Unidad 4

Configuración de sistemas operativos

Sistemas Informáticos

Índice

Introducción

En cualquier sistema operativo, la función básica que ha de ejecutar este es la de ofrecer al usuario una gestión eficiente y amigable de los recursos. Para que, además, el usuario se sienta cómodo operando en el sistema y con sus aplicaciones, debe de establecerse un entorno seguro y que haya dos tipos de usuarios con diferentes niveles de privilegios, que son el usuario estándar y el usuario administrador.

El usuario estándar no tiene por qué conocer todos los entresijos de la configuración del sistema, pero sí que es recomendable que conozca ciertas configuraciones de su propia cuenta o algo relativo a la administración de sus procesos y dispositivos.

En cambio, el usuario que sea administrador del equipo debería de conocer todo el manejo del sistema y cuáles son las herramientas que este aporta para su administración.

Además, el administrador se ocupará de la gestión de todos los procesos del sistema, ya que esta es la función mayoritaria del sistema, que controla lo que realiza todo el sistema operativo en todo momento y si se controlan los procesos, será más fácil controlar otros aspectos como pueden ser la gestión de los archivos o la seguridad del sistema.

Realmente, es el propio sistema el que gestiona los procesos y los tiempos de acceso y ejecución de cada uno de ellos, y es tarea del administrador revisar que todo esté funcionando de manera correcta.

En temas de seguridad, el administrador debe también velar por ella administrando lo más eficientemente posible los usuarios y grupos que se pueden conectar al sistema.

En esta unidad por lo tanto se tratarán la gestión de usuarios y procesos en sistemas Linux y Windows trabajando con herramientas diferentes dependiendo del sistema en el que nos encontremos.

Para finalizar la unidad también sabremos que existen una serie de aplicaciones que aportan ayuda para que el sistema se mantenga de manera adecuada y lo más eficiente posible.

Al finalizar esta unidad

- Sabremos cuales son los fundamentos de gestión de usuarios y gestión de procesos.
- Seremos capaces de crear cuentas de usuario locales y grupos.
- Podremos asegurar el acceso al sistema mediante directivas de cuenta y de contraseñas.
- Protegeremos el acceso a la información mediante permisos locales.
- Conoceremos diferentes mecanismos para la gestión de procesos.
- Emplearemos comandos para realizar tareas básicas de configuración y monitorización del sistema.
- Conoceremos diferentes herramientas para el mantenimiento de y optimización del sistema.
- Sabremos operar con software de automatización de tareas.

4.1. Gestión de usuarios por línea de comandos en Linux

Como vimos en la unidad anterior, en Linux todo son archivos, así mismo los usuarios y grupos, que se administran usando archivos o ficheros de configuración específicos. Estos ficheros de configuración no pueden ser cambiados por usuarios comunes del sistema, ya que solo los usuarios administradores o el usuario root del sistema tienen los privilegios necesarios para efectuar cambios en estas configuraciones. Aunque se podría tratar con el archivo de manera directa, lo recomendable y lo que se va a tratar en este punto de la unidad es efectuar los cambios mediante la ejecución de una serie de comandos.

4.1.1. 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.** Almacenamos 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.

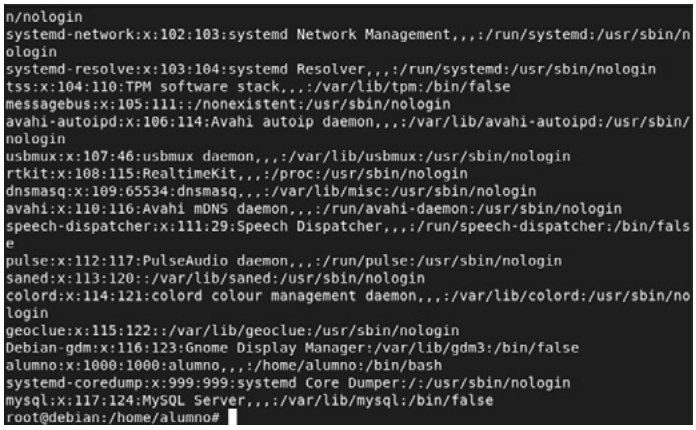


Imagen 1. /etc/passwd.

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

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

Cuadro 1. Estructura del fichero /etc/passwd.

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.

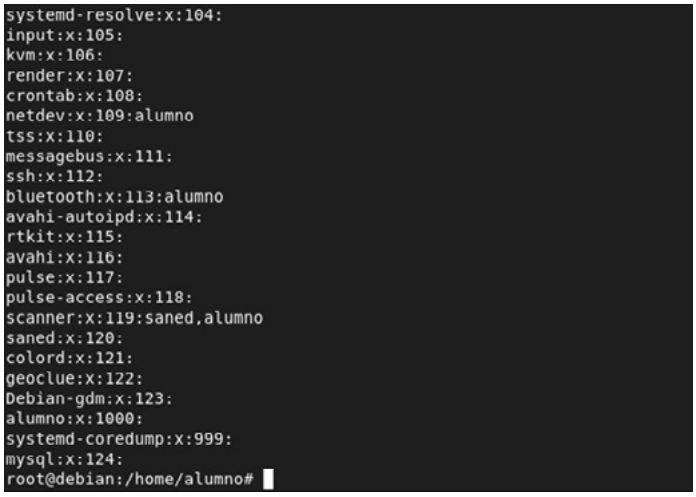


Imagen 2. /etc/group.

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

alumno	x	1000	
Nombre del grupo	Contraseña	GID	Usuarios pertenecientes al grupo

Cuadro 2. Estructura del fichero /etc/group.

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

```
su root
```

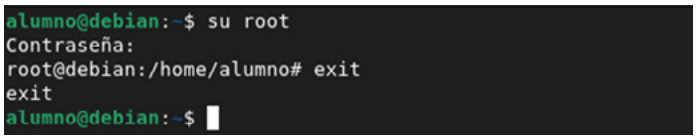


Imagen 3. Usuario root.

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**.

4.1.2. 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 4. 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 5. Nueva línea en /etc/passwd.

NOTA

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

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.

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_encrip-
tada] [-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 6. 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 7. 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 8. 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 9. /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.

Imagen 10. 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.

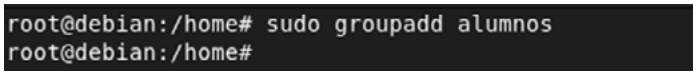


Imagen 11. Comando groupadd.

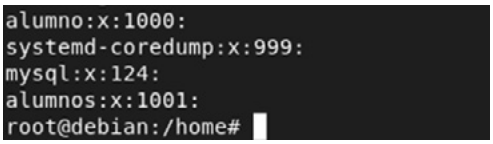


Imagen 12. Nueva línea en /etc/group.

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:

Imagen 13. Comando groupmod.

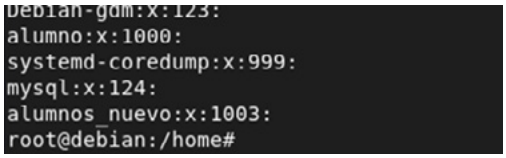


Imagen 14. Línea modificada en /etc/group.

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:

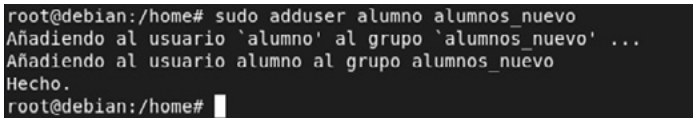


Imagen 15. Comando adduser.

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:

Imagen 16. 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:

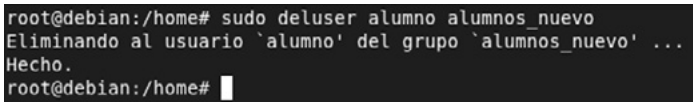


Imagen 17. Comando deluser.

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

`groupdel nombre_grupo`

Si eliminamos el grupo que hemos creado....

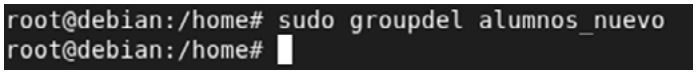


Imagen 18. Comando groupdel.

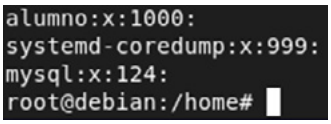


Imagen 19. Fichero /etc/group.

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

Imagen 20. 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.

4.1.3. 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	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.

Cuadro 3. Algunos grupos predeterminados.

Ahora, para cambiar permisos sin tener que tocar estos permisos, vamos a cambiar tanto el propietario como el grupo propietario de un archivo.

Lo que vamos a hacer es, por ejemplo, de la imagen de arriba, cambiar al archivo prueba para que su propietario y grupo sea el de todos los demás en los dos contextos.

1. Abrimos una terminal y nos logueamos como root.
2. Nos dirigimos a la ruta /home del usuario, aunque estaremos por defecto.
3. Ahora, para cambiar el propietario usamos el comando `chown nuevo_propietario archivo`. Y después usaremos ls -l para ver si se ha realizado el cambio.

```
root@debian:/home/miguel# chown miguel prueba
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 miguel 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
root@debian:/home/miguel#
```

Imagen 21. Cambiar propietario en Linux.

4. Si queremos cambiar el grupo lo que haremos será ejecutar el comando `chgrp grupo archivo`.

```
root@debian:/home/miguel# chgrp miguel prueba
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 miguel miguel    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
root@debian:/home/miguel#
```

Imagen 22. Cambiar grupo propietario en Linux.

4.1.4. Seguridad de cuentas de usuarios y contraseñas

Cuando se intenta acceder al sistema o a ciertas aplicaciones, Linux usa el llamado sistema centralizado de autenticación o Linux-PAM. Este sistema ayuda a que haya una mayor seguridad del sistema y que sea correspondiente para cada aplicación, y dependiendo del comando empleado variará el método de autenticación.

> Este comportamiento se puede apreciar en ejemplos anteriores cuando aun estando logueados como root debíamos de usar el comando sudo para ejecutar ciertas acciones.

Para que haya una mayor seguridad en las cuentas de usuario se establecen contraseñas para los distintos usuarios y estas se configuran en el fichero de administración /etc/shadow del sistema y al que recurre Linux-PAM cuando registra autenticaciones en el sistema.

En cada fila se almacena un usuario y cada una de estas filas se compone de ocho campos separados de nuevo por el carácter ":":

- Login del usuario.
- Password encriptada. Se pueden encriptar con diferentes algoritmos y dependiendo del comienzo del campo veremos cual es el algoritmo usado:
  - 1 → MD5
  - 5 → SHA-256
  - 6 → SHA-512
  - …
- Días transcurridos desde el 1/1/1970. Estos días determinan cuando se cambió por última vez la contraseña.
- Número de días hasta que se pueda cambiar la contraseña.
- Número máximo de días de validez de la contraseña.
- Número de días de aviso de caducidad de contraseña.

- Días que deben pasar para deshabilitar la cuenta tras la caducidad.
- Días desde 1/1/1970 para que la cuenta sea deshabilitada.

```
root@debian:/home# cat /etc/shadow
root:$y$j9T$Mch3mnGnFO2qIPWwCZx.U/$VNDGaLSDCFjqMjxv9qlBP.AANwKnqolmKUB5Rvjez48:1
9060:0:99999:7:::
daemon:*:19060:0:99999:7:::
bin:*:19060:0:99999:7:::
sys:*:19060:0:99999:7:::
sync:*:19060:0:99999:7:::
games:*:19060:0:99999:7:::
man:*:19060:0:99999:7:::
lp:*:19060:0:99999:7:::
mail:*:19060:0:99999:7:::
news:*:19060:0:99999:7:::
uucp:*:19060:0:99999:7:::
proxy:*:19060:0:99999:7:::
www-data:*:19060:0:99999:7:::
backup:*:19060:0:99999:7:::
list:*:19060:0:99999:7:::
irc:*:19060:0:99999:7:::
gnats:*:19060:0:99999:7:::
nobody:*:19060:0:99999:7:::
_apt:*:19060:0:99999:7:::
systemd-timesync:*:19060:0:99999:7:::
systemd-network:*:19060:0:99999:7:::
systemd-resolve:*:19060:0:99999:7:::
```

Imagen 23. Fichero /etc/shadow.

Si cogemos la línea correspondiente al usuario alumno, tendremos la siguiente salida:

```
alumno:$y$j9T$3Cs7v8qPMIEuNsDx9WY9A.$ggfUjzbHnKXx6zxdFN2ZCBKeBLTwAQbh2lShpqMYXh.
:19060:0:99999:7:::
```

Imagen 24. Línea alumno.

alumno	yj9T$3Cs7....	19060	0	99999	7		
Nombre de usuario	Contraseña encriptada	Último cambio	Mínimo	Máximo	Aviso	Inactivo	Caducidad

Cuadro 4. Estructura del fichero /etc/shadow.

Si creamos un usuario sin contraseña, el sistema no le asigna ninguna por defecto, por lo que este usuario realmente no podrá iniciar sesión. Pero el usuario root como superusuario del sistema puede cambiar las contraseñas de usuario (o un usuario con privilegios de sudo).

> Si un usuario no ha iniciado sesión en el sistema aún, su directorio home en caso de no existir tampoco habrá sido creado aún, este se crea cuando el usuario realiza el primer inicio de sesión.

Para asignar una contraseña a un usuario usaremos dos comandos distintos:

- **passwd.** Este comando modifica una contraseña de usuario o la crea en caso de no tenerla. Una vez creada el sistema la encripta y posteriormente la almacena en el fichero /etc/shadow. El usuario root puede modificar todas las contraseñas usando el comando passwd [usuario], pero un usuario común solo puede modificar su propia contraseña.
- **openssl passwd.** Este es un comando criptográfico que permite que la contraseña se genera en formato hash. Su estructura es:

  ```
  openssl passwd [opciones] "contraseña_ecnriptar".
  ```

  - Al igual que pasaba antes, un usuario solo puede modificar la suya propia.
  - Las opciones que más emplean son:
    1. Genera la contraseña basada en MD5.
    5. Genera la contraseña basada en SHA-256.
    6. Genera la contraseña basada en SHA-512.

En el siguiente ejemplo vamos a generar una contraseña encriptada con SHA-512 y pasársela al usuario alumno.

```
root@debian:/home# openssl passwd -6 Universae123
$6$2l7/viKq/V5M9sfF$06rv1hwQWYUZkWBVFLBHOHFcXEM/fhGwM/rBOUMm7CRoE9eN.8wJKe1w9Lxg
b1RnKMuUiZrG/4le3XtDpwGRq1
root@debian:/home# passwd alumno
Nueva contraseña:
Vuelva a escribir la nueva contraseña:
passwd: contraseña actualizada correctamente
root@debian:/home#
```

Imagen 25. Comando openssl passwd.

Vemos que el comando nos solicita que le pasemos la contraseña que queremos que se especifique.

4.1.5. Acceso a recursos y permisos locales

En Linux tenemos tres niveles de permisos de manera principal, que son los siguientes:

- Permisos del propietario
- Permisos del grupo
- Permisos del resto de usuarios

Los permisos del propietario son los permisos que tiene el creador del archivo o directorio en primera instancia cuando se crean, estos también pueden cambiar ya sea porque cambiemos el propietario o porque se cambien los permisos de este.

En Linux es común que los usuarios pertenezcan a distintos grupos de trabajo ya que por defecto se añade un nuevo usuario a ciertos grupos. Estos permisos se aplican a todos los usuarios de un grupo predefinido para el archivo. Si no se pone nada, el grupo que se pone por defecto es el de por defecto del usuario propietario.

Estos son los permisos que afectan al resto de usuarios que no forman parte del grupo en cuestión antes mencionado ni son el propietario.

Para ver los permisos por ejemplo de los directorios del home de un usuario lanzamos el comando `ls -l`.

```
miguel@debian:~$ 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 26. ls -l.

Como podemos ver en la imagen de más arriba los permisos se dividen en grupos de tres en el mismo orden que se ha establecido al principio: propietarios, grupo, otros.

Además, vemos que hay hasta tres letras, r, w y x:

- **R:** es el permiso de **lectura** que permite que se lea el contenido del archivo en cuestión. No es suficiente para que se pueda ver el contenido de un directorio.
- **W:** es el permiso de **escritura** que nos permite hacer cambios dentro de un directorio o de un archivo en concreto.
- **X:** es el permiso de **ejecutar** que nos permite ejecutar programas o script de Linux. Además, es necesario que se acompañe junto al permiso de lectura si se quiere ver el contenido de un directorio.

También podemos ver que salen dos nombres justo después, el primero hace referencia al usuario propietario y el segundo hace referencia al grupo propietario.

4.1.6. Modificación de permisos

Para cambiar los permisos de usuario en Linux, usaremos el comando `chmod`, pero tenemos dos vertientes, una usando letras y otra dotación numérica.

Con letras, tenemos claro que los permisos que cambiaremos son los de los tres grupos de arriba nombrados.

Para esto necesitaremos saber que se identifican del siguiente modo: u, dueño; g, grupo y o, otros.

Ahora que ya tenemos en cuenta esto, vamos a cambiar los permisos del archivo prueba de modo que el grupo y otros usuarios también puedan modificar.

1. Abrimos un terminal y logueamos como root.
2. Lanzamos el siguiente comando `chmod u/g/o+-r/w/x archivo/directorio`

```
root@debian:/home/miguel# chmod go+w prueba
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-rw-rw- 2 miguel miguel    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 27. Cambiar permisos en Linux por letras.

3. Como podemos ver, se pone todo de manera consecutiva, por ejemplo, hemos puesto go para hacer referencia al grupo y a otros usuarios y +w para que quede claro que se añade el permiso de escritura.
4. Cuando hemos lanzado la opción ls -l, vemos que los permisos han sido cambiados.
5. Si, por otro lado, quisiéramos quitar de nuevo estos permisos, el comando sería `chmod go-w prueba`.

Ahora vamos a ver como se cambian los permisos en Linux con el método numérico. Para esto debemos de saber lo primero que cada permiso se asocia con un número:

- **R → 4**
- **W → 2**
- **X → 1**

El comando total queda del siguiente modo `chmod n1n2n3 archivo`, donde n1 son los permisos para el propietario, n2 para el grupo y n3 para los otros usuarios.

Cada uno de los números lo sacaremos de la suma de cada uno de los números de los permisos, por eso, el máximo permiso que se puede otorgar es el 777.

El ejemplo anterior en esta numeración sería:

```
chmod 666 prueba
```

4.1.7. Permisos por defecto

Los permisos 0666 y 0777 son los permisos originales y por defecto de un archivo y un directorio respectivamente. Esto se puede cambiar, porque sobre estos permisos viene aplicada una máscara de permisos que proporciona cuales son los permisos que deben tener los archivos y directorios al ser creados. Dependiendo del usuario y de la distribución de Linux en la que se trabaje, la máscara tendrá una u otra cantidad de permisos, pero por defecto suele ser 0022 o 0002.

Si queremos ver o modificar la máscara de permisos del sistema, debemos de usar el comando **umask [máscara]**. A este comando le pasaremos los permisos en octal, pero si empelamos la opción -S, se puede usar la notación simbólica.

La máscara del usuario root es, por ejemplo:

```
root@debian:/home/alumno# umask
0022
root@debian:/home/alumno#
```

Imagen 28. Comando umask.

La modificación de la máscara es temporal, y si quisiéramos que tomara efecto en el tiempo y cuando se cierre el sistema, deberíamos de hacerlo en ficheros de configuración específicos que de momento no vamos a ver en esta asignatura.

4.1.8. Configuración de perfiles

Los perfiles de usuario contienen una información que es común para todos los usuarios de un mismo perfil como pueden ser privilegios del sistema. Gracias a los archivos globales o locales, el perfil toma función cuando el usuario inicia sesión y finaliza cuando este se desconecta. La principal diferencia entre perfiles es que los archivos de configuración global afectan a todos los usuarios y además necesitan de permisos de root para ser modificado, mientras que los archivos de configuración locales que se almacenan en el home de cada usuario solo afectan a ese mismo usuario y además pueden ser modificados por él mismo.

Por defecto, estos archivos son:

Tipos	Archivos	Descripción
Archivos globales	/etc/skel	Directorio que contiene la plantilla para la creación de perfiles de usuarios.
	/etc/profile	Contienen la configuración genérica de perfiles cuando se inicia sesión en el sistema como login shell.
	/etc/bas.bashrc	Configuración genérica de los perfiles cuando se inicia sesión con Shell Bash interactivo.
Archivos locales	~/.bashrc	Configuración local de usuario cuando este inicia sesión con Shell Bash como non-login shell.
	~/.bash_logout	Configuración local de usuario que se ejecuta cuando se finaliza la sesión con Shell Bash.
	~/.profile	Configuración local de usuario cuando este inicia sesión en el sistema con un shell de inicio de sesión como login shell.

Cuadro 5. Archivos de configuración.

El configurar estos archivos requiere de una gran destreza por parte del administrador en cuanto a shell scripting y funcionamiento a nivel profundo del sistema operativo se refiere. De manera principal esta configuración se usa para:

- Asignar o personalizar variables.
- Ejecutar comandos al entrar o salir del sistema o del terminal en cuestión.
- Asignar el PATH que indica las rutas de búsqueda para ejecutar ciertos archivos.
- Personaliza el prompt.

El shell de Linux puede ser configurado con la ayuda de las variables, los alias, y las opciones.

Variables

Las variables son identificadores que almacenan cadenas de caracteres con la intención de usar dichos valores más tarde en tareas de configuración, gestión, etc. Una variable se define del siguiente modo:

```
VARIABLE = VALOR
```

Las variables pueden almacenar más de un valor distinto si los separamos por el carácter ":". Además, si alguno de los valores contiene espacios, debe de ir entrecomillado.

Si queremos ver el contenido de una variable usaremos **$VARIABLE.**

Por ejemplo, vamos a ver el contenido de la variable **$HOME** (más adelante veremos que almacena):

```
root@debian:/home/alumno# echo $HOME
/root
root@debian:/home/alumno#
```

Imagen 29. Comando echo.

Podemos apreciar que para poder visualizar el contenido de una variable es necesario usar el comando echo.

Si la variable estuviera vacía nos devolvería una cadena de caracteres vacía, es decir, nada.

Tenemos dos tipos de variables en las que fijarnos:

- **Variables globales o de entorno:** estas son las variables que el sistema conoce y que el shell puede usar en sus procesos internos. SI queremos ver una lista de estas variables podemos usar los comandos env y printenv. En el siguiente cuadro vemos un pequeño resumen de las más usadas:

Variables globales	Descripción
SHELL	Shell por defecto.
USER	Usuario actual.
PWD	Directorio de trabajo actual.
OLDPWD	Directorio de trabajo previo.
PATH	Conjunto de rutas de directorios que contienen ejecutables. Esto se realiza para que el usuario no tenga que ejecutar la ruta completa de los comandos aquí recogidos.
HOME	Directorio home del usuario que esté trabajando en la terminal.

Cuadro 6. Variables globales.

- **Variables locales o de shell:** son las variables que solo pueden ser interpretadas por el shell. Esto quiere decir que solo se reconoce dentro de este porque se ha definido así de manera local. Por eso, por ejemplo, un programa ejecutado en el shell no detectará dichas variables. Si ejecutamos el comando set aparecerán las variables globales y las de entorno. Las más comunes son:

Variables locales	Descripción
HOSTNAME	Nombre del equipo
IFS	Valores de separación en la línea de comandos.
PS1	Valor del prompt
UID	UID del usuario actual.

Cuadro 7. Variables locales.

Tenemos, además, los siguientes comandos para trabajar con variables:

- `export VARIABLE` → para establecer una variable global.
- `export -n VARIABLE` → para convertir una variable global en local.
- `unset VARIABLE` → Para eliminar una variable de manera total.

Si queremos por otro lado, que las variables asignadas permanezcan en el tiempo, habrá que definirla en el fichero ~/.bashrc.

Alias

Los alias permiten que en un terminal los comandos se ejecuten de manera personalizada. Esto quiere decir que asignamos un "nombre" a un comando en específico para facilitar su ejecución ya sea por sintaxis compleja, longitud, etc. Al igual que las variables locales, al finalizar la sesión, estos desaparecen a no ser que se definan en el fichero ~./.bashrc.

Los alias se configuran usando el comando **alias:**

- Para crear un alias → **alias nombre_alias='comandos'.**
- Para eliminar un alias → **unalias nombre_alias.**

En el siguiente ejemplo vamos a ver cómo crear un alias de **ls -l** y luego eliminarlo.

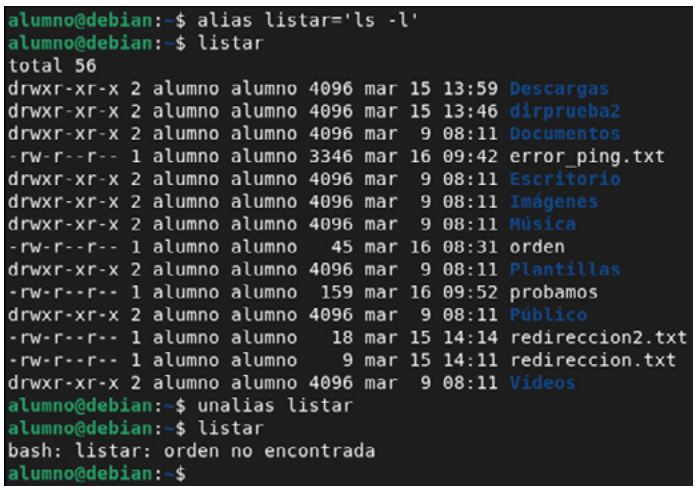


Imagen 30. Alias en Linux.

Vemos que una vez que se ha eliminado, el terminal no reconoce este alias.

4.2. Gestión de usuarios por interfaz gráfica en Windows

Lo primero que debemos hacer es definir dos conceptos principales:

Usuarios

Los usuarios son los ejecutores de las aplicaciones informáticas de un equipo en última instancia y necesitan de cuentas de acceso para poder identificarse en los equipos y poder trabajar en el sistema operativo en cuestión.

Si hablamos de tipos de usuarios, tenemos dos grandes grupos:

- **Usuarios locales:** son cuentas de usuario que se crean en el mismo equipo y sin acceso a un dominio de red, se suele dar este caso en equipos domésticos.
- **Usuarios globales:** estas son las cuentas que se crean en dominios o servidores y que tienen como función que se pueda conectar al dominio e iniciar sesión en distintos equipos unidos a este mismo dominio. Son los más usados en grandes empresas.

Sea un usuario local o un usuario global, siempre se guarda un registro de este usuario en el equipo de manera local.

Hemos de decir también, que en cuanto a usuarios globales nos referimos tenemos dos grupos que veremos más adelante en este mismo módulo: perfiles móviles y perfiles obligatorios.

Si hablamos de Windows, que es el principal software usado a nivel de sistema operativo, en cada perfil de usuario se almacena lo siguiente:

- **Configuración inicial.** Son los datos de los programas instalados solo por el usuario, el historial de dicho usuario y sus archivos temporales.
- **Información sobre preferencias de usuario.**
- **Datos de programa.** Almacena los datos específicos de un programa, tanto si es de 32 como de 64 bits (se suele almacenar en sitios distintos).
- **Entorno de red.** Accesos directos de red del usuario en cuestión.
- **Elementos del escritorio.**
- **Acceso directo a las impresoras configuradas.**
- **Menú de Inicio.** Esto realmente es un acceso directo a las aplicaciones o programas definidos en el inicio.
- **Mis documentos.** Documentos de sonido, video o simplemente de texto de pertenencia propia del usuario.

Grupos

Un grupo consiste en un grupo de usuarios que se encuentran categorizados en una misma clase para facilitar la gestión de estos. Un mismo usuario podrá pertenecer a varios grupos o a uno solo, pero siempre a uno. Normalmente estos grupos se usan para gestionar permisos y accesos a las distintas funciones de los sistemas.

De manera lógica, en un grupo puede haber varios usuarios, pero también puede haber uno solo, incluso ninguno.

Tenemos **dos tipos principales de grupos en los sistemas operativos:**

- **Grupos de seguridad.** Son usados principalmente para asignar privilegios y permisos, son los más comunes e los grupos.
- **Grupos de distribución.** Se suelen usar en entornos cloud o aplicaciones de correo para que la información llegue a varios usuarios al mismo tiempo, no suelen llevar consigo permisos ni privilegios especiales.

Vamos a hablar más específicamente de los grupos locales dentro de los grupos de seguridad, que al igual que pasaba con los usuarios son grupos que se crean en un equipo sin conexión al dominio y solo gestionan ese equipo en concreto, por lo que se añadirán a dichos grupos usuarios locales de manera general.

De manera principal, en Windows se crean 4 grupos locales principales, que son los siguientes:

- **Administradores:** son los que pertenecen a este grupo los usuarios que tienen el control sobre el sistema y se encargan de su gestión., cuentan con todo tipo de permisos.
- **Invitados:** es un grupo para usuarios que iniciarán sesión en el equipo de manera ocasional y cuentan con los permisos justos para poder trabajar en el equipo de manera temporal.
- **Usuarios:** el principal de los grupos y donde suelen encontrarse la mayoría de los usuarios, o al menos deberían. Los usuarios comprendidos en este grupo pueden hacer un uso completo del equipo a falta de funciones de instalación de aplicaciones o de gestión de permisos y recursos ya que eso se encuentra a cargo del administrador.
- **Usuarios avanzados:** estos usuarios realizan las mismas tareas que los usuarios normales en un principio per se le añade además el permiso para crear usuarios y grupos locales, pero solo podrán eliminar o modificar los de su propia creación.

4.2.1. Crear, modificar y editar usuarios y grupos

CREAR UNA CUENTA DE USUARIO LOCAL (WINDOWS 10):

1. Nos vamos a **Inicio → Configuración → Cuentas → Familia y otros usuarios**
2. Una vez aquí vemos que tenemos la opción **Otros usuarios y aquí tenemos que seleccionar 'Agregar otra persona para este equipo'**

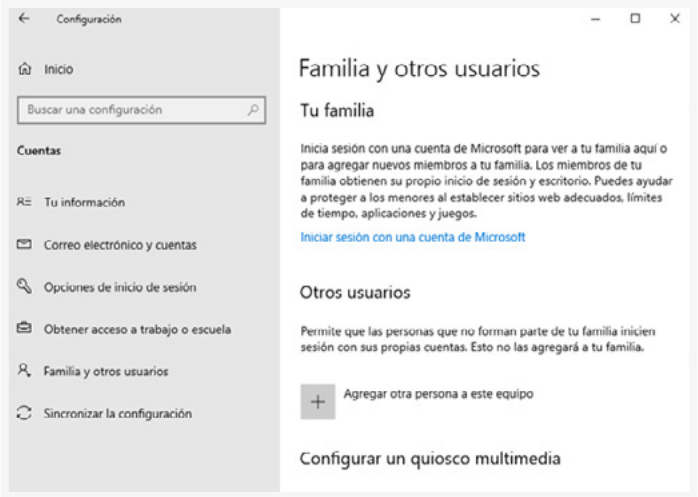


Imagen 31. Crear un usuario en Windows 10 1.

3. Nos pedirá que iniciemos con su cuenta de Microsoft y tendremos que seleccionar que no disponemos de inicio de sesión de Microsoft.
4. Después nos dirá que creemos una cuenta Microsoft y seleccionamos la opción de crear cuenta sin cuenta de Microsoft.
5. Nos aparecerá una ventana donde habrá que rellenar los datos del nuevo usuario.

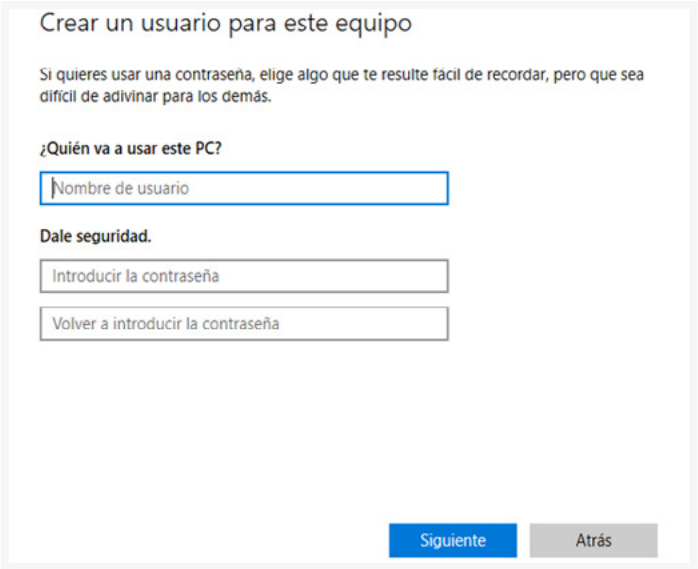


Imagen 32. Crear un usuario en Windows 10 2.

6. Ahora nos pedirá que, una vez puesta la contraseña, respondamos a unas preguntas de seguridad.
7. Ya estaría nuestro usuario creado y se vería de la siguiente forma:

Imagen 33. Crear un usuario en Windows 10 3.

EDITAR DATOS DE UNA CUENTA DE USUARIO (WINDOWS 10)

1. Seguimos los siguientes pasos: **Inicio → Panel de control → Cuentas de Usuario → Cuentas de usuario.**
2. Una vez aquí, seleccionaremos la opción **'Administrar otra cuenta'.**

Imagen 34. Editar cuentas de usuario en Windows 10 1.

3. Aquí nos aparecerán diversas opciones de las principales del usuario, entre las que destaca 'Cambiar el tipo cuenta'.

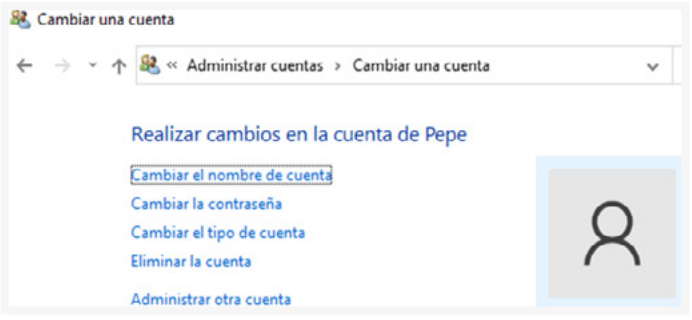


Imagen 35. Editar cuentas de usuario en Windows 10 2.

4. Con esta opción, lo que haremos será elegir si se trata de un usuario estándar o de un usuario administrador,

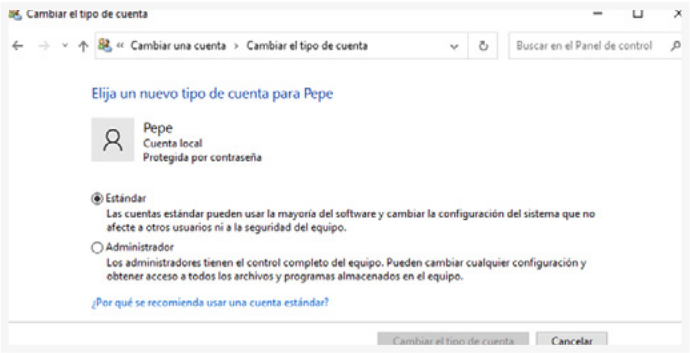


Imagen 36. Editar cuentas de usuario en Windows 10 3.

Debemos de tener en cuenta que este es solo uno de los métodos para tanto crear cuentas como editarlas en Windows 10, ya que hay numerosos.

ADMINISTRACIÓN LOCAL DE USUARIOS Y GRUPOS EN WINDOWS 10 CON POWERSHELL o POR CONSOLA

Windows 10 nos ofrece una herramienta moderna, llamada lusrmgr.msc o Administración local de usuarios y grupos.

Esta herramienta nos ayuda a que se haga una gestión de los usuarios de una manera aún más eficiente si se puede porque permite que se administren los usuarios y grupos locales.

Hay que tener en cuenta que esta función solo está disponible en Windows 10 Pro.

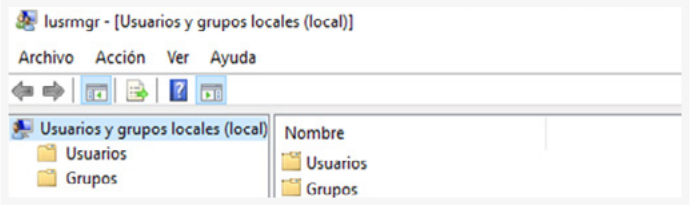


Imagen 37. Usuarios y grupos locales.

Además, para administrar los usuarios desde comandos, Windows nos ofrece dos comandos adicionales, para crear usuarios, Net User y para crear grupos, Net LocalGroup.

Como usar Net User

Para crear un usuario se usa el siguiente comando:

```
net user usuario contraseña /add
```

Para ver los usuarios que hay en una máquina local:

```
net user
```

Para modificar la contraseña de un usuario:

```
net user usuario nueva_contraseña
```

Para eliminar un usuario:

```
net user usuario /delete (/del)
```

Para desactivar un usuario:

```
net user usuario /active: no
```

Para activar un usuario:

```
net user usuario /active: yes
```

Cómo usar Net LocalGroup

Para agregar un grupo local:

```
net localgroup nombre_grupo /add
```

Para ver los grupos que hay en el equipo:

```
net localgroup
```

Para eliminar un grupo:

```
net localgroup nombre_grupo /del
```

Para añadir un comentario a un grupo:

```
net localgroup nombre_grupo /comment: "Implantación
de Sistemas Operativos"
```

Para añadir un usuario a un grupo:

```
net localgroup nombre_grupo usuario /add
```

Para eliminar un usuario de un grupo:

```
net localgroup nombre_grupo usuario /delete
```

4.2.2. Cambiar la ruta del perfil del usuario

Windows 10 nos da por defecto acceso a una serie de carpetas personalizadas para el usuario que ayudan a que haya un mejor manejo de la información.

Estas carpetas se crean en la unidad C: y son Descargas, Documentos, Música, etc. El que se almacenen en el disco principal hace que de manera común se vaya acumulando ahí todo el rastro de estos archivos.

Es para solucionar esto por lo que se propone que haya una modificación de la ruta de dichas carpetas.

Para cambiar estas carpetas de ubicación debemos realizar lo siguiente:

Cambiar ubicación carpeta usuarios Windows 10

1. Lo primero que haremos será abrir el Explorador de Archivos de Windows 10 y nos iremos a la ruta que queremos como nueva ubicación de la carpeta.
2. Después, nos desplazamos hasta el menú de Inicio y donde pone "Nuevo2, seleccionamos la opción de "nueva carpeta".

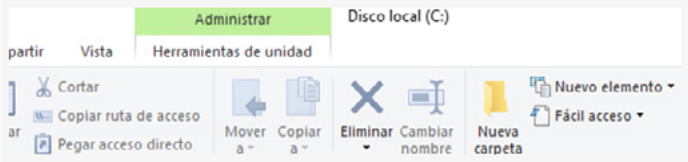


Imagen 38. Explorador de archivos de Windows 10.

También podemos usar la combinación de teclas Ctrl+Shift+N para crear esta carpeta nueva.

3. Ponemos a esta carpeta el nombre que deseamos.
4. Procedemos ahora a buscar en Inicio la opción Ejecutar.
5. Una vez dentro de ella escribimos lo siguiente: %HOMEPATH%.

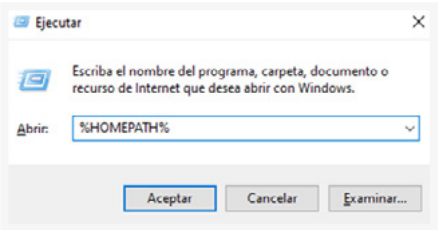


Imagen 39. Ejecutar.

6. Esta acción nos llevará a la carpeta raíz de usuario donde se almacenan todas las que hemos nombrado antes, y una vez aquí haremos click derecho sobre la que deseamos cambiar de ubicación y elegimos "Propiedades"

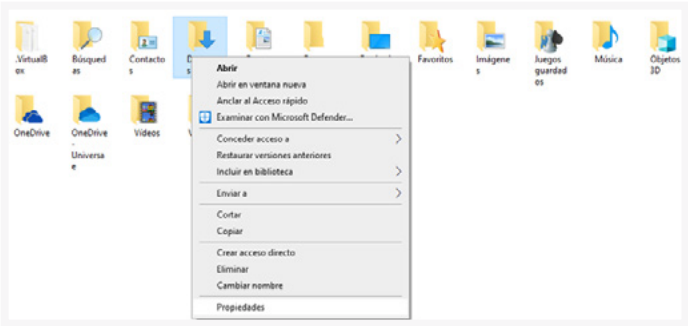


Imagen 40. Seleccionar la carpeta a mover.

7. Se nos abrirá una ventana emergente y tendremos que ir hasta la opción "Ubicación" y seleccionar dentro de esta la opción "mover".

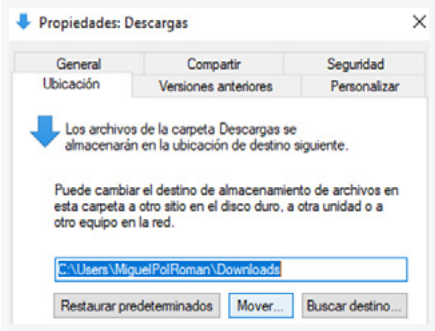


Imagen 41. Cambio de ruta.

8. Se nos abrirá otra ventana en la cuela deberemos dirigirnos a la ruta donde se halla la nueva carpeta que queremos que sea la nueva ubicación de la nuestra.

9. Seleccionamos la carpeta, y damos en Seleccionar carpeta", cuando retrocedamos veremos que se ha cambiado la ruta.

10. Damos en "Aplicar" y nos mostrará un mensaje advirtiéndonos de que se van a mover todos los archivos, habrá que decir que sí.

11. Entonces empezará a gestionar el cambio de todos los documentos de la carpeta y ya tendríamos modificada la ruta de la carpeta en cuestión para futuros objetos.

4.2.3. Seguridad de cuentas de usuario

Para poder administrar la seguridad en las distintas cuentas de usuario, tenemos una herramienta en Windows que se llama Control de cuentas de usuario.

Esta herramienta nos ayuda a que haya una mayor protección frente a distintos malware y ataques de usuarios maliciosos. Un ejemplo que puede hacer UAC (acrónimo de Control de Cuentas de usuario) es que bloquea la instalación automática de aplicaciones que no estén autorizadas por el administrador además de bloquear cambios no autorizados en la configuración de nuestro sistema.

UAC permite que los usuarios se puedan loguear en el equipo como usuarios estándar a no ser que se definan también como administrador. Esta acción impide que durante la sesión se pueda ejecutar nada que necesite de las credenciales del administrador.

Hay ciertas aplicaciones que requerirán de permisos específicos para su ejecución, sobre todo las que no son nativas del sistema operativo en cuestión y son llamadas aplicaciones heredadas.

Para cualquier acción que requiera del permiso expreso del administrador, UAC nos dará la opción de añadir las credenciales de administrador, pero solo para esa acción, teniendo que volver a solicitarlas si necesitamos realizar cualquier otra función de administrador.

Esto nos lleva a plantear que salvo en casos específicos, incluso el administrador debería iniciar sesión en una cuenta estándar y funcionar así para no causar problemas en el equipo, ya que conoce las credenciales para poder realizar cualquier acción.

Establecer la contraseña

Como en prácticamente todos los sistemas, Windows nos da la opción de administrar la seguridad de sus contraseñas, para esto habrá que dirigirnos a:

Inicio → Configuración → Cuentas → Opciones de inicio de sesión

Una vez aquí, tenemos las distintas opciones de configuración de inicio de sesión en el equipo, las cuales podemos cambiar para nosotros y para los demás usuarios.

Habilitar y deshabilitar cuentas de usuario

Como hemos visto antes, las cuentas de usuario pueden deshabilitarse y volver a habilitarse dependiendo de las necesidades que tengamos, pero esta es una tarea que solo pueden ejecutar los administradores o en su defecto los usuarios avanzados con los usuarios que ellos mismos hayan creado.

Deshabilitar una cuenta significa que por mucho que se introduzcan credenciales, no se podrá acceder a la sesión.

Para deshabilitar una cuenta de usuario podemos o realizar por comandos como vimos anteriormente o realizar lo siguiente:

1. Volver a entrar en lusrmgr.msc.
2. Seleccionar el usuario en cuestión
3. Seleccionar “Propiedades”.
4. Una vez dentro de propiedades, seleccionar la opción “La cuenta está deshabilitada”.

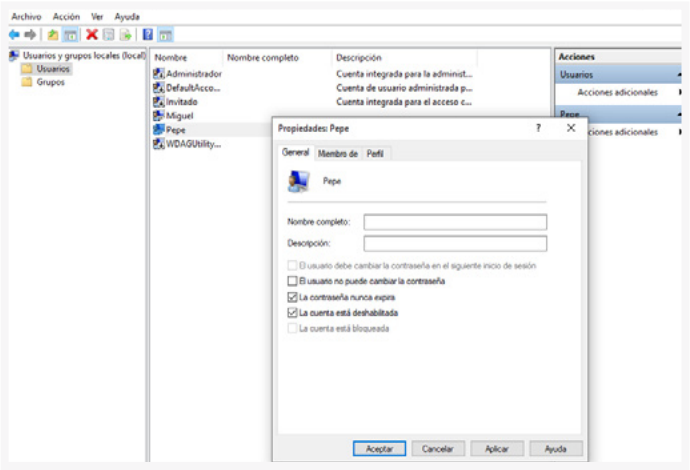


Imagen 42. Deshabilitar una cuenta de usuario.

Cabe destacar que esto tomará cabida para el próximo inicio de sesión y que ara volver a habilitar dicha cuenta solo tendremos que desmarcar esta opción.

4.3. Gestión de procesos por línea de comandos en Linux

Cuando un sistema multiusuario o multitarea trabaja con varios procesos en varios usuarios a la vez, realmente lo que está haciendo es asignar pequeños espacios de tiempo para cada tarea en cada usuario para dar la sensación de que lo aglomera todo de una.

Un propio programa en ejecución genera varias instancias que son conocidas como tareas o procesos y que el sistema operativo administra como si de otro recurso se tratase. Algunos usuarios con ciertos privilegios en Linux podrán hacer uso de comandos para cambiar la ejecución y planificación de los procesos.

4.3.1. Procesos y servicios

Para gestionar todos los procesos, el sistema operativo recurre a ciertas operaciones de creación, comunicación, compartición y finalización de estos. El módulo del sistema que se encarga de esto es el planificador de procesos.

NOTA

Es muy importante que un administrador del sistema conozca la política de planificación de procesos del sistema que administra pues esta hará que unos procesos se ejecuten antes o después y esto puede o bien beneficiar al sistema o perjudicarlo.

Los procesos pasan por distintos estados antes de finalizar y es el planificador de procesos el que se encarga de la gestión de estos y de que se modifique su estado mediante un algoritmo de planificación.

Cuando usamos un sistema operativo multiproceso en el que tenemos una serie de procesos ejecutándose al mismo tiempo en un único procesador es muy probable que muchos de ellos estén bloqueados por falta de recursos o porque simplemente no haya ocurrido la operación necesaria para que el proceso se reactive Los procesos tienen distintos estados que se suelen explicar mediante un diagrama de proceso que muestra cómo van cambiando los procesos según las necesidades que el Sistema operativo impere.

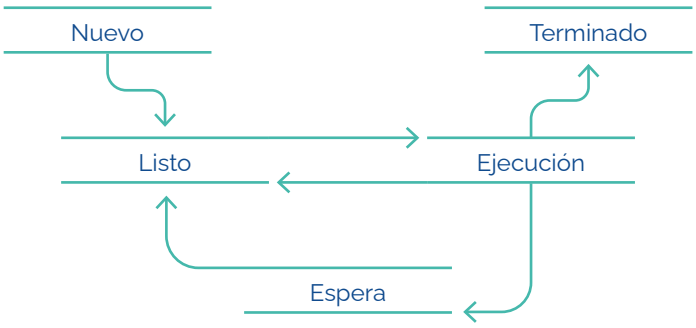


Diagrama 1. Estados de un proceso.

Como se ve en el diagrama anterior, un proceso puede pasar por múltiples estados:

- **Nuevo.** Este estado es el primer estado que puede tener un proceso, y se acaba de crear ya sea porque se ha abierto un programa o porque se ha abierto un fichero. Un proceso nuevo no es válido por sí solo ya que carece de los recursos necesarios.

- **Listo o preparado (ready).** Este estado es cuando el proceso ya está listo, pero a falta de que el procesador comience su ejecución. Cuando un proceso está en este estado puede que haya dos vertientes, o que se ponga en ejecución, o que directamente termine, porque al final no sea necesario su ejecución o porque sea necesaria su finalización para que otro proceso suceda.
- **Ejecución (run).** En estado, el proceso se encuentra trabajando en la CPU y ejecutando instrucciones. En este caso hay tres vertientes: el proceso puede ejecutar todas y cada una de las instrucciones que se le han dado hasta que termine y finalice; también puede que el proceso se bloquee por necesidades del usuario o del propio sistema; por último, puede que el proceso vuelva directamente a estar en preparado porque su tiempo de ejecución se ha excedido y no ha empezado.
- **Espera o bloqueado (wait).** El proceso se encuentra a la espera de que suceda cualquier cambio que lo haga o volver a estar preparado, o entrar en ejecución directamente e incluso finalizar si fuese necesario.
- **Terminado.** Este es el estado en el cual se encuentra el proceso una vez que ha pasado por los demás estados y es cuando desaparece, ya no tiene retorno.

El sistema operativo gestiona los procesos mediante colas, pero cada una es diferente, porque puede haber una por ejemplo para los procesos nuevos y otra para los que están en espera. Es tarea de planificador del procesador el que revise estas colas y decida qué proceso se ejecuta, cuando, como, con que recursos y donde.

Planificación de un proceso

El sistema operativo debe de encargarse de decidir que procesos son los que entran en la CPU y cuales una vez en ella se ejecutan, así como cuando sale el proceso. Todo esto no es simple intuición o suerte, todo esto se realizar debido a que el sistema operativo cuenta con una política de planificación de procesos.

Hay muchísimas maneras de definir las políticas de planificación de procesos, desde por orden de llegada del proceso hasta por orden de prioridad, una prioridad dada por el propio sistema operativo (esta suele ser la más usada). Al final, las políticas de planificación de procesos tienen la función de establecer un orden lógico a la hora de ejecutar procesos para que no haya problemas de rendimiento, funcionamiento, control, seguridad, etc.

Como hemos dicho antes, hay múltiples políticas de planificación, pero ninguna va a ser la perfecta para todos los sistemas operativos. Puede que por ejemplo en un sistema tengamos una política que se basa en la prioridad y va muy bien mientras que en otro sistema esta política cause una caída del rendimiento en el equipo. Esto viene dado por las características que tenga cada proceso porque puede que necesiten de una prioridad muy alta porque son procesos en tiempo real, o que necesiten una carga de CPU altamente importante como puede ser el caso de un proceso de un juego arrancando.

Tenemos diferentes planificadores para nuestro sistema operativo:

- En primera instancia nos encontramos el planificador a largo plazo: este se encarga de que haya un control sobre los procesos múltiples en el sistema, buscando una equivalencia en trabajos de CPU y de E/S. Además, también se encarga de suministrar proceso a la cola de planificación a corto plazo.

- Después tenemos el planificador a medio plazo: este planificador es el encargado de suspender o reactivar procesos que no tienen una mayor importancia e ir pasándolos de la memoria principal a las memorias secundarias o incluso algún dispositivo de almacenamiento externo, Este es el famoso swapping y solo se ejecuta si hay una escasez en los recursos. Como curiosidad, Linux crea una partición dedicada al swapping cuando se instala, consiguiendo un rendimiento óptimo.

- Por último, el planificador a corto plazo: como hemos disco antes, los procesos vienen dados por el planificador a largo plazo y su función es la de asignar y desasignar procesos con la CPU. Vamos a explicarlo, este planificador coge un proceso que se encuentra en estado preparado, y le asigna una CPU para que este se ponga en ejecución, después lo saca y entra el siguiente, así de manera sucesiva. Nos encontramos con dos planificadores a corto plazo:

  - El no expulsivo: en este caso el proceso abandona la CPU directamente al terminar o si lo motiva otro proceso externo.

    → Pero no por fuerza del propio planificador

  - El expulsivo: en este caso los procesos pueden pasar al estado listado por una orden del sistema operativo sin ningún problema, es el más común en los sistemas en tiempo real, que son los más usados.

    → Es el planificador del SO el que lo expulsa sin problemas.

Las principales políticas de planificación de procesos son las siguientes:

- **Primero en llegar, primero en salir (FIFO).** Es similar a una carrera, el proceso que llegue antes a la cola de preparados será el primero en ser planificado y posteriormente ejecutado. Además, es no expulsiva, lo que no la hace una política usable en los sistemas operativos de tiempo compartido como Debian o Windows.

- **Primero el proceso más corto (SJF).** Cada vez que un proceso termine su ejecución en la CPU, se cogerá el siguiente por ser el que menor tiempo de ejecución tenga. Vuelve a ser no expulsiva en su mayoría, aunque hay algunos casos con procesos muy pequeños en los que se usa SRTF, que sí es expulsiva.

- **Prioridades.** Como hemos comentado antes, un proceso tiene una serie de prioridades asignadas ya sea por parte del sistema operativo o porque el propio usuario las haya dividido así. En este caso, todos los procesos están divididos en colas dependiendo de su prioridad y el planificador elegirá los procesos de la primera cola siguiendo a su vez también el método FIFO. Una vez la primera cola se haya acabado pasará a la siguiente así sucesivamente hasta que terminen todas las colas. Esta técnica puede ser tanto expulsiva como no expulsiva, lo que se traduce en que habrá procesos que no se ejecuten debido a los cambios, pero su solución pasará por aumentar de manera progresiva la prioridad.

- **Turno rotatorio (Round-Robin)**. Esta es la mejor técnica a la hora de usar un sistema de tiempo compartido. Su funcionamiento consiste en que cada cierto tiempo se genera una interrupción de reloj periódica y cada proceso cuenta con un cuanto de tiempo máximo al que llamamos quatum. Cuando este tiempo termina el proceso que esté en ejecución pasa a estar listo y se sigue entonces la técnica FIFO para que entre el siguiente.
- **Retroalimentación.** Él último tipo de planificación que vamos a ver es el que consiste en el trabajo de diferentes colas de procesos en estado de listo o preparado, pero con distintas políticas. Esto hace que los procesos vayan pasando por las colas dependiendo del estado u otra política hasta que lleguen a la última cola y terminen.

En cuestión de sistemas operativos los procesos también se pueden dividir o agrupar, Por ejemplo, en Linux, hay tres categorías: interactivos, tiempo real o por lotes.

En Windows, por ejemplo, la planificación de los procesos usa colas múltiples ordenadas por prioridades.

Para MAC OS se usa el mismo modo que en Windows.

4.3.2. Identificación y administración

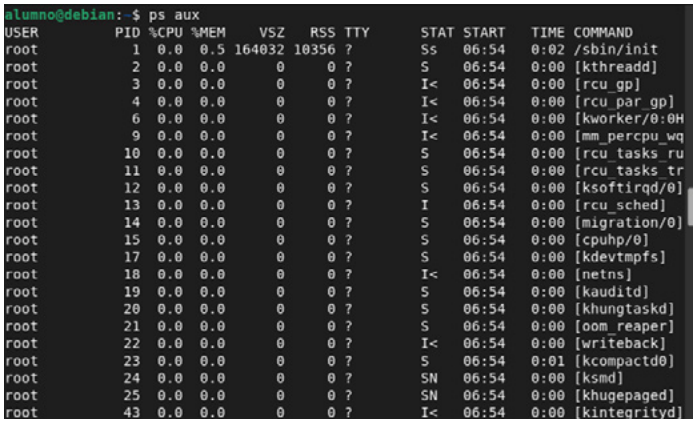
Los procesos se identifican gracias a un identificador único denominado PID, Identificador de proceso. El PCB de cada proceso almacena información acerca de este, principalmente:

- 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 43. 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:

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.

Cuadro 8. Estados de un proceso.

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.

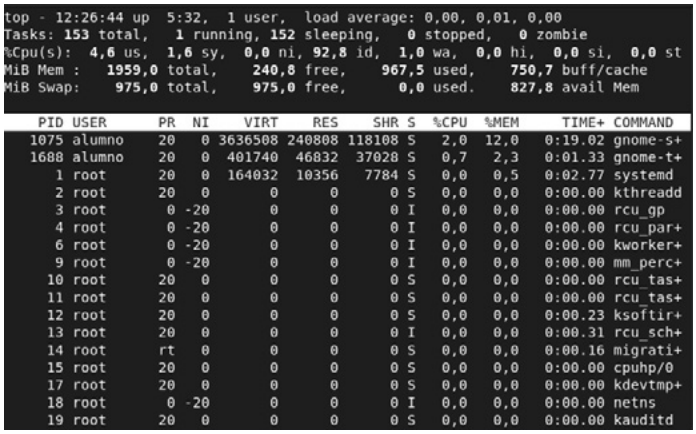


Imagen 44. Comando top.

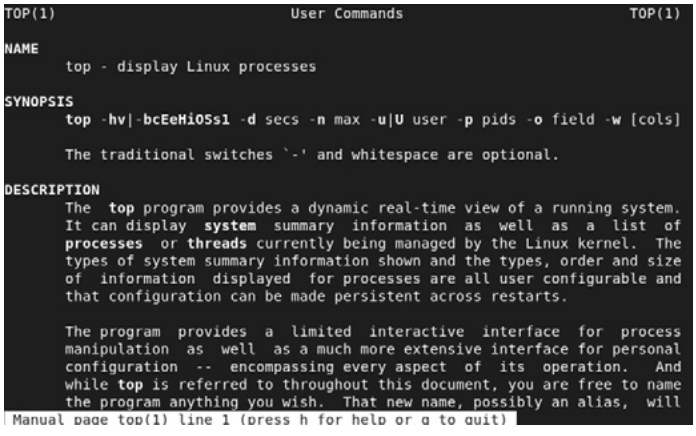


Imagen 45. man top.

NOTA

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

4.4. Gestión de procesos por Interfaz gráfica en Windows

En Windows, para ver los procesos que tenemos en ejecución y que se actualiza de manera constante, recurrimos al 'Administrador de tareas', que contiene una ventana 'Procesos'.

En esta ventana podemos ver todos los procesos en ejecución con distintas agrupaciones, además, en este mismo administrador podemos ver el rendimiento del sistema.

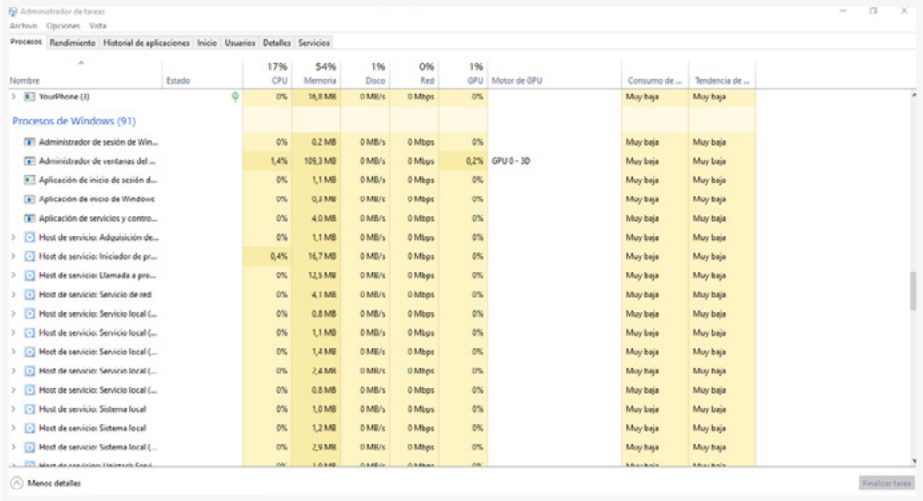


Imagen 46. Administrador de tareas.

4.5. Automatización de tareas en Linux

Los sistemas Linux tienen la suerte de que las tareas se pueden almacenar y planificar a lo largo del tiempo o para que se ejecuten de manera recurrente gracias a cron. La utilidad cron, que cuenta con comandos asociados al terminal, se basa en un demonio que fijándose en ficheros de configuración del sistema preestablecidos ejecuta ciertas acciones sobre el mismo en periodos de tiempo previamente dictaminados.

Hay varios archivos en los que se fija cron, pero el principal es /etc/crontab.

Para poder trabajar con cron debemos de realizar lo siguiente:

1. Revisar si está instalado en nuestro sistema (Algunas distribuciones no lo llevan por defecto, pero se encuentra disponible en todas). Esto se realiza con el siguiente comando:

```
dpkg -l cron
```

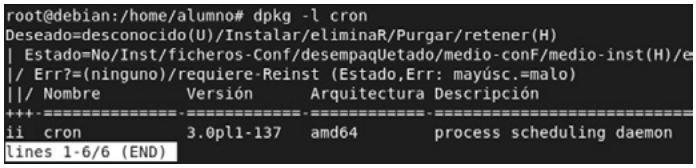


Imagen 47.Comando dpkg -l cron.

2. Si no está instalado, lo instalaremos.
3. Si está instalado, comprobamos que se encuentra activo con el comando:

```
systemctl status cron
```

Imagen 48. Comando systemctl status cron.

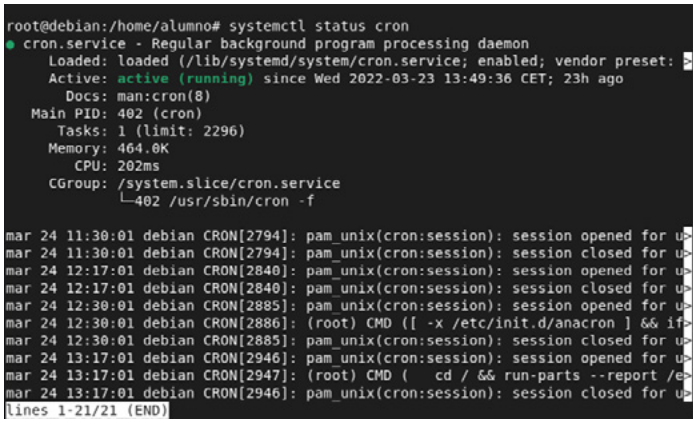


Imagen 49. Salida del comando anterior.

4. Si no está activo, lo activamos con el comando:

```
systemctl start cron
```

5. Si sí que está activo, lo reiniciamos para empezar de cero con el comando:

```
systemctl restart cron
```

6. Ya podemos empezar a trabajar con esta utilidad.

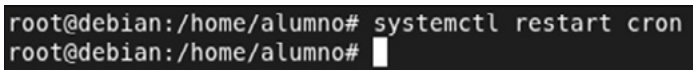


Imagen 50. Comando systemctl restart cron.

El fichero /ect/crontab solo puede ser ejecutado por el supe-rusuario root y cuenta con la siguiente estructura.

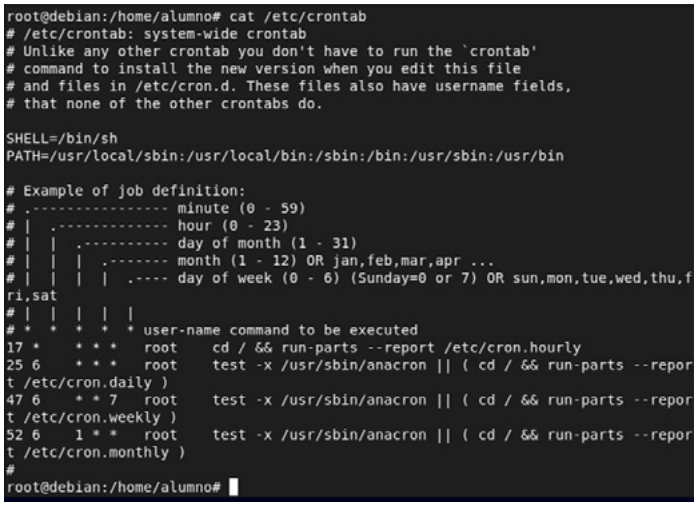


Imagen 51. Fichero /etc/crontab.

Podemos ver que la estructura que se sigue para que se ejecute una orden es la siguiente:

Minutos(0 – 59) Hora(0 – 23) Día del mes(1 – 31) Mes (1 – 12) Día de la semana(0 – 6 | día) usuario comando

Si no se especifica nada, y se pone en su lugar "*", esto quiere decir que recoger todos los valores.

Si queremos editar este fichero, lo mejor es usar el comando crontab -e. Que abrirá un editor que nos permitirá añadir opciones siguiendo la sintaxis de los ficheros.

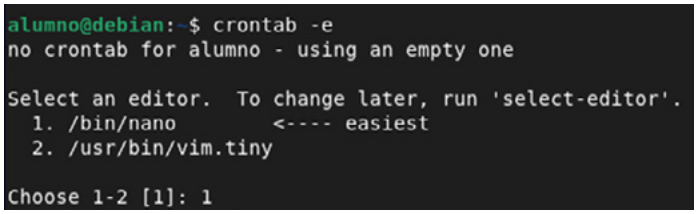


Imagen 52. Comando crontab -e.

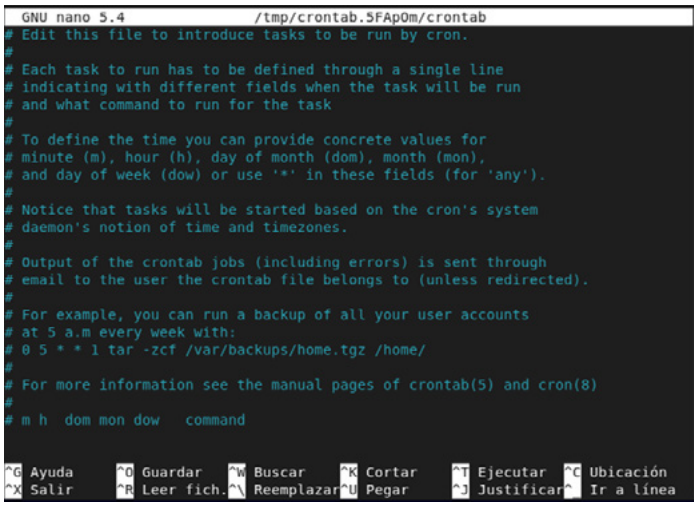


Imagen 53. Editor nano.

Si queremos ver el crontab del usuario en cuestión usaremos el comando `crontab -l`.

```
alumno@debian:~$ crontab -l
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
alumno@debian:~$
```

Imagen 54. Comando crontab -l.

Y si queremos eliminarlo usaremos el comando `crontab -r`.

```
alumno@debian:~$ crontab -r
alumno@debian:~$ crontab -l
no crontab for alumno
alumno@debian:~$
```

Imagen 55. Comando crontab -r.

4.6. Monitorización y gestión del sistema. Evaluación de prestaciones

Cuando se desempeña el trabajo de administrador de sistemas, una de las tareas que se deben de llevar a cabo entre otras es la de gestionar el rendimiento del sistema para saber cuáles son los límites de nuestro sistema y así poder estar preparados de cara a fallos del futuro. Los propios sistemas operativos nos dan una visión sobre el rendimiento del sistema y el máximo de recursos que podemos usar.

Por ejemplo, en Linux, tenemos el comando uptime, que nos permite ver la siguiente información:

```
alumno@debian:~$ uptime
 07:32:34 up  7:52,  1 user,  load average: 0,00, 0,00, 0,00
alumno@debian:~$
```

Imagen 56. Comando uptime.

Para ver información acerca de la memoria RAM tenemos dos comandos disponibles, **free y vmstat**, siendo el segundo, más completo que el primero.

```
alumno@debian:~$ free
               total        used        free      shared  buff/cache   available
Mem:         2005968     1057616      170880       16100      777472      779948
Swap:         998396           0      998396
alumno@debian:~$ vmstat
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 1  0      0 170880  36672 740828    0    0    23     9  593  149  0  0 99  0  0
alumno@debian:~$
```

Imagen 57. Comandos free y vmstat.

Por último, si queremos ver el uso del almacenamiento del sistema en Linux, tenemos que usar el comando: **df**.

```
alumno@debian:~$ df
S.ficheros     bloques de 1K  Usados Disponibles Uso% Montado en
udev                  980020       0      980020   0% /dev
tmpfs                 200600    1136      199464   1% /run
/dev/sda1           50303512 5817328    41898436  13% /
tmpfs                1002984       0     1002984   0% /dev/shm
tmpfs                   5120       4        5116   1% /run/lock
tmpfs                 200596     108      200488   1% /run/user/1000
alumno@debian:~$
```

Imagen 58. Comando df.

Para Microsoft Windows, en cambio, tenemos el Administrador de tareas de nuevo, pero esta vez en la pestaña de 'Rendimiento'.

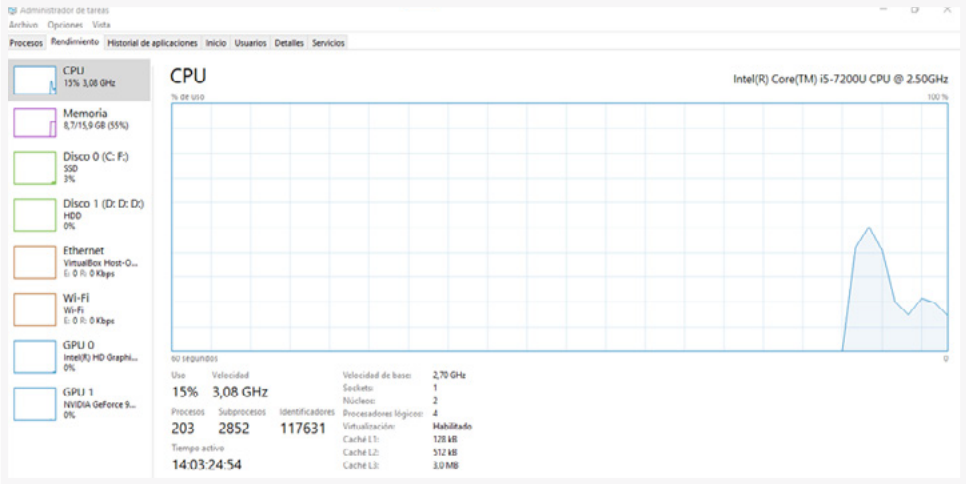


Imagen 59. Pestaña Rendimiento del Administrador de tareas.

Podemos ver que aquí se presentan los recursos que el ordenador tiene y cuál es el consumo actual de ellos.

4.7. Aplicaciones para el mantenimiento y optimización del sistema

Aunque hay aplicaciones nativas del sistema que nos ayudan a un correcto mantenimiento de este y a la optimización de sus recursos. Pero también hay ciertas aplicaciones de terceros que nos dan una mayor seguridad o fiabilidad en algunos casos para que el sistema funcione mejor. En conjunto las aplicaciones para estos requisitos se pueden agrupar en:

- **Aplicaciones de actualización y control de drivers.** Los sistemas operativos llevan en sus propias actualizaciones, una serie de actualizaciones con referencia a los drivers que hacen que estos consten de mejoras o porque han cambiado su firmware y un sinfín de opciones. No obstante, hay casos, como los de un procesador Intel, que sufren actualizaciones muy continuas y los sistemas no las tienen disponibles aún. En el caso de estos procesadores tenemos por ejemplo Intel Driver and Support Assistant que comprueba los mejores controladores y los descarga. Los equipos Lenovo cuentan por ejemplo también con Lenovo Vantage que mantiene todo el sistema actualizado.
- **Aplicaciones para sincronización, copias de seguridad e imágenes del sistema.** Volvemos a lo mismo de antes, el mismo sistema puede crear imágenes y copias de seguridad de sí mismo, pero esto sí que es recomendable que se haga con una aplicación de un tercer para que no haya errores. Aunque existen multitud de estas aplicaciones, una muy conocida es Bacula referente a los sistemas Linux o VeemBakcup, que es multiplataforma y además es software propietario.
- **Antivirus.** Como ya sabemos, este tipo de software es esencial en un equipo si queremos que no se infecte por ningún tipo de malware y que nuestra información no esté en peligro. Los más sonados son Windows Defender, nativo de Windows, y otro ejemplo es Checkpoint, de PaloAlto.

+34 919 033 434 info@universae.com