Unidad 10

Introducción a Sistemas II

Implantación de sistemas operativos

Índice

10.1. El sistema de archivos en Linux

10.1.1. Estructura de directorios en GNU/Linux
10.1.2. Gestión de archivos por línea de comandos en Linux
10.1.3. Tipos de ficheros
10.1.4. Eliminación de ficheros
10.1.5. Creación y eliminación de directorios
10.1.6. Copia de archivos
10.1.7. Renombrado o movimiento de archivos
10.1.8. Impresión de archivos
10.1.9. Cuenteo de un fichero
10.1.10. Ordenación de un fichero
10.1.11. Entrada y salida estándar. Redirecciones
10.1.12. Procesamiento de textos

10.2. Los usuarios en Linux

10.2.1. Configuración de usuarios y grupos
10.2.2. Comandos de gestión de usuarios
10.2.3. Usuarios y grupos predeterminados

10.3. Identificación y administración de procesos en Linux

Introducción

Los sistemas operativos UNIX nacieron en la década de los 70 cuando empezaron a desarrollarse por universitarios estadounidenses en lenguaje C. Ya desde el principio lo que se intento es que el sistema tuviese una gran versatilidad y facilidad a la hora de su uso por parte del usuario.

Para UNIX, tenemos diferentes versiones dependiendo de las plataformas con las que trabajemos, de libre distribución, como puede ser Linux o comerciales como Solaris. Aunque muy parecidas y con un mismo núcleo, no todos tienen las mismas características.

Linux en concreto se trata de un sistema UNIX de distribución libre que fue acogido con gran entusiasmo y que debido a esto y a que muchos desarrolladores se han interesado cada vez más en ellos, ha sido uno de los sistemas con mayor crecimiento exponencial.

En Linux, en la actualidad, vamos a ver varias distribuciones distintas, como Debian, Mint, Ubuntu O RedHat. Nosotros vamos a trabajar como en todo el curso, con la primera, Debian.

Al finalizar esta unidad

- Sabremos cual es la estructura de un sistema de ficheros en Linux.
- Seremos capaces de gestionar los ficheros en Linux.
- Podremos crear, modificar y gestionar usuarios y grupos en Linux
- Conoceremos como funcionan los procesos en Linux y su correcta gestión.

10.1. El sistema de archivos en Linux

10.1.1. Estructura de directorios en GNU/Linux

En Linux se usan los sistemas de archivos extendidos versión 4, es decir, ext4. Este sistema de archivos tiene la particularidad de que, gracias a él, **todo en Linux se puede expresar como un archivo**. Entonces, tenemos que hacer una diferenciación entre las distintas formas que puede tener un archivo.

Los archivos llevan un **identificador** que ayuda a que Linux diferencie los tipos de archivos:

- **Archivo sencillo (-):** estos pueden contener texto, datos, o ser un ejecutable.
- **Directorio (d):** es un archivo donde se almacenan o listan otros archivos distintos.
- **Vínculo (l):** es un enlace simbólico.
- **Dispositivo de bloque (b) o de carácter (c):** son usados para los dispositivos de entrada y salida que interactúan directamente con el sistema operativo y el hardware.
- **Socket (s):** este archivo se encarga de facilitar la comunicación entre distintos procesos locales.
- **Tubería (p):** este archivo nos ayuda a comunicar de manera unidireccional de los procesos.

Estos identificadores se muestran a la izquierda del fichero cuando lancemos el comando ls -l.

```
root@debian:/home/miguel# ls -l
total 32
drwxr-xr-x 2 miguel miguel 4096 ene 25 08:50 Descargas
drwxr-xr-x 2 miguel miguel 4096 ene 25 08:50 Documentos
drwxr-xr-x 2 miguel miguel 4096 ene 27 09:15 Escritorio
drwxr-xr-x 2 miguel miguel 4096 ene 25 08:50 Imágenes
drwxr-xr-x 2 miguel miguel 4096 ene 25 08:50 Música
drwxr-xr-x 2 miguel miguel 4096 ene 25 08:50 Plantillas
-rw-r--r-- 2 root   root      0 ene 26 14:13 prueba
drwxr-xr-x 2 miguel miguel 4096 ene 25 08:50 Público
drwxr-xr-x 2 miguel miguel 4096 ene 25 08:50 Vídeos
```

Imagen 1. Salida comando ls -l.

Diferente a lo que pasaba con Windows, donde los directorios colgaban de una unidad lógica, en Linux cuelgan de un directorio raíz. A partir de este directorio raíza o root se extiende el árbol de directorios donde destacan los siguientes:

- **/bin.** Este directorio contiene todos los archivos ejecutables y los comandos básicos del sistema.
- **/boot.** Aquí se contienen los archivos que son necesarios para arrancar el sistema operativo además de las imágenes del núcleo del sistema operativo.
- **/dev.** Se incluyen aquí los dispositivos del sistema.
- **/etc.** En este directorio se almacenan todos los archivos de configuración del sistema operativo además de los de los programas instalados.
- **/home.** Se incluyen los directorios personales de los usuarios, para cada usuario se generará un directorio dentro de este. Se puede designar con el símbolo "~".
- **/lib.** Almacena las librerías que son necesarias a la hora de la ejecución de programas y comandos del sistema.
- **/media.** Se incluyen las unidades de almacenamiento físicas conectada al equipo.
- **/mnt.** Es el directorio que se debe usar para el montaje de nuevos dispositivos.
- **/root.** Es el directorio home del usuario superadministrador o root.
- **/sbin.** Aquí se almacenan los programas y comandos que requieren de privilegios avanzados para su ejecución.
- **/usr.** Este es el directorio más grande que hay ya que contienen toda la información relativa a los programas y aplicaciones no básicos del sistema.
- **/var.** Contiene todo tipo de información variable como puede ser las colas de gestión de recursos del sistema.

10.1.2. Gestión de archivos por línea de comandos en Linux

Aunque muchos sistemas como Windows usan la interfaz gráfica para el manejo de los archivos, en sistemas como Linux, un administrador debe de saber manejar una serie de comandos de la línea de comandos porque los terminales tienen una potencia superior y una amplia versatilidad de funciones a ejecutar.

Los comandos en Linux siguen toda una misma sintaxis básica:

```
comando [opciones] [argumentos]
```

Para cada comando, las opciones y argumentos serán diferente, y además habrá muchas veces que estos sean opcionales y no obligatorios. Estas opciones pueden ser cortas, de hecho, puede que solo sean una letra o puede que se formen por una palabra entera. Aunque hay varios tipos de opciones que pueden usarse en un mismo comando, las cortas a veces hasta se pueden unificar en una única expresión, pero eso dependerá del comando. En Linux, comandos y opciones son sensibles a mayúsculas y minúsculas, por lo que hay que escribir siempre de manera correcta el comando o puede que nos de error o que el resultado no sea el deseado.

Anteriormente ya hablamos sobre el comando ls, que significa listra, pero no hemos hablado de las opciones que este tiene. Estas opciones, que siempre van precedidas de un guion. son:

- **l:** muestra el formato largo.
- **t;** se ordena por fecha de modificación.
- **r:** se invierte el orden de salida.
- **R:** se lista de manera recursiva el contenido de cada uno de los directorios.
- **i:** se muestra el número del i-nodo.
- **a:** muestra todos los archivos, hasta los ocultos, que en Linux comienzan por ".".
- **h:** se muestra el tamaño de los ficheros en la medida que se desee.
- **size:** muestra el tamaño de los ficheros en bloques.
- **S:** se ordenan los archivos por tamaño.

Cuando lanzamos el comando `ls` sin opciones ni argumentos, se muestra el contenido del directorio ordenado alfabéticamente, y si se pide que se muestre de varios directorios, se mostrarán de manera seguida.

Si empleamos la opción `-l` se verá la información de cada uno de los ficheros clasificados por columnas:

1. Los permisos del fichero en cuestión. Esta columna se divide en otras dos subpartes:

   a. El primer carácter indicará si se trata de un directorio, "d", un enlace simbólico, "l", un archivo regular, "-", un dispositivo de tipo carácter, "c" o un dispositivo de tipo bloque, "b".

   b. Los demás caracteres que aparecen en la columna son los permisos que tiene el dichero o directorio para usuarios, grupos y otros, en grupos de tres en tres.

2. La segunda de las columnas establece el número de enlaces duros que tiene el fichero asociado.

3/4. La tercera y cuarta de las columnas nos indicarán el propietario y el grupo de cada fichero de manera respectiva.

5. La quinta columna indicará que tamaño en bytes tiene el archivo.

6. La sexta columna hará referencia a la fecha de modificación última, o en caso de no haberse modificado, a la fecha de creación.

7. La última columna, que es la séptima, nos indica el nombre que tienen el fichero.

SABER MÁS

Los nombres de un fichero en Linux pueden tener una extensión desde 1 a 255 caracteres, y además tienen las siguientes restricciones:

- No se puede usar el carácter "/".
- No deberíamos usar los caracteres que se pueden emplear para operaciones aritméticas.
- Si queremos que haya algún espacio en el nombre del fichero, entonces debemos de entrecomillar el nombre para que esto tome efecto.
- No es necesario indicar la extensión a no ser que sea necesario por la naturaleza del fichero.
- Si queremos que el archivo esté oculto, debe de empezar por un punto, ".".

Dentro de un mismo directorio no puede haber dos directorios con un mismo nombre, porque necesitamos las rutas enteras para poder localizar un archivo.

Todos los directorios tienen, además, dos archivos ocultos mínimo, que son "." Y ".." estos hacen referencia al directorio actual y al directorio padre respectivamente. Siempre se crean estas dos entradas cuando se crea un nuevo directorio.

10.1.3. Tipos de ficheros

En Linux es bastante importante el tipo de fichero con el que se trabaja. De hecho, en Linux, cualquier elemento físico o lógico se va a representar como un archivo para facilitar la gestión y administración de estos. Tenemos cuatro tipos elementales de ficheros:

- **Regulares.** Son los ficheros ordinarios, es decir, se encargan del almacenamiento de información de diversa naturaleza En ellos también se incluyen ficheros ejecutables que contienen código ejecutable y permisos de ejecución asociados.
- **Directorios.** En su bloque de datos se almacenan los números de i-nodo y el nombre de los archivos que contienen.
- **Enlaces.**
- **Dispositivos.** Estos archivos hacen referencia a dispositivos físicos. En Linux la mayoría de esos los encontraremos en el directorio /dev. Existen dos tipos de archivos de dispositivos:
  - Dispositivos por caracteres: estos dispositivos suelen ser los que no disponen de un sistema de archivos como tal como pueden ser teclados. Los datos son transferidos carácter a carácter.
  - Dispositivos por bloques: se almacena la información en bloques de datos físicamente como pueden ser los discos duros.

  Además de estos, tenemos una serie de dispositivos virtuales que se tratan de manera específica como el directorio /dev/null que se suele usar para enviar la información que no es necesaria para posteriormente eliminarla.

Tenemos una opción en el comando `ls. --color`, que viene activada por defecto en Ubuntu y permite que se distingan por el siguiente registro de colores los archivos por su tipo:

- Blanco: archivo regular.
- Verde: archivo ejecutable.
- Azul: directorio.
- Cian: enlace simbólico.
- Rojo: enlace roto.

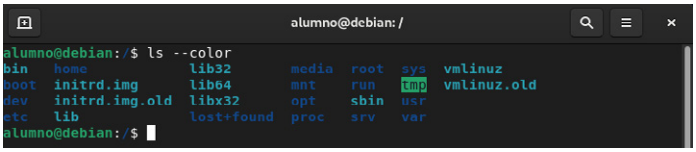

Imagen 2. Comando ls --color.

10.1.4. Eliminación de ficheros

Para eliminar archivos usaremos el comando `rm`, y su sintaxis es la siguiente:

```
rm [-irf] ficheros_eliminar
```

Las opciones que tiene este comando son:

- **i**: solicita que se confirme antes de que se elimine el fichero.
- **r o R**: se trata de una eliminación recursiva sobre los directorios, es decir, borrar la información que se encuentra dentro de los directorios.
- **f**: se fuerza la eliminación, aunque el archivo se encuentre protegido contra escritura.

Por ejemplo, en la siguiente imagen vemos como tenemos un archivo, ejecutamos el comando y después ya no lo tenemos.

```
alumno@debian:~$ ls
Descargas   Escritorio  Música      prueba.txt  Vídeos
Documentos  Imágenes    Plantillas  Público
alumno@debian:~$ rm prueba.txt
alumno@debian:~$ ls
Descargas   Escritorio  Música      Público
Documentos  Imágenes    Plantillas  Vídeos
alumno@debian:~$
```

Imagen 3. Comando rm.

Para seleccionar los archivos a eliminar se pueden también usara las llamadas expresiones regulares, como "*" y "?". El asterisco sustituye a una cadena de caracteres cualquier mientras que el símbolo de interrogación sustituye a un único carácter. Estas se usan si queremos hacer referencia a varios archivos a la vez que tengan una estructura de nombre similar.

10.1.5. Creación y eliminación de directorios

Para crear un directorio usaremos el comando `mkdir`. Este comando tiene la siguiente estructura:

```
mkdir directorios
```

En la siguiente imagen podemos ver como se crea un directorio.

```
alumno@debian:~$ ls
Descargas   Escritorio  Música      Público
Documentos  Imágenes    Plantillas  Vídeos
alumno@debian:~$ mkdir dirprueba
alumno@debian:~$ ls
Descargas  Documentos  Imágenes  Plantillas  Vídeos
dirprueba  Escritorio  Música    Público
alumno@debian:~$
```

Imagen 4. Comando mkdir.

Si por otro lado lo que queremos es borrar un directorio, usaremos la expresión `rm -r` en caso de que el directorio tenga contenido y `rmdir` si los directorios están vacíos.

Si eliminamos el directorio anterior.

```
alumno@debian:~$ ls
Descargas  Documentos  Imágenes  Plantillas  Vídeos
dirprueba  Escritorio  Música    Público
alumno@debian:~$ rmdir dirprueba/
alumno@debian:~$ ls
Descargas   Escritorio  Música      Público
Documentos  Imágenes    Plantillas  Vídeos
alumno@debian:~$
```

Imagen 5. Comando rmdir.

Las opciones son las mismas que las del comando `rm`.

10.1.6. Copia de archivos

Si queremos copiar archivos debemos de usar el comando `cp`. La sintaxis de este comando es:

```
cp [-irR] archivos_que_copiar destino
```

Si en el destino tenemos archivos de igual nombre, por defecto se sobrescribirá su información al copiar los nuevos. Las opciones más usadas por este comando son:

- **i:** pedirá confirmación para la copia si se tiene que sobrescribir un archivo.
- **r o R:** se copia de manera recursiva cuando se usen directorios.

En el siguiente ejemplo vemos como se copia el directorio dirprueba dentro de Descargas:

```
alumno@debian:~$ ls
Descargas  Documentos  Imágenes  Plantillas  Vídeos
dirpueba   Escritorio  Música    Público
alumno@debian:~$ cp dirpueba/ Descargas/
cp: -r not specified; omitting directory 'dirpueba/'
alumno@debian:~$ cp -r dirpueba/ Descargas/
alumno@debian:~$ ls Descargas/
dirpueba
alumno@debian:~$
```

Imagen 6. Comando cp.

Hemos visto en el ejemplo, que si no se usa la opción `-r`, no nos dejará copiar el directorio.

IMPORTANTE

Debemos de distinguir que el copiar archivos o directorios no los elimina del origen, sino que simplemente plasma una copia.

10.1.7. Renombrado o movimiento de archivos

Para mover archivos de ubicación o renombrarlos, se usará el mismo comando, mv. Su funcionamiento es similar al de cp, pero en este caso los archivos de origen se mueven de su ubicación. Su sintaxis es:

```
mv [-iu] archivos_origen destino
```

Las opciones más usadas son:

- **i:** se usa para solicitar confirmación frente a la sobreescritura de los archivos del destino.
- **u:** solo se mueven los archivos o directorios que tengan el mismo nombre entre origen y destino si los del origen son más modernos que los del destino.

Vamos a ver dos imágenes a continuación, la primer a moviendo un directorio con el mismo nombre (el que habíamos copiado previamente), y la segunda cambiándole el nombre.

```
alumno@debian:~$ mv Descargas/dirpueba/
alumno@debian:~$
```

Imagen 7. Comando mv para mover archivos o directorios.

```
alumno@debian:~$ ls
Descargas  Documentos  Imágenes  Plantillas  Vídeos
dirpueba   Escritorio  Música    Público
alumno@debian:~$ mv dirpueba/ dirprueba2
alumno@debian:~$ ls
Descargas   Documentos  Imágenes  Plantillas  Vídeos
dirprueba2  Escritorio  Música    Público
alumno@debian:~$
```

Imagen 8.Comando mv para renombrar archivos o directorios.

10.1.8. Impresión de archivos

Con impresión de archivos, nos referimos a la impresión por pantalla, no en papel, es decir, a la visualización del contenido de un archivo. Siempre que queramos que su contenido se pueda ver correctamente, todos sus caracteres deben de encontrarse en formato ASCII. Si no, no será legible.

Uno de los principales comandos que tenemos a la hora de mostrar en la línea de comandos la información de un archivo es:

```
cat [archivos] [ { > | >> | < | << } archivo ]
```

Se muestra la información de todos los archivos concatenada de manera secuencial. Además, la salida puede ser redireccionada a un archivo con las siguientes opciones:

+ **>** sobrescribe en caso de que el archivo exista, si no es así, lo crea.
+ **>>** el contenido se añade a un fichero existente justo después del que ya almacena. Si no existe el fichero lo crea.

Esta orden entonces podemos decir que se usa para crear archivos de una extensión corta rápidamente.

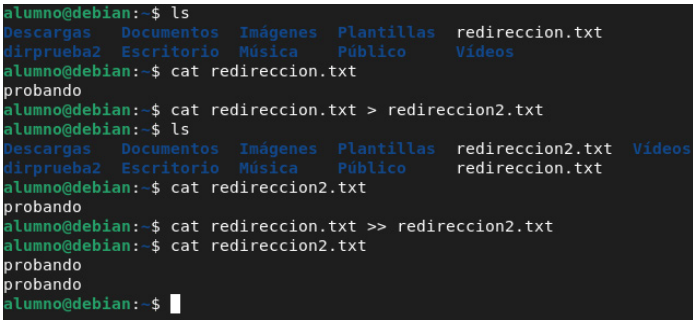

Imagen 9.Comando cat.

Podemos ver en la imagen anterior como se ha usado la orden para: crear un nuevo fichero, añadir información al fichero, ver información del fichero.

Tenemos otros dos comandos muy usados para la visualización de la información, estos permiten que el contenido se visualice por páginas y son:

> **more [ficheros]**
> **less [ficheros]**

De estos dos comandos se suele usar más el segundo, puesto que es más potente y versátil que el primero.

Además, tenemos también estos dos comandos:

> **head [-número]** → Muestra las primeras líneas del contenido, si no se indica ningún número, por defecto muestra las 10 primeras.
> **tail [-número]** → Muestra las diez últimas líneas del contenido a no ser que se le indique otro número.

SABER MÁS

Cuando trabajemos con los comandos more y les, debemos de saber que se navega por su impresión en pantalla del siguiente modo:

+ Barra espaciadora: salta de página.
+ Enter: salta de línea.
+ q: sale de la visualización.
+ /texto: busca el texto que se inserte aquí (se pueden usar expresiones regulares).
+ n: busca la siguiente coincidencia o expresión regular del texto.
+ :n: salta de fichero.
+ :p: retrocede al fichero anterior.

10.1.9. Cuenteo de un fichero

El comando wc nos permite que se cuenten las líneas, palabras y bytes que tiene un fichero. Sigue la siguiente estructura:

```
wc [-lwcL] ficheros
```

Las opciones para que nos indique el número que queremos y que se pueden usar de manera asilada o conjunta son:

- **l:** número de líneas.
- **w:** número de palabras.
- **c:** número de bytes.
- **L:** longitud de la línea más larga.

```
alumno@debian:~$ wc -lwcL redireccion2.txt
 2  2 18  8 redireccion2.txt
alumno@debian:~$
```

Imagen 10.Comando wc.

En la imagen anterior podemos ver que se ha mostrado la información y que el fichero redireccion2.txt (que hemos creado antes), tiene 2 líneas, 2 palabras, 18 bytes, y que su línea más larga tiene 8 caracteres.

10.1.10. Ordenación de un fichero

Hay ocasiones en las que queremos que se nos muestre el contenido de un fichero con cierta ordenación sin necesidad de que nos salgan todos los parámetros como cuando ordenamos con `ls`. Para esta función se usa el comando `sort`, que, aunque tiene varias opciones, por defecto muestra el contenido ordenado por caracteres ASCII donde las letras minúsculas tienen un orden mayor que las mayúsculas.

Su sintaxis es:

```
sort [-fnru] [-t delimitador] [-k número_campo] archivos
```

Sus principales opciones son:

- **f:** no distingue entre minúsculas y mayúsculas.
- **r:** se muestra el orden invertido.
- **n:** se ordena de manera numérica y no alfabética.
- **u:** no muestra las entradas repetidas.
- **t delimitador:** se indica un delimitador o separador para los campos.
- **k:** indica por qué campo se va a empezar con la ordenación.

En la siguiente imagen vamos a ver un ejemplo de ordenación de un fichero con varias entradas.

```
alumno@debian:~$ cat orden
hoola
desarrollo
sistemas
probando
licencias
alumno@debian:~$ sort orden
desarrollo
hoola
licencias
probando
sistemas
alumno@debian:~$ sort -n orden
desarrollo
hoola
licencias
probando
sistemas
alumno@debian:~$ sort -r orden
sistemas
probando
licencias
hoola
desarrollo
alumno@debian:~$
```

Imagen 11.Comando sort.

Como podemos ver, aunque hemos usado la opción -n, no se ha cambiado el orden alfabético porque no lleva una numeración.

10.1.11. Entrada y salida estándar. Redirecciones

Por lo general, la gran mayoría de comandos de una terminal reciben la información que procesar mediante un flujo de entrada y los datos son enviados o mostrados mediante otro flujo de salida. Dependiendo del comando, el SO por defecto asigna entradas y salidas estándar para los dos flujos respectivamente.

Por defecto en el sistema existen tres ficheros que cumplen con entradas y salidas estándar, que son:

- Entrada estándar (stdin).
- Salida estándar (stdout).
- Salida de errores estándar (stderr).

Estos ficheros tienen un número que los describe, 0, 1 y 2, respectivamente.

Suele asociarse la entrada estándar al teclado y la salida estándar y salida de errores estándar a la pantalla, pero esto no es siempre así, porque hay veces en las que las entradas o salidas se pueden nutrir de otros ficheros del sistema.

Por último, como veremos a continuación este flujo de entrada y salida puede cambiarse con diversas opciones de los comandos.

Redirecciones de las salidas estándar

Como ya vimos cuando hablábamos del comando cat, se puede usar el operador > para que la salida de un comando se redirija y se añada a un fichero en vez de mostrarse por pantalla.

Se sigue la siguiente sintaxis:

```
Comando > fichero
```

Como vimos, si el fichero no existe, lo crea. Y si existe, sobrescribe su contenido. Podríamos además usar la expresión 1> que daría el mismo resultado al llevar el descriptor del fichero de salida estándar.

Se puede usar también el operador >>, que hará la misma función, pero en este caso si el fichero existe, el contenido se anexa al que ya tiene, pero no se sobrescribe, si no existe, lo crea. La sintaxis es la misma que la anterior.

```
Redirecciones de la entrada estándar
```

Al igual que se redirecciona la salida, también se puede redireccionar la entrada, que se ejecuta con el operador <.

La sintaxis que sigue es:

```
Comando < fichero
```

También se puede usar un tipo de redirección, que lo que hace es que permite escribir tantas líneas como queramos hasta que se encuentre con el delimitador que nosotros establezcamos.

La sintaxis de este es como la siguiente:

```
cat <<delimitador
```

En la siguiente imagen podemos ver un ejemplo de cada uno de los dos casos:

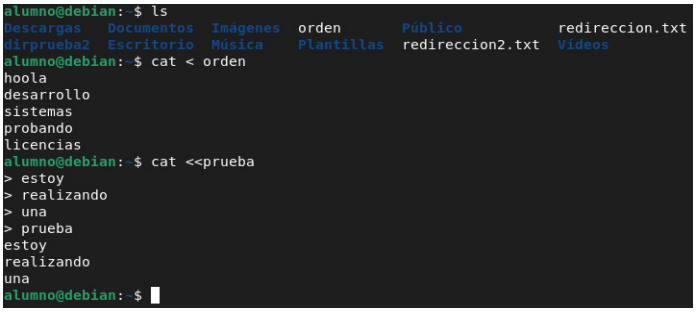

Imagen 12. Redirecciones 1.

Como podemos ver en la imagen anterior, al usar la redirección con un delimitador, cuando escribimos en el prompt dicho delimitador, la orden muestra el resultado y termina.

Redirección de la salida de error estándar

Habrá ocasiones en las que un comando nos de error y esto es normal, porque nos podemos equivocar en la sintaxis, en los argumentos, etc. Cuando esto sucede, se suele mostrar un resultado de error por pantalla, pero si por ejemplo queremos almacenar estos errores para llevar un registro, podríamos hacerlo también, usando el operador `2>`. Su sintaxis es la que sigue:

```
comando 2> fichero
```

Se podría también usar el operador `2>>` para que el fichero no se sobrescriba como pasaba con la salida estándar.

Por ejemplo, si realizamos un **ping** a una dirección que no existe, se vería del siguiente modo:

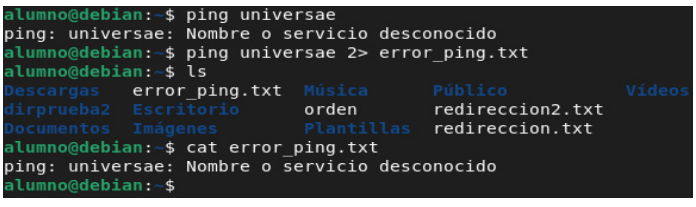

Imagen 13. Redirecciones 2.

Redirección de la salida estándar y la salida de error estándar al mismo destino

Al igual que se redirigen la salida estándar y la salida de errores estándar, también podemos realizar una redirección de ambas a la vez al mismo fichero siguiendo las sintaxis anteriores, pero usando el operador `&>`.

Si seguimos con el ejemplo del **ping** anterior:

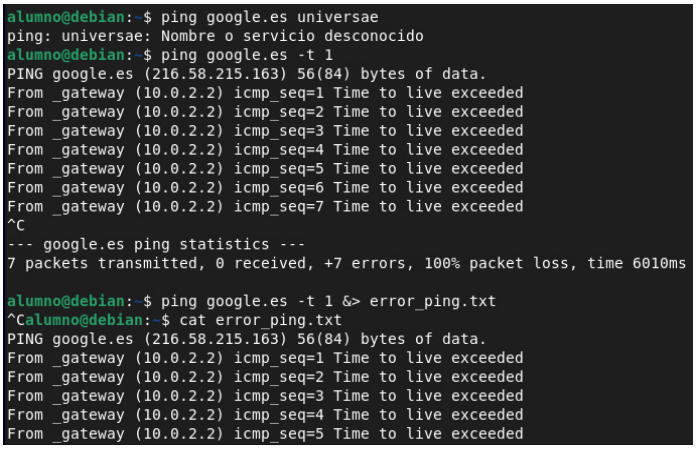

Imagen 14. Redirecciones 3.

Podemos fijarnos en que el ping se ha parado de manera natural porque era continuo para que mostrase algún tipo de error

Redirección de la salida estándar y la salida de error estándar de una orden con la entrada estándar de otra orden

Tenemos también en Linux la opción de concatenar la salida de varios comandos para que sirvan como entrada a otros y viceversa. Esto se realiza con el uso de tuberías o pipes que se representan con el símbolo ASCII "|". La sintaxis de estas redirecciones es muy sencilla:

```
Comando1 | comando2
```

Para redireccionar ambas salidas, estándar y de error estándar se usa el símbolo `|&` siguiendo la misma sintaxis que la salida estándar.

Y, además, entre comandos podemos usar el comando `tee` para añadir información a la salida del anterior y consecuentemente a la entrada del siguiente.

Vamos a ver un ejemplo:

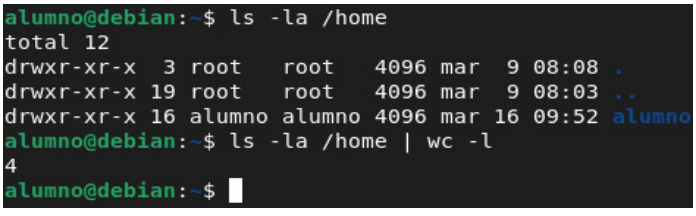

Imagen 15. Redirecciones 4.

Podemos ver que la salida del primer comando mostraba 4 líneas, por eso cuando hemos ejecutado la concatenación siguiente, nos ha devuelto dicho número.

Redirección de la salida de error a la salida estándar

Podemos usar el operador 2>&1 si queremos que la salida de error salga por el mismo sitio que la salida estándar, pero realmente no tiene mucho efecto y no suele usarse.

10.1.12. Procesamiento de textos

Linux tiene multitud de comandos que se pueden usar para el procesamiento de textos, pero los dos principales y más usados con cut y grep. El primero se usa para seleccionar que filas i, caracteres o grupos de caracteres de un fichero se van a mostrar. El segundo se usa para el filtrado del propio archivo por palabras clave o expresiones regulares.

La sintaxis de cut es la siguiente:

```
cut -c caracteres | -f columnas [-d delimitador] fichero
```

Sus opciones son las siguientes:

- **-c caracteres:** se corta el fichero por los caracteres que se especifiquen.
- **-f lista de columnas:** se seleccionan los campos que se hayan establecido. Por defecto, se delimitan las columnas por espacios, tabuladores o fin de línea., pero existe la opción -d que nos permitirá establecer un delimitador propio.
- **El comando siempre debe de ir con una de estas dos opciones, pero nunca con las dos juntas.**

En la siguiente imagen vamos a ver un ejemplo de como podemos cortar el fichero /etc/networks.

```
alumno@debian:~$ cat /etc/networks
default         0.0.0.0
loopback        127.0.0.0
link-local      169.254.0.0

alumno@debian:~$ cut -f1 -d. /etc/networks
default         0
loopback        127
link-local      169

alumno@debian:~$ cut -c1-10 /etc/networks
default         0
loopback        1
link-local

alumno@debian:~$
```

Imagen 16. Comando cut.

Como podemos ver, nos valemos del corte tanto por caracteres como por campos con un delimitador que hemos seleccionado.

La sintaxis del comando `grep` para filtrar es:

```
grep [-nvlicw] patrón ficheros
```

Y sus opciones son las siguientes:

- **l:** solo se van a mostrar los ficheros que contengan el patrón que hemos seleccionado.
- **i:** no distingue entre mayúsculas y minúsculas.
- **c:** nos muestra el número total de las líneas que contienen el patrón que se ha especificado.
- **w:** busca el patrón en una palabra completa y no como parte de una cadena de caracteres.
- **n:** se muestra el número de línea de cada una de las que tengan el patrón especificado.
- **v:** busca las líneas que no contengan el patrón que se ha pasado.

Por ejemplo, vamos a filtrar por el usuario root en /etc/group:

```
alumno@debian:~$ grep -ni root /etc/group
1:root:x:0:
alumno@debian:~$ grep -i root /etc/group
root:x:0:
alumno@debian:~$ grep -c root /etc/group
1
alumno@debian:~$ grep -cv root /etc/group
64
alumno@debian:~$
```

Imagen 17.Comando grep.

Podemos ver las cuatro ocasiones que lo hemos usado y que su resultado ha sido correcto.

Para el comando `grep` también podemos usar las expresiones regulares.

Existe la opción `-E` o el comando `egrep` que permite el uso de las expresiones regulares extendidas para patrones más complejos.

También existe la opción `--color` para marcar de color los resultados de coincidencia del patrón.

SABER MÁS

Todos los comandos que hemos visto en este punto y los que se verán más adelante tienen la opción man que muestra un manual de uso del comando en sí.

Un resumen de este manual sería la opción --help.

10.2. Los usuarios en Linux

10.2.1. Configuración de usuarios y grupos

Para la configuración de usuarios y grupos en Linux usaremos los archivos de configuración del sistema /etc/passwd y /etc/group respectivamente.

Hay otros archivos de configuración que también se tratarán de manera más indirecta, como el archivo /etc/shadow que almacena las contraseñas de los usuarios encriptadas.

Para poder trabajar con los archivos, aunque sea mediante comandos, debemos de conocer su estructura interna.

En el caso del fichero /etc/passwd, en cada línea nueva se almacena un usuario diferente, y, además, cada línea consta de siete campos que se delimitan por el símbolo ":" y que expresan lo siguiente:

1. **Login.** Almacena el nombre de usuario que se usa para el acceso al sistema.
2. **Password.** Alamcenamos la contraseña necesaria para que el usuario entre al sistema, vendrá marcada con "x" ya que se encuentra ubicada en el fichero /etc/shadow.
3. **UID.** El número de identificador de usuario único. El 0 corresponde al superusuario del sistema, del 1 al 99son las cuentas predeterminadas del sistema, del 100 al 999 son las cuentas administrativas del sistema y los nuevos usuarios se asociarán con un identificador a partir del 1 000.
4. **GID.** El número de identificado de grupo identifica el grupo principal del usuario.
5. **Información personal del usuario.** Cualquier información que se haya añadido como podría ser su nombre completo.
6. **Home o directorio de trabajo del usuario.** Es el directorio principal del usuario y donde se almacena por defecto toda su información. Además, es el directorio por defecto cuando el usuario accede al sistema.
7. **Shell.** Para identificar que intérprete de comandos usará el usuario del sistema.

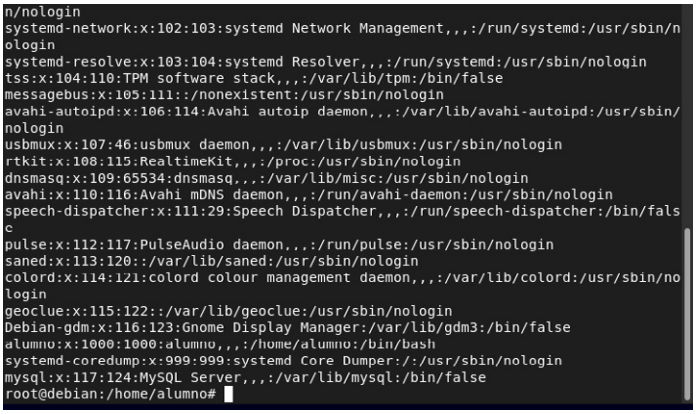

Imagen 18. /etc/passwd

Si por ejemplo nos fijamos en el usuario alumno de la imagen anterior, podríamos distinguir lo siguiente:

Estructura del fichero /etc/passwd						
alumno	**x**	**1 000**	**1 000**	**alumno,,,**	**/home/alumno**	**/bin/bash**
Nombre de usuario	Contraseña	ID de usuario (UID)	ID de grupo (GID)	Información de usuario	Directorio home del usuario	Shell

Un usuario administrador del sistema en Linux es el que tiene privilegios sobre la gestión de este, pero no tiene necesariamente por qué ser el usuario root. Para poder otorgar privilegios se puede usar el comando sudo (hay que habilitar su uso, lógicamente) o añadir el usuario a un grupo que tenga privilegios sobre configuraciones concretas asignados.

Para administrar los privilegios en Linux son muy usados los grupos, que permiten centralizar de manera más eficiente todos estos permisos hacia usuarios.

Para configurar los grupos se usará el fichero /etc/group.

Como pasaba con los usuarios, cada una de las filas hará referencia a un grupo distinto, y constará de cuatro campos separados de nuevo por el símbolo ":", con el siguiente orden:

1. **Nombre del grupo.** Es el nombre del grupo asociado a su identificador.
2. **Contraseña.** Aunque no se suele usar, como en los usuarios, se encuentra marcada con "x" que hace referencia a que se encuentran almacenadas en el fichero /etc/gshadow
3. **Identificador de grupo.** GID o número de identificación de grupo único.
4. **Lista de usuarios.** Se listan los usuarios que pertenecen a dicho grupo como grupo secundario.

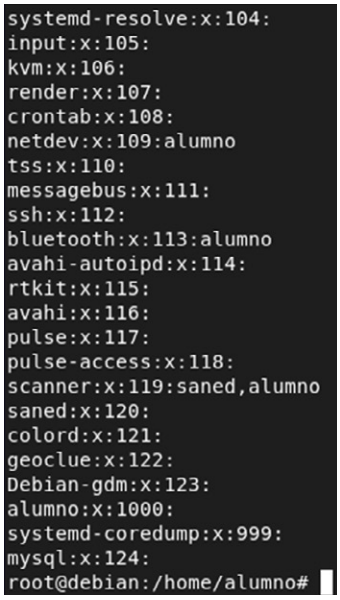

Imagen 19. /etc/group.

Si nos fijamos en el grupo alumno, de la imagen anterior:

Estructura del fichero /etc/group			
alumno	x	1000	
Nombre del grupo	Contraseña	GID	Usuarios pertenecientes al grupo.

Para identificarnos como usuario root del sistema deberíamos de ejecutar el siguiente comando en Debian:

```
su root
```

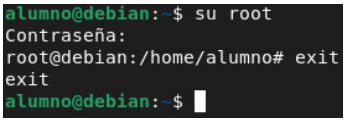

Imagen 20. Usuario root.

Y, como podemos ver, una vez que se haya establecido la contraseña, estaremos logueados como el superusuario del sistema y podremos ejecutar tareas administrativas. Si queremos salir de la sesión usaremos el comando `exit`.

10.2.2. Comandos de gestión de usuarios

Si queremos añadir en Linux un usuario nuevo por la línea de comandos, como superusuario ejecutaremos el comando useradd con la siguiente estructura:

```
useradd [-g grupo] [-G grupo[, grupo …]] [-d directorio_
home [-m]] [-p contraseña_encriptada] [-s shell] login
```

Este comando añadirá una línea nueva al fichero /etc/passwd con los datos aportados en el comando y además copiará los archivos del directorio /etc/skel, que es el que almacena por defecto los archivos de configuración del directorio de trabajo de un usuario común. Las opciones que más usa este comando son:

- **g grupo:** se usa para asignar el grupo principal del usuario. Todos los usuarios tienen por lo menos un grupo principal al que se pertenece, y todos los demás serán grupos secundarios. Si no se especifica esta opción, habrá un grupo por defecto con el mismo nombre que el del usuario.
- **G grupo:** se listan todos los grupos secundarios, separados por comas y sin espacios.
- **d directorio_home:** se establece el directorio home del usuario, donde el usuario trabajará de manera normal. Si no se especifica nada, se usará el directorio /home/nombre_usuario.
- **p contraseña_encriptada:** se especifica la contraseña de usuario que se encriptará para que no se pueda descubrir. Si no se especifica, no se podría loguear con este usuario al sistema.
- **m:** si no existe o no se especifica el directorio, lo crea y se copian los archivos de /etc/skel.
- **s shell:** indica cual será el shell por defecto del usuario a la hora de la ejecución de los comandos.

En el siguiente ejemplo creamos un usuario pepe sin especificar opciones:

```
root@debian:/home/alumno# sudo useradd pepe
```

Imagen 21. Comando useradd.

```
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
mysql:x:117:124:MySQL Server,,,:/var/lib/mysql:/bin/false
pepe:x:1001:1001::/home/pepe:/bin/sh
root@debian:/home/alumno#
```

Imagen 22. Nueva línea en /etc/passwd.

Podemos ver que en el ejemplo anterior se ha añadido una línea al fichero /etc/passwd y que su directorio home se ha situado en /home/pepe.

*Notemos que, aunque estamos logueados como superusuario, ese necesario que se indique la opción **sudo** para poder ejecutar dicho comando.

Pero no solo se pueden crear usuarios por comandos, sino que también se pueden modificar usando el comando `usermod`. La sintaxis de este comando es la siguiente:

```
usermod [-c comentario] [-g grupo] [-G grupo[, grupo …]] [-d directorio_home [-m]] [-p contraseña_encriptada] [-e fecha] [-f días] [-l nuevologin] [-L] [-U] [-s shell] login
```

Dentro de este comando hay muchas opciones que ya hemos descrito en el comando anterior, y las nuevas que se incorporan son:

- **c comentario:** establece los valores asociados al quinto campo de la línea añadida al fichero /etc/passwd.
- **e fecha.**
- **f días.**
- **l nuevologin:** se cambia el login anterior por uno nuevo que se aporte.
- **L.**
- **U.**

En el siguiente ejemplo le asignamos al usuario pepe el directorio de trabajo /home/probando.

```
root@debian:/home# sudo usermod pepe -d /home/probando -m
root@debian:/home#
```

Imagen 23. Comando usermod.

```
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nolo
mysql:x:117:124:MySQL Server,,,:/var/lib/mysql:/bin/false
pepe:x:1001:1001::/home/probando:/bin/sh
root@debian:/home#
```

Imagen 24. Línea modificada en /etc/passwd.

Vemos en el fichero /etc/passwd que se ha cambiado el directorio por el que nosotros hemos especificado.

Si, por último, queremos eliminar a un usuario lo haremos con el comando `userdel` que sigue la siguiente sintaxis:

```
userdel [-r] login
```

La opción `-r` hace que también se borre el directorio home del usuario.

Vamos a borrar ahora el usuario pepe.

```
root@debian:/home# sudo userdel pepe
root@debian:/home#
```

Imagen 25. Comando userdel.

```
alumno:x:1000:1000:alumno,,,:/home/alumno:/bin/bash
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
mysql:x:117:124:MySQL Server,,,:/var/lib/mysql:/bin/false
root@debian:/home#
```

Imagen 26. /etc/passwd.

Y podemos ver en la imagen como este ya no aparece en el fichero /etc/passwd.

Además, ya vimos que los sistemas Linux, y Debian en concreto son sistemas multiusuario, eso quiere decir que puede que haya más de un usuario trabajando al mismo tiempo en el mismo equipo. Si queremos comprobar que usuarios se encuentran trabajando en el sistema podemos usar el comando `who`, cuya estructura es la siguiente:

```
who [am i] [-u] [-H] [-q]
```

Y las opciones que sigue son:

- **am i:** que muestra cual es el usuario actual que está trabajando en dicha terminal. Aunque parezca una tontería esto tiene sentido cuando se trabaja en varias terminales con varios usuarios al mismo tiempo.
- **u:** muestra la información más relevante de los usuarios que están conectados al equipo.
- **H:** imprime las cabeceras.
- **q:** solo nos muestra el login de los usuarios y el número de usuarios conectados.

```
root@debian:/home# who
alumno   tty2         2022-03-22 15:33 (tty2)
root@debian:/home#
```

Imagen 27. Comando who.

Para añadir un grupo nuevo usaremos el comando `groupadd` con la siguiente sintaxis:

```
groupadd [-g GID] nombre_grupo
```

La opción -g se usa para darle nosotros mismo un identificador de grupo. Debemos de recordar que este siempre debe de ser superior a 1000.

```
root@debian:/home# sudo groupadd alumnos
root@debian:/home#
```

Imagen 28. Comando groupadd.

```
alumno:x:1000:
systemd-coredump:x:999:
mysql:x:124:
alumnos:x:1001:
root@debian:/home#
```

Imagen 29. Nueva línea en /etc/group.

Vemos que nada más crearlo se añade una nueva línea al fichero /etc/group.

Si lo que queremos es modificar las características de los grupos debemos de seguir el comando `groupmod`, y su sintaxis es:

`groupmod [-g GID] [-n nuevo_nombre] nombre_grupo.`

La única opción nueva que encontramos es -n que nos da la opción de cambiarle el nombre al grupo por uno nuevo.

En el siguiente ejemplo vamos a cambiarle el nombre al grupo creado anteriormente y también su GID por otro:

```
root@debian:/home# sudo groupmod -g 1003 -n alumnos_nuevo alumnos
root@debian:/home#
```

Imagen 30. Comando groupmod.

```
Debian-gdm:x:123:
alumno:x:1000:
systemd-coredump:x:999:
mysql:x:124:
alumnos_nuevo:x:1003:
root@debian:/home#
```

Imagen 31. Línea modificada en /etc/group.

Podemos apreciar en el fichero que los cambios se han llevado a cabo correctamente.

Para añadir usuarios a un grupo, el comando que se usará será `adduser [login_usuario] grupo`.

Veamos cómo sería añadir al usuario alumno al grupo creado y modificado:

```
root@debian:/home# sudo adduser alumno alumnos_nuevo
Añadiendo al usuario `alumno' al grupo `alumnos_nuevo' ...
Añadiendo al usuario alumno al grupo alumnos_nuevo
Hecho.
root@debian:/home#
```

Imagen 32. Comando adduser.

Para ver a que grupos pertenecen los usuarios, tenemos dos comandos que podemos usar: `groups [usuario]` e `id [usuario]`.

Como podemos ver, el segundo aporta más información que el primero de estos:

```
root@debian:/home# groups alumno
alumno : alumno cdrom floppy audio dip video plugdev netdev bluetooth scanner al
umnos_nuevo
root@debian:/home# id alumno
uid=1000(alumno) gid=1000(alumno) grupos=1000(alumno),24(cdrom),25(floppy),29(au
dio),30(dip),44(video),46(plugdev),109(netdev),113(bluetooth),119(scanner),1003(
alumnos_nuevo)
root@debian:/home#
```

Imagen 33. Comando groups e id.

Si queremos eliminar a un usuario de un grupo el comando que hay que usar sería deluser [login_usuaro] nombre_grupo.

Para eliminar al usuario alumno del grupo alumnos_nuevo:

```
root@debian:/home# sudo deluser alumno alumnos_nuevo
Eliminando al usuario `alumno' del grupo `alumnos_nuevo' ...
Hecho.
root@debian:/home#
```

Imagen 34. Comando deluser.

Por último, la opción de eliminar un grupo viene estipulada por el comando:

```
groupdel nombre_grupo
```

Si eliminamos el grupo que hemos creado....

```
root@debian:/home# sudo groupdel alumnos_nuevo
root@debian:/home#
```

Imagen 35. Comando groupdel.

```
alumno:x:1000:
systemd-coredump:x:999:
mysql:x:124:
root@debian:/home#
```

Imagen 36. Fichero /etc/group.

Vemos que ya no aparece en el fichero /etc/group, pero ¿y si quisiéramos eliminar un grupo principal de usuario? Vamos a verlo.

```
root@debian:/home# sudo groupdel alumno
groupdel: no se pudo eliminar el grupo primario del usuario «alumno»
root@debian:/home#
```

Imagen 37. Eliminar grupo principal.

Esto nos va a dar error porque es el grupo primario de un usuario que reside en el sistema, por lo cual primero habría que borrar su usuario.

10.2.3. Usuarios y grupos predeterminados

En Linux no solo existen las cuentas de usuario predeterminadas de superusuario, sino que además existen algunas cuentas de grupo que se pueden ver en /etc/group que también se crean cuando se instala el sistema. Estas cuentas surgen para que el sistema envíe una serie de permisos a estas y que sirvan para la gestión de este. En el siguiente cuadro podemos ver algunas de estas:

Grupos predeterminados

Grupos	Descripción
adm	Grupo de administración que permite accesos a archivos de registro y comandos como **sudo** y **su**.
users	Grupo de todos los usuarios estándar.
nobody	Sin privilegios.
root	Administración sin ninguna restricción sobre todo el sistema.
tty	Privilegios sobre dispositivos.
lpadmin	Contiene los privilegios sobre los dispositivos conectados al puerto paralelo.

10.3. Identificación y administración de procesos en Linux

Los procesos se identifican gracias a un identificador único denominado PID, Identificador de proceso. El PCB de cada proceso almacena información acerca de este, principalmente:

- El PID del proceso.
- Identificación del proceso padre, PPID.
- Usuario propietario.
- Valores del estado del proceso en el momento de producirse el cambio de contexto,
- Estado.
- Valores de referencia de memoria RAM.
- Ficheros abiertos.
- Buffers de memoria usados.

Para obtener información acerca de los procesos del sistema usaremos el comando `ps [modificadores]`. Este comando tiene un sinfín de opciones y una potencia mayor aún, por lo que para poder verlos todos debemos de acceder a su manual de ayuda con el comando `man ps.`

No obstante, los principales modificadores son:

Para obtener información de todos los procesos del sistema:

- **ps aux**
- **ps -ef**

Para imprimir información jun a un árbol de procesos.

- **ps axjf**

```
alumno@debian:~$ ps aux
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root           1  0.0  0.5 164032 10356 ?        Ss   06:54   0:02 /sbin/init
root           2  0.0  0.0      0     0 ?        S    06:54   0:00 [kthreadd]
root           3  0.0  0.0      0     0 ?        I<   06:54   0:00 [rcu_gp]
root           4  0.0  0.0      0     0 ?        I<   06:54   0:00 [rcu_par_gp]
root           6  0.0  0.0      0     0 ?        I<   06:54   0:00 [kworker/0:0H
root           9  0.0  0.0      0     0 ?        I<   06:54   0:00 [mm_percpu_wq
root          10  0.0  0.0      0     0 ?        S    06:54   0:00 [rcu_tasks_ru
root          11  0.0  0.0      0     0 ?        S    06:54   0:00 [rcu_tasks_tr
root          12  0.0  0.0      0     0 ?        S    06:54   0:00 [ksoftirqd/0]
root          13  0.0  0.0      0     0 ?        I    06:54   0:00 [rcu_sched]
root          14  0.0  0.0      0     0 ?        S    06:54   0:00 [migration/0]
root          15  0.0  0.0      0     0 ?        S    06:54   0:00 [cpuhp/0]
root          17  0.0  0.0      0     0 ?        S    06:54   0:00 [kdevtmpfs]
root          18  0.0  0.0      0     0 ?        I<   06:54   0:00 [netns]
root          19  0.0  0.0      0     0 ?        S    06:54   0:00 [kauditd]
root          20  0.0  0.0      0     0 ?        S    06:54   0:00 [khungtaskd]
root          21  0.0  0.0      0     0 ?        S    06:54   0:00 [oom_reaper]
root          22  0.0  0.0      0     0 ?        I<   06:54   0:00 [writeback]
root          23  0.0  0.0      0     0 ?        S    06:54   0:01 [kcompactd0]
root          24  0.0  0.0      0     0 ?        SN   06:54   0:00 [ksmd]
root          25  0.0  0.0      0     0 ?        SN   06:54   0:00 [khugepaged]
root          43  0.0  0.0      0     0 ?        I<   06:54   0:00 [kintegrityd]
```

Imagen 38. Comando ps aux

El comando `ps aux` muestra una serie de información que se establece en la cabecera:

- Usuario propietario del proceso.
- PID del proceso.
- CPU consumida en porcentaje.
- Memoria RAM consumida en porcentaje.
- Tamaño del proceso en la memoria virtual en KB.
- Tamaño de la memoria residente de proceso en KB.
- Terminal de lanzamiento.
- Estado del proceso.
- Tiempo de inicio del proceso.
- Tiempo de CPU consumido.
- Comando que lo ejecuta.

Los estados de un proceso en el comando pueden ser los siguientes:

Estados de un proceso	
Estado	**Descripción**
R	Ejecutándose o listo para ser ejecutado. (Runnable)
S	Bloqueado o durmiendo (Sleeping).
T	Parado (Trace).
Z	Zombi (proceso muerto pero el proceso padre no ha detectado su final).
I	Inactivo en creación (idle)
N	Con prioridad menor de lo normal (NICE).
<	Con prioridad mayor de lo normal.
+	Se encuentra en el grupo de procesos en primer plano.
s	Proceso líder de sesión.
L	Proceso multihilo.

Si en cambio, lo que queremos es ver una actualización constante de cómo evolucionan los procesos del sistema, el comando a ejecutar será `top`, y para salir de este comando pulsaremos la tecla "q".

Si ejecutamos el comando veremos que antes de listar los procesos, tenemos una serie de líneas que nos aportan información sobre el sistema.

- **Línea 1:** hora actual, tiempo del sistema encendido, número de usuarios y carga media en intervalos de 1,5 y 15 minutos, respectivamente.
- **Línea 2:** número de tareas, número de proceso en estad, ejecutándose o listos, bloqueados o hibernando parados y zombis respectivamente.
- **Línea 3:** tiempos de CPU, de usuario, kernel, etc.
- **Línea 4:** tamaño en MB de memoria física en total, libre. Usada y utilizada por buffer.
- **Línea 5:** tamaño en MB de memoria virtual total, libre usada y disponible.

Una vez terminadas estas líneas, la salida de procesos es similar a ps y sus opciones al igual que con el comando anterior, son muchas y muy específicas, por lo que antes d usarlo es recomendable leer su manual.

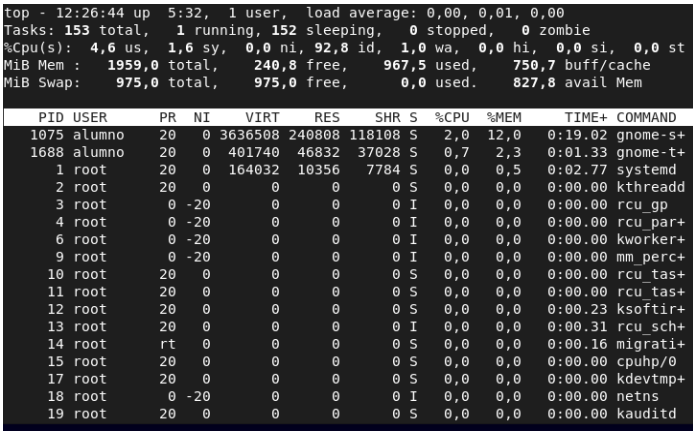

Imagen 39. Comando top

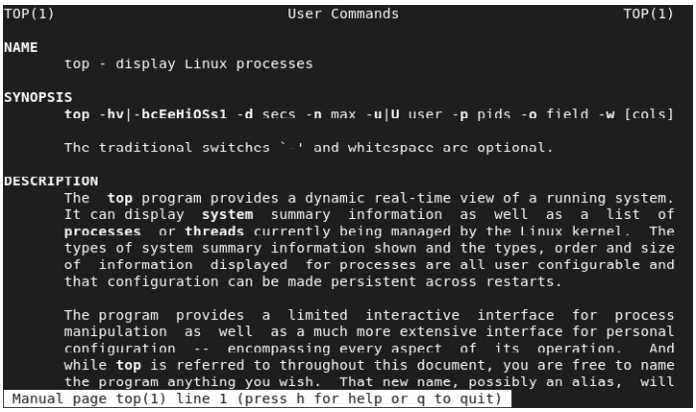

Imagen 40.man top

SABER MÁS

De manera lógica, si usamos estos comandos con el usuario root, su salida será más amplia.

www.universae.com