fichero.
- Ejecución. El usuario puede ejecutar el contenido del fichero.

Los permisos aplicados a directorios posibilitan las acciones:
- Lectura. El usuario puede leer el contenido del directorio, es decir los números de inodo y los nombres de los ficheros. Con este permiso podría listarse el contenido del directorio con el comando **ls**.
- Escritura. El usuario puede modificar el contenido del directorio, es decir, crear, renombrar y eliminar ficheros y subdirectorios dentro de él.
- Ejecución. El usuario puede acceder a los objetos que contiene. Teniendo sólo permiso de ejecución, el usuario podría ejecutar **cd** pero no **ls**.

El carácter previo a la cadena de permisos es el código de tipo de fichero. Puede ser:

-	Fichero de datos normales.
d	Directorio.
l	Enlace simbólico.
p	Pipe con nombre.
s	Socket.
c	Fichero de dispositivo de caracteres.
b	Fichero de dispositivo de bloque.
D	Door. Comunicación entre procesos.

```
$ ls -l /export/home/gustavo/
total 4
drwxr-xr-x   2 gustavo  gustavo      512 jul 28 11:51 directorio
-rw-r--r--   2 gustavo  gustavo        0 jul 28 11:52 enlace duro
lrwxrwxrwx   1 gustavo  gustavo        7 jul 28 11:52 enlace simbolico -> fichero
-rw-r--r--   2 gustavo  gustavo        0 jul 28 11:52 fichero
```

➔ Los primeros caracteres de cada línea, de arriba abajo: "**d**" (directorio), "**-**" (enlace duro a fichero), "**l**" (enlace simbólico) y "**-**" (fichero convencional)

Comando **chmod**

✓ Permite cambiar los permisos de acceso a los ficheros y directorios indicados.

-v	Lista los archivos y directorios a los que se les va aplicando.
-R	Recursivo: cambia permisos de directorio y de sus ficheros y subdirectorios contenidos.

✓ Para modificar los permisos sobre un elemento existen dos modos:
- Modo carácter. Se indican las entidades y los permisos que se otorgan (**+**), deniegan (**-**) o establecen (**=**). Se distinguen cuatro entidades en cuanto a permisos sobre un elemento: usuario (**u**), grupo (**g**), otros (**o**) y todos (**a**); y tres tipos de permisos: lectura (**r**), escritura (**w**) y ejecución (**x**).
  chmod [u,g,o,a] [+,-,=] [r,w,x] elemento

- Modo octal. Los permisos se representan por un valor numérico de tres dígitos. El primer dígito, de izquierda a derecha, representa los permisos de usuario, el segundo dígito representa los permisos de grupo y el tercero los de otros.
  Cada dígito es la suma decimal de los siguientes valores según los permisos que se vayan a otorgar:
  chmod digito_usuario + digito_grupo + digito_otros elemento

Permiso	Binario	Decimal
Lectura	001	2^0
Escritura	010	2^1
Ejecución	100	2^2

```
$ touch prueba
$ ls -l prueba
-rw-r--r--   1 gustavo     gustavo           0 ago  9 09:50 prueba
$ chmod g+wx,o+w prueba
$ ls -l prueba
-rw-rwxrw-   1 gustavo     gustavo           0 ago  9 09:50 prueba
```

➔ Generación de fichero "prueba" con los permisos que se establecen con los valores por defecto de la máscara. Se añaden, en modo carácter, permisos de escritura y ejecución a nivel "grupo", y de escritura a "otros".

```
$ umask
022
$ chmod 644 prueba
$ ls -l prueba
-rw-r--r--   1 gustavo     gustavo           0 ago  9 09:50 prueba
```

➔ A continuación, y sobre el fichero del ejemplo anterior, se restablecen los permisos por defecto empleando la notación octal. Para la obtención del valor octal a indicar, se resta a 666 el valor 022 de la máscara.

Los bits de permisos especiales contemplan tres opciones:

- SUID (set user ID). Le indica al sistema que ejecute el programa con los permisos del propietario del fichero. Se indica mediante una "**s**" en la posición del bit de ejecución del propietario.
  chmod u+s fichero
  chmod 4??? fichero

- SGID (set group ID). Le indica a Linux que ejecute el programa con los permisos del grupo del propietario del fichero. Se indica mediante una "**s**" en la posición del bit de ejecución del grupo.
  chmod g+s fichero
  chmod 2??? fichero

- Sticky bit. Aplicado a un directorio, indica que un fichero contenido en él sólo podrá ser eliminado o renombrado por su propietario o el usuario **root**. Se indica mediante una "**t**" en la posición del bit de ejecución de otros usuarios.
  chmod o+t fichero
  chmod 1??? fichero

Establecer el SUID o el SGID tiene sentido únicamente si se han establecido los permisos de ejecución previamente. Si no se ha establecido el permiso de ejecución, se sustituye la "**s**" por "**S**".

```
$ touch prueba
$ chmod a+x prueba
$ chmod u+s,g+s prueba
$ ls -l prueba
-rwsr-sr-x   1 gustavo  gustavo        0 ago 14 23:28 prueba
$ chmod 0755 prueba
$ ls -l prueba
-rwxr-xr-x   1 gustavo  gustavo        0 ago 14 23:36 prueb
```

➔ Se crea un nuevo fichero "prueba", se le otorga permiso de ejecución a todos los niveles en modo carácter, y, posteriormente, los permisos especiales SUID y GUID.
Tras mostrar con el comando **ls** cómo quedan los permisos del fichero, se eliminan los permisos especiales empleando la notación octal.

Comando **umask**

- ✓ Si no se le pasa ningún parámetro, muestra la máscara de usuario para permisos de creación. Si se le pasa un parámetro, establece el valor de éste como nueva máscara.
- ✓ La propiedad de un nuevo fichero o directorio se establece por la diferencia entre el valor octal 777 en directorios, y 666 en ficheros, y el valor de la máscara.
- ✓ Usualmente, para un fichero nuevo se le asignan los permisos **rw-r--r--** (644), mientras que para un directorio nuevo **rwxr-xr-x** (755).

```
$ umask
0022
$ umask 0020
$ umask
0020
```

➔ El primer comando muestra el valor actual de la máscara de permisos. En el segundo, se establece el valor de la máscara a "0020" y, con el tercer comando, se observa que el dato ha sido modificado.

Comando **chown**

- ✓ Permite cambiar el propietario de un fichero o directorio.
- ✓ El usuario propietario de un fichero sólo puede ser modificado por el usuario **root**.
- ✓ El grupo propietario puede ser modificado, además de por el usuario **root**, por un usuario que pertenezca al grupo actual del fichero y al grupo al que se quiere cambiar.

chown [nuevo_propietario:nuevo_grupo] fichero

```
# ls -l prueba
-rw-r--r--   2 gustavo  gustavo        0 jul 28 11:52 prueba
# chown root:root prueba
# ls -l prueba
-rw-r--r--   2 root     root           0 jul 28 11:52 prueba
```

➔ Cambio de usuario y grupo propietario del fichero "prueba" a usuario **root** y grupo **root**.

Comando **chgrp**

- ✓ Permite cambiar el grupo propietario de un fichero o directorio.
- ✓ El grupo propietario de un fichero puede ser modificado por el usuario **root** y por cualquier otro usuario si pertenece al grupo actual y al grupo que se quiere establecer como nuevo propietario.

```
chgrp nuevo_grupo fichero
```

```
# ls -l prueba
-rw-r--r--   2 gustavo  gustavo        0 jul 28 11:52 prueba
# chgrp root prueba
# ls -l prueba
-rw-r--r--   2 gustavo  root           0 jul 28 11:52 prueba
```

➔ Cambio de grupo propietario del fichero "prueba" a **root**.

2.5. Visualizar, configurar y eliminar listas de control de acceso en ficheros y directorios.

Las listas de control de acceso (ACL) proveen de un nivel adicional de seguridad a los ficheros, extendiendo el clásico esquema de permisos en Unix, al permitir asignar permisos a usuarios o grupos concretos.
Cuando existen permisos en la ACL de un elemento, en la columna de permisos de la salida del comando **ls**, se muestra el símbolo "**+**".

```
$ ls -l prueba
-rw-r--r--+  1 gustavo  gustavo        0 sep 28 23:46 prueba
```

El comando **ls** con la opción **-v** devuelve la información de la ACL.

```
$ ls -v prueba
-rw-r--r--+  1 gustavo  gustavo        0 sep 28 23:46 prueba
     0:user::rw-
     1:user:gustavo:rwx          #effective:r--
     2:group::r--                #effective:r--
     3:mask:r--
     4:other:r--
```

Comando **getfacl**

- ✓ Muestra la lista de control de acceso de un fichero o directorio que se le pasa como argumento.
- ✓ Los tres primeros valores se corresponden al nombre del elemento, a su usuario y grupo propietario.
- ✓ A continuación se detallan los permisos ACL para las entidades user, group, mask y other.

```
$ getfacl prueba

# file: prueba
# owner: gustavo
# group: gustavo
user::rw-
group::r--                  #effective:r--
mask:r--
other:r--
```

Comando **setfacl**

- ✓ Permite modificar la lista de acceso de un fichero o directorio.
- ✓ Debe ir acompañado de una de las siguientes opciones que indican la acción a realizar sobre los permisos de la ACL:

-s	Establece todos los permisos de la ACL.
-m	Modifica la ACL con los datos otorgados.
-d	Elimina las entradas de la ACL indicadas.
-R	Para emplear recursividad.

- ✓ Además de las opciones anteriores, se detalla a continuación de las mismas las entidades sobre las que la acción sobre la ACL tendrá efecto:

u::permisos	Otorga o revoca los permisos a nivel de usuario.
u:uid:permisos	Otorga o revoca los permisos a un usuario específico identificado por su UID o nombre de usuario.
g::permisos	Otorga o revoca los permisos a nivel de grupo.
g:gid:permisos	Otorga o revoca los permisos a un grupo específico identificado por su UID o nombre de usuario.
o:permisos	Otorga o revoca los permisos a los usuarios que no pertenecen al usuario o grupo propietario.
m:permisos	Indica los permisos máximos permitidos para los usuarios (que no sea el propietario) y para grupos. La máscara es una forma rápida de cambiar los permisos de todos los usuarios y grupos.
d:u::permisos	Establece al usuario propietario los permisos por defecto.
d:g::permisos	Establece al grupo propietario los permisos por defecto.

➔ En el siguiente ejemplo, se establecen los permisos sobre el fichero "prueba" para las entidades genéricas usuario, grupo y otros y, asimismo, los permisos específicos para el usuario "gustavo". Se muestra la ACL del fichero tras el establecimiento de permisos.

```
$ setfacl -s u::rw-,g::r--,o:r--,u:gustavo:rwx,m:r-- prueba
$ getfacl prueba

# file: prueba
# owner: gustavo
# group: gustavo
user::rw-
user:gustavo:rwx                #effective:r--
group::r--              #effective:r--
mask:r--
other:r--
```

➔ A continuación, y sobre los permisos del ejemplo anterior, se eliminan los permisos específicos de la ACL para el usuario "gustavo". Se muestra la ACL del fichero tras la modificación.

```
$ setfacl -d u:gustavo prueba
$ getfacl prueba

# file: prueba
# owner: gustavo
# group: gustavo
user::rw-
group::r--              #effective:r--
mask:r--
other:r--
```

3. Búsqueda de ficheros y directorios.

3.1. Comandos para búsqueda de ficheros y directorios y búsqueda de contenidos dentro de ficheros.

Comando **find**

- ✓ Se emplea para encontrar ficheros y directorios que cumplan el criterio indicado.
- ✓ La sintaxis del comando es:
  `find punto_partida opciones expresión [acción]`

- ✓ Las opciones más relevantes aceptadas son:

-atime N	Búsqueda basada en el número de días desde el último acceso. Puede verse con **`ls -lu`**.
-mtime N	Búsqueda basada en el número de días desde el último cambio en la fecha de actualización. Puede verse con **`ls -l`**.
-ctime N	Búsqueda basada en el número de días desde el último cambio de estado, como cambio en los permisos o en el propietario. Puede verse con **`ls -lc`**.
-newer fichero	Búsqueda de ficheros con fecha de creación/modificación más reciente que la del fichero indicado.
-user usuario	Búsqueda de ficheros propiedad del usuario indicado.
-user UID	Búsqueda de ficheros propiedad del usuario con el UID (ID de usuario) indicado.
-group grupo	Búsqueda de ficheros propiedad del grupo indicado.
-group GID	Búsqueda de ficheros propiedad del grupo con el GID (ID de grupo) indicado.
-name patrón	Búsqueda de ficheros cuyo nombre cumpla con el patrón proporcionado.
-size N[c]	Búsqueda de ficheros de N bloques, por defecto, o de N caracteres o bytes si se indica el parámetro c. Un bloque es igual a 512 bytes. Las búsquedas por tamaño pueden ser: -size +1 — Elementos con tamaño superior a 1 bloque. -size -1 — Elementos con tamaño inferior a 1 bloque. -size 1 — Elementos con tamaño de un 1 bloque (entre 1 y 512 bytes).
-inum N	Búsqueda del fichero con el número de inodo indicado.
-type [f,d,s,D,c,b,l]	Búsqueda por el tipo de elemento: ficheros (**f**), directorios (**d**), sockets (**s**), doors (**D**), dispositivos de carácter (**c**), dispositivos de bloque (**b**) y enlaces simbólicos (**l**).
-perm permisos	Búsqueda por los permisos otorgados al elemento. Se pueden expresar en modo octal o carácter. Si se quiere buscar los elementos que tengan al menos determinados permisos, el bloque de permisos irá precedido de un guión: Ejemplo: **`find / -perm -u=rw,g=rw -print`** Devolverá aquellos elementos cuyo usuario y grupo propietarios tengan, al menos, permisos de lectura y escritura.

- ✓ Para las opciones **atime**, **mtime** y **ctime**, la especificación del valor **N** de número de días podrá tener los siguientes comportamientos:

N	Hace exactamente N días.
-N	En los N últimos días.
+N	Hace más de N días.

✓ Es posible emplear más de una condición de búsqueda, bien sea con AND lógico (**-a**) o con OR (**-o**). En el siguiente ejemplo, el comando buscará aquellos elementos cuyo nombre sea "fichero" o cuyo número de inodo sea 222:

```
$ find / \( -name fichero -o -inum 222 \) -print
```

✓ También es posible realizar búsquedas de elementos que no cumplan alguna condición. Por ejemplo:

```
$ find / -type d \! -name Desktop -print
```

Busca directorios cuyo nombre no sea "Desktop".

✓ Las acciones posibles a ejecutar sobre los resultados de la búsqueda son:

-print	Muestra los resultados de la búsqueda. Es la acción por defecto.
-exec comando {} \;	Ejecuta el comando indicado sobre los resultados de la búsqueda.

```
$ find . -name fichero -print
./fichero
$ find . -type l -print
./enlace simbolico
lrwxrwxrwx   1 gustavo  gustavo        7 jul 28 11:52 ./enlace simbolico -> fichero
$ find . -user root -exec ls -l {} \;
-rw-r--r--   2 root     root           0 jul 28 11:52 ./fichero
-rw-r--r--   2 root     root           0 jul 28 11:52 ./enlace duro
```

➔ Tres comandos:
- Búsqueda desde el directorio activo de elementos con el nombre "fichero" y salida de resultados por pantalla.
- Búsqueda desde el directorio activo de enlaces simbólicos y salida de resultados por pantalla.
- Búsqueda de elementos desde el directorio activo que sean propiedad del usuario **root**.

Comando **grep**

✓ Recorre los ficheros indicados extrayendo las líneas que cumplen un patrón de cadena de caracteres.
✓ Su sintaxis es:
grep [opciones] expresión ficheros

✓ Las opciones más habituales del comando son:

-c	Sólo muestra la cantidad de líneas en las que existe el patrón.
-i	No diferencia entre mayúsculas y minúsculas.
-n	Indica el número de línea de cada ocurrencia.
-v	Muestra las líneas que no cumplen el patrón.
-e	Permite buscar por varios patrones (OR).
-f fichero	Se pasan los patrones de búsqueda desde un fichero (igual a **fgrep**).
-E	Para hacer búsquedas por expresiones regulares (igual a **egrep**).
-l	Muestra el nombre del fichero en el que se produce al menos una ocurrencia.

Importante: El binario **grep** de Solaris (ubicado en /bin/grep) no admite varias de las opciones que son habituales en el comando en otros sistemas UNIX.
Existe dos ubicaciones en Solaris con los binarios de comandos que cumplen los formatos abiertos (véase organización **X/Open**): **/usr/xpg4/bin** y **/usr/xpg6/bin**.
En el caso de que el comando **grep** "por defecto" no admita alguna opción deseada, se puede invocar con la ruta completa a su binario alternativo.
Esta situación es extensible a otros comandos.

```
$ grep -iE 'output|end' /var/log/postrun.log
grep: opción no permitida -- E
Sintaxis: grep -hblcnsviw patrón archivo . . .
$ /usr/xpg4/bin/grep -iE 'output|end' /var/log/postrun.log
<<< commands end
>>> Command output follows:
<<< commands end
>>> Command output follows:
<<< commands end
>>> Command output follows:
```

➔ En el primer comando, reporta error la ejecución de **grep** (/bin/grep) con las opciones para no diferenciar entre mayúsculas y minúsculas y buscar en el contenido del fichero por dos patrones ("output" o "end").
En el segundo comando se invoca al binario **grep** de **/usr/xpg4/bin,** con las mismas opciones que en la ejecución anterior, obteniéndose los resultados deseados.

3.2. Administración de procesos.

Comando **ps**

- ✓ Muestra los procesos activos en el sistema e información relativa a los mismos.
- ✓ Sin parámetros, muestra sólo los procesos que están asociados con la sesión de inicio.
- ✓ Opciones:

-ef	Muestra información completa sobre todos los procesos que se están ejecutando en el sistema.
-c	Muestra información del programador del proceso.
-l	Formato largo.

✓ La salida del comando (según las opciones especificadas) mostrará varias columnas cuya correspondencia entre el nombre y el tipo de dato, se detalla en la siguiente tabla:

S	Estado en que se encuentra el proceso. Puede ser:

R	Runnable	Se encuentra en la cola para ejecución.
S	Sleeping	En espera de un evento para ser completado.
Z	Zombie	Terminado pero no recogido el control por el proceso padre.
T	Traced	Parado, ya sea por una señal de control de trabajo o porque está siendo rastreado.
W	Waiting	A la espera de tiempo de CPU.
O	Running	El proceso se encuentra corriendo en el procesador.

UID	ID de usuario propietario del proceso.
PID	ID de proceso.
PPID	ID de proceso principal (padre).
CLS	La clase de programación a la que pertenece el proceso, como tiempo real, sistema o tiempo compartido (TS). Este campo sólo se incluye con la opción **-c**.
PRI	La prioridad de programación del subproceso del núcleo. Los números más altos indican una prioridad superior.
NI	El número de **nice** del proceso, que contribuye a su prioridad de programación. Aumentar el valor del comando **nice** de un proceso significa reducir su prioridad.
ADDR	La dirección de la estructura **proc**.
SZ	El tamaño de la dirección virtual del proceso.
WCHAN	La dirección de un evento o bloqueo para el que el proceso está inactivo.
STIME	La hora de inicio del proceso en horas, minutos y segundos.
TTY	El terminal desde el cual se inició el proceso o su proceso principal. Un signo de interrogación indica que no existe un terminal de control.
TIME	La cantidad total de tiempo de CPU utilizado por el proceso desde que comenzó.
CMD	El comando que generó el proceso.

Comando **pgrep**

✓ Examina los procesos activos en el sistema y devuelve los identificadores de proceso (PID) de aquellos cuyos atributos coincidan con los criterios especificados en la línea de comandos.

✓ Opciones:

-x	Sólo devuelve resultados si el patrón indicado coincide exactamente con el nombre del proceso.
-n	Muestra sólo el PID del más reciente (lanzado hace menos tiempo) de los procesos que coinciden con el patrón.
-U	Muestra los valores de PID de los procesos cuyo propietario es el usuario indicado.
-l	Además del PID devuelve el nombre del proceso.

```
$ pgrep ksh
1386
```

➔ Obtención del PID del proceso de la shell de Korn.

Comandos de proceso.

Puede obtenerse información detallada sobre los procesos mostrados en el directorio **/proc** mediante los comandos de proceso.
El directorio **/proc** también se conoce como el sistema de archivos de procesos (procfs). Las imágenes de procesos activos se almacenan aquí por número de ID de proceso.
La siguiente tabla muestra los comandos de proceso más habituales:

pcred PID	Muestra información de credenciales del usuario propietario del proceso: PID, UID y GID.
pldd PID	Muestra las bibliotecas dinámicas que están enlazadas a un proceso.
prun PID	Inicia un proceso parado.
pstop PID	Detiene el proceso: pasa al estado T (Traced).
ptime PID	Registra el tiempo de un proceso mediante la contabilidad según los estados.
ptree PID	Muestra los árboles del proceso que contienen el proceso.
pwait PID	Espera a que los procesos indicados terminen.
pwdx PID	Muestra el directorio de trabajo actual de un proceso.
pfiles PID	Muestra los ficheros abiertos por el proceso.

Terminación de un proceso.

Un usuario puede terminar cualquier proceso propio.
El usuario **root** puede terminar cualquier proceso del sistema, excepto los procesos con ID de proceso 0, 1, 2, 3 y 4.

Comando **kill**

- ✓ Envía señales a los procesos.
- ✓ Sintaxis:
  kill [-n° señal|-s nombre señal] PID

  Siendo PID el ID del proceso.

- ✓ Cuando no se incluye ninguna señal la señal predeterminada que se utiliza es **-15** (SIGTERM).
- ✓ Otras señales son:

1	SIGHUP
2	SIGINT
9	SIGKILL
15	SIGTERM
19	SIGSTOP

- ✓ Las señales SIGINT y SIGSTOP se pueden enviar al proceso que se está ejecutando en el momento, y desde terminal, mediante los atajos de teclado Control+c y Control+z, respectivamente.

```
$ kill -9 126
```

➔ Envío de señal de terminación (SIGKILL) al proceso con PID nº 126.

Comando **pkill**

- ✓ Es un comando de proceso con funcionalidad similar a **kill**.
- ✓ La diferencia entre ambos comandos se encuentra en el hecho de que **pkill** acepta un patrón como parámetro para identificar el nombre del proceso al que enviar la señal.
- ✓ Sintaxis:
  pkill [-nº señal|-s nombre señal] patrón

```
$ pkill ksh
```

➔ Envío de señal de terminación (SIGKILL) al proceso cuyo nombre sea "ksh".

Programación de procesos.

El demonio **cron** programa tareas del sistema según los comandos encontrados en cada archivo **crontab**.

Un archivo **crontab** consta de comandos (uno por línea) que se ejecutarán en intervalos regulares. El principio de cada línea de comando en el fichero contiene información de la fecha y hora que indicará al demonio **cron** cuándo debe ejecutar el comando.

Los campos de cada línea de **crontab** son, de izquierda a derecha:

```
minuto hora dia_mes mes dia_semana [usuario] comando
```

Según el siguiente ejemplo, se ejecutará

Los archivos **crontab** se almacenan en el directorio **/var/spool/cron/crontabs**.
El fichero **/etc/cron.d/cron.deny** incluye la lista de los usuarios que no pueden invocar a **cron**.

Comando **crontab**

- ✓ Gestiona el fichero **crontab** del usuario que lo ejecuta.
- ✓ Las opciones más relevantes del comando son:

-e	Abre el editor preestablecido para editar el fichero **crontab**.
-l	Lista el contenido del **crontab** actual del usuario.
-r	Elimina el fichero del **crontab** actual del usuario.

```
# crontab -l
#ident  "@(#)root       1.21    04/03/23 SMI"
#
# The root crontab should be used to perform accounting data collection.
#
#
10 3 * * * /usr/sbin/logadm
15 3 * * 0 /usr/lib/fs/nfs/nfsfind
30 3 * * * [ -x /usr/lib/gss/gsscred_clean ] && /usr/lib/gss/gsscred_clean
#
# The rtc command is run to adjust the real time clock if and when
# daylight savings time changes.
#
1 2 * * * [ -x /usr/sbin/rtc ] && /usr/sbin/rtc -c > /dev/null 2>&1
#10 3 * * * /usr/lib/krb5/kprop_script ___slave_kdcs___
# crontab -r
# crontab -l
crontab: no es posible abrir su archivo crontab.
```

➔ Salida de contenido de **crontab** de usuario **root**. A continuación se elimina su contenido y se observa cómo se reporta error al intentar volver a listarlo.

Comando **at**

- ✓ Permite programar tareas (comandos y scripts) para su ejecución posterior.
- ✓ Su sintaxis para planificar una tarea es:
  `at [opciones] hora[fecha]`

- ✓ Al igual que en el caso de cron, la lista de los usuarios contenida en el fichero **/etc/cron.d/at.deny** implicará que estos no puedan emplear la utilidad.
- ✓ Opciones:

-m	Envía un email al usuario una vez que el **job** ha sido ejecutado.
-l	Muestra los **jobs** actuales del usuario.
-r ID	Elimina el **job** indicado.

```
$ at 14:53
at> rm /export/home/gustavo/prueba
at> <EOT>
los comandos se ejecutarán con /bin/bash
trabajo 1453211580.a en mar ene 19 14:53:00 2016
```

➔ Creación de **job** para borrado del fichero "prueba" del directorio **home** del usuario "gustavo". El trabajo se ejecutará a las 14:53 horas del presente día.

Comando **atq**

✓ Muestra los **jobs** encolados pendientes de ejecución con información relativa a los mismos.

```
$ atq
 Rank     Execution Date     Owner      Job            Queue   Job Name
  1st   Jan 19, 2016 14:53   gustavo    1453211580.a     a     stdin
```

➔ Se muestran los datos del job programado en el anterior ejemplo.

4. Funcionalidad avanzada de la shell.

4.I. Realizar funciones avanzadas de la shell de Korn: administrar trabajos, usar alias y funciones, configurar el entorno y ejecutar scripts.

Administración de procesos.

Comandos **fg** y **bg**

- ✓ Un programa ejecutado en primer plano toma el control del terminal e impide realizar otras tareas. Si al final de un comando se escribe el símbolo "&" estaremos ejecutándolo directamente en **background**.
- ✓ Con los comandos **fg** (**foreground**) y **bg** (**background**) es posible manipular procesos, transfiriéndolos de un plano al otro. El comando **fg** permite transferir un proceso al primer plano. Por el contrario, el comando **bg** envía procesos a segundo plano.
- ✓ El parámetro de estos comandos será el símbolo "**%**" seguido del número identificador de la tarea.
- ✓ Si a los comandos **fg** o **bg** no se le pasa ningún parámetro, actuará sobre el proceso más reciente (último ejecutado).

```
# prstat -p 1 &
[1]     1741
# fg %1
prstat -p 1
   PID USERNAME  SIZE   RSS STATE  PRI NICE      TIME  CPU PROCESS/NLWP
     1 root     2556K 1220K sleep   59    0   0:00:01 0,0% init/1
```

➔ Se ejecuta en segundo plano el comando **prstat** sobre el proceso con PID **1** (**init**). Con el comando **fg** se trae el proceso de **prstat** a primer plano, obteniéndose así su resultado.

Comando **jobs**

- ✓ Lista las tareas de la sesión actual que se están ejecutando en segundo plano.
- ✓ Con la opción **-p** muestra además el PID de los procesos.

```
$  vi&
[1]     7045
$  jobs
[1] +  Running                 vi&
$  jobs -p
7045
[1] + Stopped (SIGTTOU)        vi&
```

➔ El primer comando ejecuta el editor Vi en segundo plano.
La salida de **jobs** muestra el proceso en segundo plano de Vi y, al ejecutar el comando **jobs** con el parámetro **-p** se reporta además el PID del proceso.

Comando **nohup**

- ✓ Permite mantener la ejecución, en segundo plano, de un comando o programa pasado como argumento pese a salir de la terminal.

- ✓ Por defecto, la salida del comando, que normalmente aparecería directamente en la terminal, será procesada a un fichero llamado **nohup.out** que aparecerá en la ruta donde nos encontremos al ejecutar el comando.
- ✓ Sintaxis:
  nohup programa &

```
# nohup prstat -p 1 &
[1]     1751
# Enviando salida a nohup.out

# ls -l nohup.out
-rw-------   1 root     root         615 jul 29 12:35 nohup.out

# head nohup.out
   PID USERNAME  SIZE   RSS STATE  PRI NICE      TIME  CPU PROCESS/NLWP
     1 root     2556K 1220K sleep   59    0   0:00:01 0,0% init/1
Total: 1 processes, 1 lwps, load averages: 0,34, 0,37, 0,33
   PID USERNAME  SIZE   RSS STATE  PRI NICE      TIME  CPU PROCESS/NLWP
     1 root     2556K 1220K sleep   59    0   0:00:01 0,0% init/1
Total: 1 processes, 1 lwps, load averages: 0,34, 0,37, 0,33
   PID USERNAME  SIZE   RSS STATE  PRI NICE      TIME  CPU PROCESS/NLWP
     1 root     2556K 1220K sleep   59    0   0:00:01 0,0% init/1
Total: 1 processes, 1 lwps, load averages: 0,36, 0,37, 0,33
   PID USERNAME  SIZE   RSS STATE  PRI NICE      TIME  CPU PROCESS/NLWP
```

➔ Ejecución en segundo plano y con **nohup** del proceso **prstat**. El comando **ls** muestra que se ha creado el fichero **nohup.out**. Con **head** se muestran las primeras líneas del fichero **nohup.out** que contiene ejecuciones del comando **prstat**.

Comando **pargs**

- ✓ Resuelve el antiguo problema de no poder mostrar todos los argumentos que se transfieren a un proceso con el comando **ps**.
- ✓ Sintaxis:
  pargs PID

- ✓ Con la opción **-e** muestra las variables de entorno asociadas al proceso.

```
# pgrep ksh
1386
# pargs -e 1386
1386:   /bin/ksh
envp[0]: AB_CARDCATALOG=/usr/dt/share/answerbooks/es_ES.ISO8859-1/ab_cardcatalog
envp[1]: COLORTERM=gnome-terminal
envp[2]: DISPLAY=:0.0
envp[3]:
DTAPPSEARCHPATH=//.dt/appmanager:/usr/dt/appconfig/appmanager/%L:/usr/dt/appconfig/appman
ager/C
envp[4]:
DTDATABASESEARCHPATH=//.dt/types,/usr/dt/appconfig/types/%L,/usr/dt/appconfig/types/C
envp[5]: DTDEVROOT=
```

➔ Con el comando **pgrep** se obtiene el PID del proceso del intérprete Ksh. Con el segundo comando se obtienen las variables de entorno del proceso (por la extensión de la salida, no se han incluido todos los resultados).

Comando **preap**

- ✓ Es un comando de proceso empleado para eliminar procesos extintos, denominados **zombie**.
- ✓ Un proceso **zombie** es un proceso para el cual su superior, o proceso padre, aún no ha obtenido su estado de salida. Estos procesos suelen ser inofensivos, pero pueden gastar recursos del sistema si son muy numerosos.
- ✓ Sintaxis:
  preap PID

Comando **nice**

- ✓ Permite modificar la prioridad con que se lanza un proceso.
- ✓ La prioridad de los procesos se establece con valores numéricos de 0 a +39. 0 es la máxima prioridad.
- ✓ Por defecto, los procesos corren con prioridad 20.
- ✓ Sintaxis:
  nice [-n] N comando

- ✓ Si al ejecutar **nice** se omite el valor de prioridad se le asignará al proceso prioridad 30.
- ✓ Si el incremento se indica como **-N**, se aumenta la prioridad por defecto en N unidades.
- ✓ Si el incremento se indica como **--N**, se reduce la prioridad por defecto en N unidades.
- ✓ Ejemplos:
  - Se ejecuta programa con la mínima prioridad (+39).

```
# nice -19 firefox
# nice -n 19 firefox
# nice -n +19 firefox
```

  - Se ejecuta el programa con la máxima prioridad (0).

```
# nice --20 firefox
# nice -n -20 firefox
```

Comando **renice**

- ✓ Modifica la prioridad, o valor NI, de los procesos en ejecución.
- ✓ El usuario propietario del proceso sólo puede aumentar el valor de prioridad.
- ✓ El usuario **root** puede modificar la prioridad de cualquier proceso a cualquier valor entre 0 y 39.
- ✓ Opciones:

-g	El argumento se interpreta como ID de grupo (GID).
-i	El argumento se interpreta como clase de procesos.
-p	El argumento se interpreta como ID de proceso (PID).
-u	El argumento se interpreta como ID de usuario (UID).
-n	El argumento se interpreta como de incremento de prioridad actual.

- ✓ Sintaxis:
  renice [-n] incremento/prioridad [-i | -g | -p | -u] ID

- ✓ Si el incremento se indica como **-N** o **+N** se reduce o aumenta el valor de la prioridad por defecto en N unidades.

```
# ps -le |head -n 1 ; ps -le |grep vi
 F S    UID   PID  PPID   C PRI NI     ADDR     SZ    WCHAN TTY          TIME CMD
 0 T      0   998   997   0  80 30        ?   1089          syscon       0:00 vi
# renice -20 -p 998
# ps -le |head -n 1 ; ps -le |grep vi
 F S    UID   PID  PPID   C PRI NI     ADDR     SZ    WCHAN TTY          TIME CMD
 0 T      0   998   997   0  40  0        ?   1089          syscon       0:00 vi
```

- ✓ Si el incremento se indica como **-n -N** o **-n +N** se reduce o aumenta el valor de la prioridad actual en N unidades.

```
# ps -le |head -n 1 ; ps -le |grep vi
 F S    UID   PID  PPID   C PRI NI     ADDR     SZ    WCHAN TTY          TIME CMD
 0 T      0   998   997   0  99 39        ?   1089          syscon       0:00 vi
# renice -n -30 -p 998
# ps -le |head -n 1 ; ps -le |grep vi
 F S    UID   PID  PPID   C PRI NI     ADDR     SZ    WCHAN TTY          TIME CMD
 0 T      0   998   997   0  40  9        ?   1089          syscon       0:00 vi
```

Comando **priocntl**

- ✓ Asigna procesos a clases de prioridad.
- ✓ El valor de prioridad global de los procesos se muestra en la columna PRI de la salida del comando **ps**.
- ✓ Opciones:

-e	Ejecuta el comando.
-c clase	Permite especificar la clase dentro de la que se ejecutará el proceso.
-l	Muestra la lista de clases de prioridad posibles y su rango de valores.

- ✓ Las posibles clases de prioridad de procesos son:

SYS	System Class	
TS	Time Sharing	-60 a 60
SDC	System Duty-Cycle Class	
FX	Fixed priority	0 a 60
IA	Interactive	-60 a 60

Comando **prstat**

- ✓ Reporta información de los procesos en el sistema.
- ✓ Opciones:

-a	Muestra, además de la información de procesos, los datos de uso de recursos por cada usuario.
-p PID	Reporta los datos del proceso con el PID indicado.
-n N	Limita el número de líneas de la lista de procesos a mostrar.

```
# pgrep ksh
1386
# prstat -p 1386
   PID USERNAME  SIZE   RSS STATE  PRI NICE      TIME  CPU PROCESS/NLWP
  1386 root     1928K 1132K sleep   49    0   0:00:01 0,1% ksh/1
```

➔ Se obtiene el PID del proceso "ksh" y, a continuación, la información relativa al mismo. Aunque en la captura no se aprecia, los datos se refrescan periódicamente.

- ✓ Para salir de la ejecución de **prstat** se puede emplear la tecla de escape "**q**" o enviar señal de terminación al proceso con la combinación **Control+c**.

Alias.

Comando **alias**

- ✓ Permite reemplazar una palabra o serie de palabras por otra. Su uso principal es el de abreviar órdenes que se usan con mucha frecuencia.
- ✓ Los alias se mantienen hasta que se termina la sesión en la terminal.
- ✓ Utilizado sin argumentos muestra la lista de los alias existentes.
- ✓ Sintaxis:
  alias nombre="comando"

- ✓ Para poder emplear un alias siempre que se inicie una nueva sesión, será necesario añadir su declaración al fichero de perfil del usuario (.profile, .bashrc…)
- ✓ Los alias por defecto de la shell Ksh son:

```
# alias
autoload='typeset -fu'
command='command '
functions='typeset -f'
history='fc -l'
integer='typeset -i'
local=typeset
nohup='nohup '
r='fc -e -'
stop='kill -STOP'
suspend='kill -STOP $$'
```

El valor "**$$**" que aparece en la última línea de la captura anterior, representa el PID del proceso actual.

```
$ alias h="hostname"
$ h
Solaris10host
```

➔ Se crea el alias "h" para el comando **hostname**. A continuación se invoca dicho comando a través de su alias.

Funciones.

Comando `typeset`

- ✓ Permite establecer en la shell atributos y valores para variables y funciones.
- ✓ Opciones:

-f	Permite crear variables **function**. Sin parámetro, muestra todas las funciones existentes.
-r	Permite crear variables de solo lectura.
-i	Permite crear variables **integer**.
-a	Permite crear variables **array**.
-x	Exporta la variable declarada.

- ✓ Los comandos **set** y **declare**, tienen una función similar a **typeset**.

Ejecución de scripts.

Usualmente los nombres de los scripts de shell contienen la extensión "**sh**" o "**ksh**" (en el caso de que su ejecución esté orientada a la shell de Korn).
En UNIX las extensiones de los ficheros son meramente informativas: se consideran parte del nombre del elemento. Por este motivo, cualquier extensión, o ausencia de la misma, no afectará a la ejecución de scripts.

En shell script es importante la línea en el guión que establece la shell del sistema que actuará como intérprete de su contenido.
Se indica de la forma:
`#!/ruta/shell`

Por ejemplo:
`#!/bin/ksh`
`#!/usr/bin/bash`

De no informarse este dato, la shell que interpretará el código del script en la ejecución será la de la sesión.

El script a ejecutar debe contener permiso de ejecución, al menos, al nivel del usuario que lo desea ejecutar: permiso a nivel de usuario, si el propietario del script y el usuario que lo ejecuta es el mismo; permiso a nivel de grupo, si el usuario ejecutor no es propietario del fichero pero pertenece al mismo grupo que el propietario; o a nivel de otros, si el usuario ejecutor no cumple ninguno de los anteriores casos.

Para lanzar la ejecución de un script, se debe indicar su nombre desde el directorio en que se encuentra o desde otro si se especifica la ruta absoluta al mismo.
Si la ruta al script es un directorio contenido en la variable de entorno PATH no será necesario que se cumpla ninguna de las formas anteriores, bastará con indicar su nombre en el prompt desde cualquier ubicación.

```
$ script.sh
```

➔ Ejecución de script válida tanto desde la ubicación del mismo como desde otra ubicación si esta está contenida en la variable PATH del entorno.

```
$ /export/home/gustavo/script.sh
```

➔ Ejecución de script indicando la ruta absoluta al mismo.

```
$ ./script.sh
```

➔ Ejecución de script exclusivamente desde la ubicación del fichero.

En algunos casos los scripts contemplan en su código fuente la recepción de parámetros o argumentos. Estos serán determinados valores indicados en el momento de ejecución y a continuación del nombre del script y separados por espacios.

5. Archivo de ficheros y transferencias remotas.

5.1. Utilizar comandos para crear ficheros y visualizar su contenido.

Comando **cat**

- ✓ Si recibe el nombre de un fichero como parámetro, muestra su contenido por pantalla.
- ✓ En el caso de recibirse varios nombres de ficheros, los concatena y los muestra en la salida estándar.
- ✓ Opciones:

-n	Numera todas las líneas.
-b	Numera las líneas que no están en blanco.

Comando **head**

- ✓ Muestra el encabezado de un fichero. Por defecto las 10 primeras líneas.
- ✓ Opciones:

-n N -N	Muestra las N primeras líneas del fichero.

```
# head -2 /etc/vfstab
#device         device          mount           FS      fsck    mount   mount
#to mount       to fsck         point           type    pass    at boot options
# head -n 2 /etc/vfstab
#device         device          mount           FS      fsck    mount   mount
#to mount       to fsck         point           type    pass    at boot op
```

➔ Salida por pantalla de las dos primeras líneas del fichero indicado.

Comando **tail**

- ✓ Muestra la última parte de un fichero. Por defecto las 10 últimas líneas.
- ✓ Opciones:

-f	Continúa tratando de leer: útil para ficheros en crecimiento.
-N	Muestra las últimas N líneas del fichero.
+N	Muestra el contenido del fichero a partir de la línea número N.

```
$ tail -2 /etc/vfstab
objfs   -       /system/object  objfs   -       no      -
swap    -       /tmp    tmpfs   -       yes     -
$ tail +2 /etc/vfstab
#to mount       to fsck         point           type    pass    at boot options
#
fd      -       /dev/fd fd      -       no      -
/proc   -       /proc   proc    -       no      -
/dev/dsk/c0t0d0s1       -       -       swap    -       no      -
/dev/dsk/c0t0d0s0       /dev/rdsk/c0t0d0s0      /       ufs     1       no      -
/dev/dsk/c0t0d0s7       /dev/rdsk/c0t0d0s7      /export/home    ufs     2
yes     -
/devices        -       /devices        devfs   -       no      -
sharefs -       /etc/dfs/sharetab       sharefs -       no      -
ctfs    -       /system/contract        ctfs    -       no      -
objfs   -       /system/object  objfs   -       no      -
swap    -       /tmp    tmpfs   -       yes     -
```

➔ El primer comando devuelve las dos últimas líneas del fichero. El segundo, en cambio, devuelve todo el contenido del fichero a partir de la segunda línea.

Comando **more**

- ✓ Paginador del contenido de ficheros.
- ✓ Permite realizar búsquedas dentro del contenido de los mismos.
- ✓ La tecla **Enter** avanza en una línea en la paginación del fichero, la barra espaciadora lo hace en páginas.
- ✓ Opciones:

-N	La paginación se realiza cada N líneas.
/patrón	Busca el patrón dentro del contenido del fichero.

Comando **less**

- ✓ Paginador del contenido de ficheros. Es una evolución de **more**.
- ✓ Opciones:

-N	Muestra el número de posición de cada línea.
:N	Se posiciona en la línea número N del fichero.
=	Información del fichero: nombre, número de líneas y bytes de tamaño.
h	Muestra ayuda.
q	Salida.
-N	La paginación se realiza cada N líneas.
/patrón	Busca el patrón dentro del contenido del fichero.
n	Ir a siguiente coincidencia (después de una búsqueda).
N	Ir a coincidencia anterior (después de una búsqueda).

5.2. Usar comandos para comprimir y descomprimir ficheros, y realizar transferencias de los mismos.

Comando **gzip**

✓ Para comprimir:

```
$ ls -lrt fichero*
-rw-r--r--   1 gustavo  gustavo    17907 jul 29 13:18 fichero1
-rw-r--r--   1 gustavo  gustavo       60 jul 29 13:18 fichero2
$ gzip fichero1 fichero2
$ ls -lrt fichero*
-rw-r--r--   1 gustavo  gustavo     3115 jul 29 13:18 fichero1.gz
-rw-r--r--   1 gustavo  gustavo       83 jul 29 13:18 fichero2.gz
```

Se comprimen los ficheros con los nombres de los originales añadiendo la extensión "**gz**" (fichero1.gz, fichero2.gz…)
No conserva los ficheros originales.

✓ Para descomprimir:

```
$ ls -lrt fichero*
-rw-r--r--   1 gustavo  gustavo     3115 jul 29 13:18 fichero1.gz
-rw-r--r--   1 gustavo  gustavo       83 jul 29 13:18 fichero2.gz
$ gunzip fichero1.gz fichero2.gz
$ ls -lrt fichero*
-rw-r--r--   1 gustavo  gustavo    17907 jul 29 13:18 fichero1
-rw-r--r--   1 gustavo  gustavo       60 jul 29 13:18 fichero2
```

✓ Para ver el contenido de los ficheros, pero sin descomprimir:

```
$ gunzip -c fichero2
SunOS Solaris10host 5.10 Generic_147148-26 i86pc i386 i86pc
```

Comando **zip**

✓ Para comprimir:
zip fichero[.zip] fichero1 fichero2…

```
$ ls -lrt
total 38
-rw-r--r--   1 gustavo  gustavo    17907 jul 29 13:18 fichero1
-rw-r--r--   1 gustavo  gustavo       60 jul 29 13:18 fichero2
$ zip comprimido.zip fichero1 fichero2
  adding: fichero1 (deflated 83%)
  adding: fichero2 (deflated 7%)
$ ls -lrt fichero* comprimido*
-rw-r--r--   1 gustavo  gustavo    17907 jul 29 13:18 fichero1
-rw-r--r--   1 gustavo  gustavo       60 jul 29 13:18 fichero2
-rw-r--r--   1 gustavo  gustavo     3454 jul 29 13:26 comprimido.zip
```

➔ Se genera el fichero "fichero.zip" conteniendo, comprimidos, los ficheros con los nombres de los originales. Conserva los ficheros originales.

✓ Para descomprimir:
unzip fichero.zip

✓ Para listar los ficheros y directorios contenidos:
unzip -v fichero.zip

```
$ unzip -v comprimido.zip
Archive:  comprimido.zip
 Length   Method    Size  Cmpr    Date    Time   CRC-32   Name
--------  ------  ------- ---- ---------- ----- --------  ----
   17907  Defl:N     3088  83% 07-29-2016 13:18 d06a19e4  fichero1
      60  Defl:N       56   7% 07-29-2016 13:18 c699182a  fichero2
--------          -------  ---                            -------
   17967             3144  83%                            2 files
```

✓ Para ver el contenido de los ficheros, pero sin descomprimir:
unzip -c fichero.zip

Comando **bzip2**

✓ Para comprimir:
bzip2 fichero1 fichero2…

```
$ bzip2 fichero1 fichero2
$ ls -lrt fichero*
-rw-r--r--   1 gustavo  gustavo     2453 ago 14 23:46 fichero1.bz2
-rw-r--r--   1 gustavo  gustavo      173 ago 14 23:46 fichero2.bz2
```

➔ Se comprimen los ficheros con los nombres de los originales añadiendo la extensión "**bz2**"
Por defecto no conserva los ficheros originales.
Para conservar el fichero original se emplea la opción **-k**:
bzip2 -k fichero

```
$ bunzip2 fichero*
$ ls -lrt fichero*
-rw-r--r--   1 gustavo  gustavo    20191 ago 14 23:46 fichero1
-rw-r--r--   1 gustavo  gustavo      196 ago 14 23:46 fichero2
```

✓ Para descomprimir:
bzip2 -d fichero.bz2
bunzip2 fichero.bz2

Comandos **compress** y **uncompress**

✓ Para comprimir:
compress fichero1

```
$ compress fichero1
$ ls -lrt fichero1*
-rw-r--r--   1 gustavo  gustavo    17907 jul 29 13:38 fichero1.Z
```

➔ Se comprime el fichero con el nombre del original añadiendo la extensión ".Z". Por defecto no conserva los ficheros originales.

✓ Para descomprimir:
uncompress fichero.Z

```
$ uncompress fichero1.Z
$ ls -lrt fichero1*
-rw-r--r--   1 gustavo  gustavo    17907 jul 29 13:50 fichero1
```

Comando **tar**

✓ Para empaquetar:
tar cvf fichero.tar fichero1 fichero2...

✓ Para desempaquetar:
tar xvf fichero.tar

✓ Para listar los ficheros y directorios contenidos:
tar tvf fichero.tar

✓ Opciones:

c	Crea un paquete **tar** con los elementos que se indiquen como argumentos.
x	Extrae los elementos contenidos en el fichero **tar**.
t	Lista el contenido del fichero **tar**.
v	Muestra como salida de la ejecución todos los ficheros que se leen o extraen.
f	Permite especificar el fichero **tar** como argumento. El nombre del **tar** irá como primer argumento del comando.

```
$ tar cvf paquete.tar fichero*
a fichero1 18K
a fichero2 18K

$ tar tvf paquete.tar
-rw-r--r-- 100/101  17907 jul 29 13:39 2016 fichero1
-rw-r--r-- 100/101  17907 jul 29 13:39 2016 fichero2

$ tar xvf paquete.tar
x fichero1, 17907 bytes, 35 bloques de cinta
x fichero2, 17907 bytes, 35 bloques de cinta
```

Comando `jar`

- ✓ Para empaquetar:
  `jar cvf fichero.jar fichero1 fichero2...`
- ✓ Para desempaquetar:
  `jar xvf fichero.jar`
- ✓ Para listar los ficheros y directorios contenidos:
  `jar tvf fichero.jar`

5.3. Conexiones remotas para establecer sesión y transferir ficheros y directorios.

Comandos `rlogin` y `rsh`

- ✓ El comando **rlogin** permite establecer una sesión en un equipo remoto siempre que en el equipo remoto se encuentre en ejecución el demonio **rlogind**.
- ✓ Emplea el puerto 513.
- ✓ Sintaxis:
  `rlogin nombre_host_remoto|IP_host_remoto`
- ✓ El comando **rsh** ejecuta comandos en equipos remotos.
- ✓ Su sintaxis es:
  `rsh equipo_remoto [-l usuario] "comando"`
- ✓ El fichero **.rhosts** del equipo remoto almacena los datos de acceso del usuario cliente, como la contraseña del mismo.

Secure Shell (SSH).

Es un protocolo que permite el acceso remoto a un sistema a través de la red y en modo texto.
Encripta todas las transferencias de contraseñas y de datos en general.
Además permite transferir ficheros de forma segura y que los protocolos de red no encriptados transporten sus datos sobre una conexión SSH.
Los ficheros de configuración más destacables para los parámetros de cliente y servidor SSH son:

- El fichero de configuración para el cliente SSH **/etc/ssh/ssh_config**.
- El fichero de configuración del servidor OpenSSH **/etc/ssh/sshd_config**.

El fichero **/etc/ssh/sshd_config** permite establecer valores para parámetros como los siguientes:

- **Protocol**. Niveles de protocolo que entiende SSH. Pueden ser 1, 2 o ambos.
- **PermitRootLogin**. Por defecto viene marcado como yes y permite establecer accesos remotos como **root**.
- **X11Forwarding**. Establece si se permite o no a los usuarios remotos ejecutar programas **X** (servidor gráfico UNIX) a través de SSH.
- **Port**. Número de puerto a través del que se establece la conexión SSH. Por defecto, viene establecido el puerto 22.

Sistema de autenticación basada en claves SSH.

- SSH emplea dos claves: una pública y otra privada.
- Estas dos claves están vinculadas matemáticamente de forma que los datos encriptados con una clave pública concreta sólo se pueden desencriptar con la clave privada coincidente.

- Cuando se establece una conexión cada lado envía al otro su clave pública. Tras ello, cada lado encripta la información con la clave pública del otro, lo que asegura que los datos sólo los podrá desencriptar el destinatario deseado.
- Los clientes SSH suelen conservar las claves públicas de los servidores a los que se conectan.
- En total se emplean de cuatro a seis claves. Estas claves se denominan **ssh_host_rsa_key** (pública) y **ssh_host_dsa_key** (privada) y se almacenan en **/etc/ssh**.
- Las claves pueden regenerarse empleando el comando **ssh-keygen**.
- El programa SSH registra las claves de host para cada usuario individual en el fichero **~/.ssh/known_hosts**.
- El fichero **~/.ssh/authorized_keys** permite almacenar la información de las claves públicas de los hosts remotos para que no se requiera autenticación al conectar.
- La utilidad **ssh-agent** permite que sólo se solicite la contraseña del usuario una vez por cada conexión local.

File Transfer Protocol (FTP).

El comando **ftp** abre la interfaz de usuario del protocolo de transferencia de ficheros de Internet. Esta interfaz de usuario permite conectar a un sistema remoto y realizar una variedad de operaciones sobre su sistema de ficheros.
El principal beneficio de FTP sobre rlogin y rcp es que FTP no requiere que el sistema remoto ejecute UNIX.
Comandos FTP más relevantes:

ftp host	Establece una conexión FTP al host remoto.
bye	Sale del intérprete de comandos FTP.
help	Muestra el listado de comandos disponibles.
ls	Lista el contenido del directorio activo remoto.
pwd	Muestra el path del directorio activo del host remoto.
cd	Cambia el directorio activo remoto.
lcd	Cambia el directorio activo local.
get	Copia un fichero del host remoto al local.
mget	Copia varios ficheros del host remoto al local.
put	Copia un fichero del host local al remoto.
mput	Copia varios ficheros del host local al remoto.
delete	Elimina un fichero remoto.
mdelete	Elimina varios ficheros remotos.

El servicio cliente FTP es:

```
# svcs |grep ftp
online         14:02:09 svc:/network/ftp:default
```

Los ficheros de configuración del servidor FTP son:

```
# ls -l /etc/ftpd
       -rw-r--r--   1 root     sys         1518 ene 11  2013 ftpaccess
       -rw-r--r--   1 root     sys          946 ene 11  2013 ftpconversions
```

```
-rw-r--r--   1 root     sys          104 ene 11  2013 ftpgroups
-rw-r--r--   1 root     sys          108 ene 11  2013 ftphosts
-rw-r--r--   1 root     sys          114 ene 11  2013 ftpservers
-rw-r--r--   1 root     sys          206 ene 11  2013 ftpusers
```

La función de configuración del servidor FTP de cada uno de ellos es:

ftpaccess	Es el fichero principal de configuración y permite establecer la mayoría de los parámetros.
ftpconversions	Establece conversiones de fichero para compresión y empaquetado.
ftpgroups	Contiene los nombres de los grupos de los usuarios con acceso FTP permitido o denegado.
ftphosts	Establece, por IP o nombre, aquellos host que tienen permitido (allow) o no (deny) el acceso FTP.
ftpservers	Permite configurar hosts virtuales.
ftpusers	Contiene los nombres de los usuarios a los que no se les permite el acceso FTP al servidor.

```
$ ftp 192.168.1.128
Connected to 192.168.1.128.
220 solaris FTP server ready.
Name (192.168.1.128:gustavo):
331 Password required for gustavo.
Password:
230 User gustavo logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> cd /export/home/gustavo
250 CWD command successful.
ftp> get fichero1 fichero2
200 PORT command successful.
150 Opening BINARY mode data connection for fichero1 (20191 bytes).
226 Transfer complete.
local: fichero2 remote: fichero1
20191 bytes received in 0,055 seconds (355,75 Kbytes/s)
ftp> bye
221-You have transferred 20191 bytes in 1 files.
221-Total traffic for this session was 20641 bytes in 1 transfers.
221-Thank you for using the FTP service on solaris.
221 Goodbye.
```

➔ Conexión FTP a host remoto con IP 192.168.1.128 y usuario actual (usuario "gustavo" en este caso). Posicionamiento en el directorio home del usuario /export/home/gustavo y descarga de copia de los ficheros "fichero1" y "fichero2" al host original. Finalmente, se cierra la conexión.

6. Apéndice.

6.1. Sistemas de ficheros.

Basados en disco.

UFS	Es el sistema de ficheros por defecto en Solaris.
ZFS	Permite administrar los volúmenes de disco.
HSFS	Es de solo lectura. Se emplea para CD-ROM.
PCFS	Sistema de archivos de PC, que permite el acceso de lectura y escritura a los datos y programas en discos con formato DOS que se escriben para los ordenadores personales basados en DOS. Se emplea en diskette.
UDFS	Orientado a DVD.

Basados en red.

NFS	Es un servicio de sistema de archivos distribuido que puede ser usado para compartir recursos (ficheros o directorios) de un sistema, típicamente un servidor, con otros sistemas de la red. Puede compartir un recurso mediante el comando **share** o **shareall** o añadiendo una entrada en el fichero **/etc/dfs/dfstab** y reiniciando posteriormente el sistema.

Virtuales.

TMPFS	Sistema de ficheros temporal montado en **/tmp**.
PROCFS	Montado en **/proc**.
DEVFS	Montado en **/devices**.
SWAPFS	Sistema de ficheros para la memoria de intercambio.

6.2. Ficheros de configuración de sistemas de archivos.

/etc/mnttab	Tabla de sistemas de ficheros, de solo lectura, y mantenido por el sistema. Los campos que contiene son: “special”, “mount_point”, “fstype”, “options” y “time”.
/etc/vfstab	Tabla de sistemas de ficheros con sus valores por defecto. Puede ser modificado por el administrador del sistema. Los campos que contiene son: “device to mount”, “device to fsck”, “mount point”, “FS type”, “fsck pass”, “mount at boot” y “mount options”.
/etc/dfs/dfstab	Contiene los sistemas de ficheros compartidos en red por NFS.

6.3. Árbol de directorios.

/bin	Es un enlace simbólico al directorio **/usr/bin** con los ficheros binarios.
/boot	Ficheros de inicio del sistema.
/cdrom	Punto de montaje para el dispositivo CD-ROM.

/dev	Contiene los ficheros de dispositivo. En realidad, los ficheros son enlaces simbólicos a los ficheros del directorio **/devices**.
/devices	El sistema de archivos **devfs** gestiona el directorio **/devices**, que es el espacio de nombres de todos los dispositivos del sistema. Este directorio representa los dispositivos físicos que consiste en direcciones reales de bus y de dispositivos.
/Documents	
/etc	Contiene los ficheros de configuración.
/export/home	Directorios home de los usuarios para almacenar sus ficheros personales.
/home	
/kernel	Contiene componentes del núcleo comunes a todas las plataformas dentro de un sistema de instrucción particular, que son necesarios para arrancar el sistema.
/lib	Contiene los ficheros de las librerías del sistema.
/mnt	Directorio empleado para contener los puntos de montaje de los sistemas de ficheros.
/opt	Contiene software comercial que no viene con el sistema.
/platform	Contiene los ficheros de definición de plataformas (arquitecturas).
/proc	Es un sistema de ficheros virtual que contiene información de los procesos ejecutándose en el sistema. Los ficheros de los procesos se estructuran en directorios de la forma **/proc/**<PID>.
/sbin	Contiene ficheros binarios habitualmente ejecutados por el administrador del sistema.
/system	**/system/contract** es un sistema de ficheros virtual que mantiene información de contrato. El subdirectorio /system/object es, igualmente, un sistema de archivos virtual para debuggers que permite acceder a información del kernel sin necesidad de acceder a él directamente.
/tmp	Directorio de ficheros temporales del sistema o de los usuarios. Los ficheros dentro de este directorio son eliminados cuando el sistema es iniciado.
/usr	Mantiene los ficheros y directorios del sistema que deben ser compartidos con otros usuarios.
/var	Contiene ficheros que se espera que crezcan en tamaño a corto plazo, como los logs del sistema o los ficheros de backup.
/vol	Puntos de montaje de los volúmenes del disco.

6.4. Enlaces.

Un enlace es un medio de proporcionar varias identidades a un fichero. Los enlaces se emplean para facilitar la accesibilidad a los ficheros, dotar de varios nombres a un comando, permitir a los programas buscar los mismos ficheros en distintas ubicaciones, etc.

Enlace duro.

- El enlace duro y el fichero enlazado comparten el mismo número de inodo, es decir, son referencias a un mismo fichero.
- Los cambios que realicemos sobre cualquiera de los enlaces, será reflejado en el resto, debido a que es el mismo fichero.
- Si borramos algún enlace esto no afecta al fichero mientras existan enlaces que apunten a su inodo.
- No pueden ser utilizados para directorios.
- Sólo pueden ser creados en el mismo sistema de ficheros donde se encuentre el fichero original.

Enlace simbólico.

- El enlace simbólico no enlaza con el inodo del fichero origen, sino que es un acceso directo al fichero. En este caso el inodo del fichero origen y del enlace simbólico son diferentes.
- El enlace y el fichero tienen distintos tamaños, ya que el enlace simbólico únicamente contiene la ruta para poder acceder al fichero origen.
- Los cambios que realicemos sobre cualquiera de los enlaces, será reflejado en el resto.
- Si borramos el fichero origen, el enlace simbólico permanece pero sin ninguna utilidad.
- Los enlaces simbólicos sí que pueden ser creados para directorios y pueden extenderse a otros sistemas de ficheros.

Comando `ln`

✓ Es el comando encargado de la construcción de enlaces duros y simbólicos en el sistema.
✓ Sintaxis:
`ln [-opciones] fichero nombre_enlace`

✓ Opciones:

-s	Crea enlace simbólico en lugar de duro.
-f	Forzoso: elimina ficheros en destino existentes con el mismo nombre.
-i	Interactivo: pide confirmación antes de sobrescribir ficheros existentes con el mismo nombre.

```
$ touch fichero
$ ln fichero "enlace duro"
-bash-3.2$ ln -s fichero "enlace simbolico"
-bash-3.2$ ls -lrt
total 2
-rw-r--r--   2 gustavo  gustavo        0 jul 29 14:02 fichero
-rw-r--r--   2 gustavo  gustavo        0 jul 29 14:02 enlace duro
lrwxrwxrwx   1 gustavo  gustavo        7 jul 29 14:02 enlace simbolico -> fichero
```

➔ Se crea el fichero "fichero". Se crea un enlace duro a "fichero" llamado "enlace duro". Se crea un enlace simbólico "enlace simbolico" a "fichero".

6.5. Apagado y reinicio.

Comando `reboot`

✓ Se trata de un ejecutable que sincroniza los discos y pasa las instrucciones de inicio al sistema. A su vez, esta llamada al sistema detiene el procesador.
✓ Es preferible emplear el comando **init**.

Comando `init`

✓ Termina todos los procesos activos y sincroniza los discos antes de cambiar los niveles de ejecución.
✓ Este comando proporciona un apagado del sistema más rápido. Es preferible para apagar sistemas autónomos cuando otros usuarios no se vean afectados.

Comando **shutdown**

- ✓ Apagado/reinicio del sistema.
- ✓ Es un ejecutable que llama al programa **init** para apagar el sistema. El sistema pasa al nivel de ejecución **S** por defecto.
- ✓ Se emplea para el apagado del sistema cuando se encuentra en el nivel de ejecución **3**.
- ✓ Opciones:

-i N	Permite cambiar el nivel de ejecución. Puede tomar los valores 0, 1, 2, 5 o 6.
-g N	Tiempo en segundos del periodo de gracia previo al comienzo del apagado.
-y	No solicita confirmación.

```
# shutdown -y -g 0

Shutdown started.    viernes 29 de julio de 2016 14:10:03 CEST

Changing to init state s - please wait
Broadcast Message from root (pts/3) on Solaris10host vie jul 29 14:10:03...
THE SYSTEM Solaris10host IS BEING SHUT DOWN NOW ! ! !
Log off now or risk your files being damaged
```

➔ Paso del sistema al nivel de ejecución "**S**" sin periodo de gracia previo y sin solicitar confirmación.

Comandos **poweroff** y **halt**

- ✓ Apagado del sistema. Sincroniza los discos y apaga el procesador.
- ✓ No es recomendable, ya que no se cierran todos los procesos ni se desmontan los sistemas de ficheros. La detención de los servicios, sin hacer un apagado limpio, sólo debe hacerse en caso de emergencia o si la mayoría de los servicios que ya están parados.
- ✓ Opciones:

-l	No se envía mensaje al demonio **syslogd**, por lo que no se registra el evento en el log del sistema.
-q	No hay periodo de gracia antes del apagado.

6.6. Niveles de ejecución.

Nivel	Estado	Descripción
0	Power-down state	Para apagar el sistema operativo para que sea seguro apagar la alimentación del sistema.
S	Single-user state	Para ejecutar como un usuario único con algunos sistemas de archivos montados y accesibles.
1	Administrative state	Para acceder a todos los sistemas de archivos disponibles. Los inicios de sesión de los usuarios están deshabilitados.
2	Multiuser state	Para las operaciones normales. Varios usuarios pueden acceder al sistema y a todos los sistemas de archivos. Todos los

		demonios están corriendo a excepción de los demonios del servidor NFS y SMB.
3	Multiuser level with NFS resources shared	Para las operaciones normales con NFS y SMB. Este es el nivel de ejecución predeterminado.
4	Alternative multiuser state	No configurado por defecto, pero está disponible para uso del cliente.
5	Power-down state	Para apagar el sistema operativo para que sea seguro apagar la alimentación del sistema. Si es posible, se apaga automáticamente la energía en los sistemas que admitan esta función.
6	Reboot state	Para apagar el sistema al nivel de ejecución 0, y luego reiniciar el nivel multiusuario con NFS y recursos compartidos SMB (o cualquier nivel con el valor predeterminado en el archivo **inittab**).

6.7. Administración de paquetes.

Comando **pkgadd**

- ✓ Transfiere el contenido de los paquetes de software al sistema.
- ✓ La opción **-d** permite que se pueda especificar la ruta absoluta a los paquetes de software. El directorio de paquetes por defecto es **/var/spool/pkg**.

```
# pkgadd -d /export/home/gustavo/pkgutil.pkg
```

Comando **pkginfo**

- ✓ Si se ejecuta sin argumentos, lista los paquetes instalados en el sistema, junto a una breve descripción del objetivo de cada uno.
- ✓ Si se le pasa como parámetro el nombre de un paquete, proporciona información sólo del mismo.

```
# pkginfo CSWpkgutil
```

Comando **pkgrm**

- ✓ Elimina del sistema el paquete cuyo nombre se le pasa como argumento.

```
# pkgrm CSWpkgutil
```

Comandos `patchadd y patchrm`

- ✓ El comando **patchadd** permite añadir parches a los paquetes instalados en el sistema.
- ✓ El comando **patchrm**, por el contrario, es la utilidad para eliminar los parches aplicados que se detallen junto a él.

Comando `pkgutil`

- ✓ Consiste en un metainstalador que facilita la administración de paquetes.
- ✓ En vez de hacer que sea el administrador del sistema el encargado de aplicar los paquetes a partir los ficheros **pkg** descargados al sistema, y resolver manualmente las dependencias de los mismos; los paquetes se aplican desde repositorios en línea mapeados y sus dependencias son resueltas de manera automática por la utilidad.
- ✓ Opciones:

-i	Instala el paquete indicado.
-u	Actualiza la versión del paquete instalado.
-r	Elimina el paquete del sistema.
-d	Descarga el paquete pero sin realizar su instalación.
-U	Actualiza los datos del catálogo de paquetes en el sistema.
-a	Lista los paquetes disponibles para ser instalados.
-l	Lista los paquetes instalados en el sistema.

6.8. Administración básica de red.

Los ficheros del sistema que contienen la configuración de red del sistema son:

- Para nombre de host:
  - **/etc/hostname.<interfaz>**
  - **/etc/hosts**
  - **/etc/nodename**
- Para IP:
  - **/etc/hosts**
  - **/etc/inet/ipnodes**

Comando `ping`

- ✓ Se usa para determinar el estado de un host remoto. Al ejecutar el comando ping, el protocolo ICMP envía al host un determinado datagrama para solicitar una respuesta.
- ✓ Es posible especificar el tiempo de espera para la respuesta. Por defecto es de 20 segundos.
- ✓ Con la opción -s se envían constantemente paquetes al host destino hasta que se envía un carácter de finalización (como Control+c) o salta el time out de espera de respuesta.
- ✓ Sintaxis:
  `ping <host|IP>`

Comando **traceroute**

- ✓ Muestra la ruta del datagrama y el tiempo de ida y vuelta para cada puerta de entrada a lo largo de la ruta de acceso al host de destino.
- ✓ Sintaxis:
  traceroute <host|IP>

Comando **ifconfig**

- ✓ Muestra y permite configurar los parámetros de las interfaces de red
- ✓ Opciones:

ifconfig <interfaz>	Muestra los datos de red de la interfaz indicada.
ifconfig **-a**	Muestra los datos de red de todas las interfaces.
ifconfig <interfaz> **down**	Marca la interfaz como no operativa.
ifconfig <interfaz> **up**	Habilita la interfaz.
ifconfig <interfaz> **dhcp**	Habilita el protocolo DHCP para la interfaz.
ifconfig <interfaz> **auto-dhcp**	
ifconfig <interfaz> <IP>	Estable como IP fija la indicada como parámetro.

Comando **netstat**

- ✓ Muestra el estado de la red y estadísticas de la misma.
- ✓ Opciones:

-s	Estadísticas de la red por protocolos.
-i	Muestra el estado de las interfaces de red y el número de paquetes transmitidos y recibidos en cada una de ellas.
-r	Muestra la tabla de enrutamiento.
-a	Muestra el estado de los sockets y la tabla de enrutamiento.
-P protocolo	Muestra los datos relativos al protocolo indicado (**TCP**, **UDP** o **SCTP**).