

Usuarios, Servicios y Permisos

1) Bloque 1: Comandos de Información y Preparación

- 1) Identidad del Usuario: Abre una terminal y ejecuta un comando para saber qué usuario eres y a qué grupos perteneces.

```
servidor@servidor:~$ whoami
servidor
servidor@servidor:~$ groups
servidor adm cdrom sudo dip plugdev lxd
```

- 2) Usuarios Conectados: Muestra quién está conectado actualmente al sistema. Luego, ejecuta otro comando que te dé información más detallada, como el tiempo que llevan conectados y qué están ejecutando.

```
servidor@servidor:~$ who
servidor pts/0      2025-10-23 06:45 (10.140.42.214)
servidor@servidor:~$ w servidor
 06:49:05 up 4 min,  1 user,  load average: 0.01, 0.08, 0.04
USER          TTY          FROM          LOGIN@      IDLE        JCPU   PCPU   WHAT
servidor      pts/0      10.140.42.214  06:45      4:21       0.00s   0.03s  sshd: servidor [priv]
```

- 3) Historial de Conexiones: Lista los últimos inicios de sesión en el sistema.

```
servidor@servidor:~$ last
servidor pts/0      10.140.42.214      Thu Oct 23 06:45    still logged in
reboot    system boot      6.8.0-86-generic   Thu Oct 23 06:44    still running
servidor pts/0      10.140.42.214      Thu Oct 23 06:43 - 06:44    (00:00)
reboot    system boot      6.8.0-86-generic   Thu Oct 23 06:42 - 06:44    (00:01)
reboot    system boot      6.8.0-86-generic   Thu Oct 23 06:39 - 06:42    (00:03)
servidor pts/0      10.140.42.214      Thu Oct 23 06:32 - 06:38    (00:06)
reboot    system boot      6.8.0-85-generic   Thu Oct 23 06:31 - 06:38    (00:07)
reboot    system boot      6.8.0-85-generic   Wed Oct 22 10:13 - 10:24    (00:11)
reboot    system boot      6.8.0-85-generic   Wed Oct 22 10:02 - 10:04    (00:01)
servidor pts/0      10.140.42.214      Mon Oct 20 06:52 - crash (2+03:09)
reboot    system boot      6.8.0-85-generic   Mon Oct 20 06:49 - 10:04    (2+03:14)
reboot    system boot      6.8.0-85-generic   Mon Oct 20 06:44 - 06:47    (00:03)
reboot    system boot      6.8.0-85-generic   Fri Oct 17 06:36 - 07:24    (00:48)
reboot    system boot      6.8.0-85-generic   Thu Oct 16 08:00 - 10:24    (02:23)
reboot    system boot      6.8.0-85-generic   Thu Oct 16 07:39 - 07:39    (00:00)
reboot    system boot      6.8.0-85-generic   Thu Oct 16 07:35 - 07:38    (00:02)
reboot    system boot      6.8.0-85-generic   Thu Oct 16 07:34 - 07:38    (00:03)
reboot    system boot      6.8.0-85-generic   Thu Oct 16 07:33 - 07:33    (00:00)
reboot    system boot      6.8.0-85-generic   Thu Oct 16 07:29 - 07:32    (00:02)
reboot    system boot      6.8.0-85-generic   Thu Oct 16 06:57 - 07:29    (00:32)
reboot    system boot      6.8.0-85-generic   Thu Oct 16 06:29 - 06:57    (00:27)
reboot    system boot      6.8.0-85-generic   Wed Oct 15 09:36 - 10:23    (00:46)
reboot    system boot      6.8.0-85-generic   Wed Oct 15 09:28 - 09:30    (00:01)
reboot    system boot      6.8.0-85-generic   Wed Oct 15 09:27 - 09:28    (00:00)
reboot    system boot      6.8.0-85-generic   Wed Oct 15 09:26 - 09:27    (00:00)
reboot    system boot      6.8.0-85-generic   Wed Oct 15 09:19 - 09:26    (00:06)

wtmp begins Wed Oct 15 09:19:06 2025
```

- 4) Crear Entorno de Trabajo: En tu directorio personal (/home/tu_usuario), crea una carpeta principal para todos los ejercicios llamada practicas_linux.

```
servidor@servidor:~$ mkdir practicas_linux
servidor@servidor:~$ ll
total 1104
drwxr-x--- 8 servidor servidor 4096 Oct 23 06:50 ./
drwxr-xr-x 3 root      root      4096 Oct 16 09:02 ../
-rw-rw-r-- 1 servidor servidor    0 Oct 16 10:19 a.txt
-rw----- 1 servidor servidor 4733 Oct 23 06:44 .bash_history
-rw-r--r-- 1 servidor servidor  220 Mar 31 2024 .bash_logout
-rw-r--r-- 1 servidor servidor 3771 Mar 31 2024 .bashrc
-rw-rw-r-- 1 servidor servidor    0 Oct 16 10:19 b.log
drwx----- 2 servidor servidor 4096 Oct 15 09:20 .cache/
-rwxrwxr-x 1 servidor servidor   74 Oct 20 07:14 cambia_extension.sh*
-rw-rw-r-- 1 servidor servidor    0 Oct 16 10:19 c.jpg
-rwxrwxr-x 1 servidor servidor   89 Oct 20 07:09 compara_numeros.sh*
-rw-rw-r-- 1 servidor servidor 5026 Oct 17 07:12 config_files.txt
-rwxrwxr-x 1 servidor servidor  147 Oct 20 08:06 directorio_fallador.sh*
drwxrwxr-x 2 servidor servidor 4096 Oct 20 07:18 documentos/
-rw-rw-r-- 1 servidor servidor    0 Oct 16 08:14 'dos palabras.txt'
-rw-rw-r-- 1 servidor servidor   11 Oct 17 07:09 ejemplo_redireccion.txt
lrwxrwxrwx 1 servidor servidor   26 Oct 20 07:52 enlace -> /home/servidor
-rw-rw-r-- 1 servidor servidor  249 Oct 17 07:12 errors.txt
-rw-rw-r-- 1 servidor servidor 511989 Oct 20 07:54 home.tar.gz
-rwx----- 1 servidor servidor  169 Oct 20 07:04 imprime_nombre.sh*
-rwxrwxr-x 1 servidor servidor  150 Oct 20 08:02 lee_archivo.sh*
-rw----- 1 servidor servidor   66 Oct 20 07:38 .lessht
-rw-rw-r-- 1 servidor servidor  436 Oct 16 10:19 letras.zip
drwxrwxr-x 3 servidor servidor 4096 Oct 16 06:58 .local/
drwxrwxr-x 2 servidor servidor 4096 Oct 16 08:17 mi_bin/
-rw-rw-r-- 1 servidor servidor  453 Oct 17 06:38 mis_archivos.txt
-rwxrwxr-x 1 servidor servidor  124 Oct 20 08:09 numeros_grandes.sh*
-rw-rw-r-- 1 servidor servidor 2987 Oct 20 07:28 october_files.txt
drwxrwxr-x 2 servidor servidor 4096 Oct 23 06:50 practicas_linux/
```

- 5) Estructura de Directorios: Dentro de practicas_linux, crea la siguiente estructura de directorios: proyectos, documentos y scripts.

```
servidor@servidor:~$ cd practicas_linux/
servidor@servidor:~/practicas_linux$ mkdir proyectos documentos scripts
servidor@servidor:~/practicas_linux$ ll
total 20
drwxrwxr-x 5 servidor servidor 4096 Oct 23 06:52 ./
drwxr-x--- 8 servidor servidor 4096 Oct 23 06:50 ../
drwxrwxr-x 2 servidor servidor 4096 Oct 23 06:52 documentos/
drwxrwxr-x 2 servidor servidor 4096 Oct 23 06:52 proyectos/
drwxrwxr-x 2 servidor servidor 4096 Oct 23 06:52 scripts/
```

2) Bloque 2: Gestión de Usuarios y Grupos

- 1) Crear Grupos: Crea tres nuevos grupos en el sistema: desarrolladores, analistas y becarios.

```
servidor@servidor:~$ sudo groupadd desarrolladores
[sudo] password for servidor:
servidor@servidor:~$ sudo groupadd analistas
servidor@servidor:~$ sudo groupadd becarios
```

- 2) Verificar Grupos: Confirma que los grupos se han creado correctamente buscando sus nombres en el archivo /etc/group.

```
servidor@servidor:~$ tail /etc/group
rdma:x:106:
tcpdump:x:107:
tss:x:108:
landscape:x:109:
fwupd-refresh:x:989:
servidor:x:1000:
plocate:x:110:
desarrolladores:x:1001:
analistas:x:1002:
becarios:x:1003:
```

- 3) Crear un Usuario Básico: Crea un nuevo usuario llamado juan.

```
servidor@servidor:~$ sudo useradd juan
```

- 4) Crear Usuario con Grupo Primario: Crea una usuaria llamada ana y asígnala directamente al grupo primario desarrolladores.

```
servidor@servidor:~$ sudo useradd -g desarrolladores ana
```

- 5) Crear Usuario Completo: Crea un usuario david asignándolo al grupo primario analistas y, a la vez, como miembro de los grupos secundarios desarrolladores y becarios.

```
servidor@servidor:~$ sudo useradd -g analistas -G desarrolladores,becarios david
```

- 6) Establecer Contraseñas: Asigna una contraseña a los usuarios juan, ana y david.

```
servidor@servidor:~$ sudo passwd juan
New password:
Retype new password:
passwd: password updated successfully
servidor@servidor:~$ sudo passwd ana
New password:
Retype new password:
passwd: password updated successfully
servidor@servidor:~$ sudo passwd david
New password:
Retype new password:
passwd: password updated successfully
```

- 7) Verificar Usuarios: Comprueba que los tres nuevos usuarios existen en el sistema, inspeccionando el final del archivo /etc/passwd.

```
servidor@servidor:~$ sudo tail /etc/passwd
tcpdump:x:105:107::/nonexistent:/usr/sbin/nologin
tss:x:106:108:TPM software stack,,,:/var/lib/tpm:/bin/false
landscape:x:107:109::/var/lib/landscape:/usr/sbin/nologin
fwupd-refresh:x:989:989:Firmware update daemon:/var/lib/fwupd:/usr/sbin/nologin
usbmux:x:108:46:usbmux daemon,,,:/var/lib/usbmux:/usr/sbin/nologin
sshd:x:109:65534::/run/sshd:/usr/sbin/nologin
servidor:x:1000:1000:Pablo:/home/servidor:/bin/bash
juan:x:1001:1004::/home/juan:/bin/sh
ana:x:1002:1001::/home/ana:/bin/sh
david:x:1003:1002::/home/david:/bin/sh
```

- 8) Cambiar de Usuario: Conviértete en el usuario juan usando el comando su. Una vez dentro de su sesión, comprueba quién eres y en qué directorio te encuentras. Vuelve a tu sesión de usuario original.

```
servidor@servidor:~$ su - juan
Password:
su: warning: cannot change directory to /home/juan: No such file or directory
$ whoami
juan
$ pwd
/home/servidor
$ exit
```

- 9) Modificar Grupos de un Usuario: Modifica al usuario juan para que su grupo primario sea becarios y añádelo también al grupo secundario analistas.

```
servidor@servidor:~$ sudo usermod -g becarios juan
servidor@servidor:~$ sudo usermod -a -G analistas juan
```

- 10) Verificar Modificación: Comprueba que los cambios del usuario juan se han aplicado correctamente.

```
servidor@servidor:~$ id juan
uid=1001(juan) gid=1003(becarios) groups=1003(becarios),1002(analistas)
```

- 11) Bloquear una Cuenta: Bloquea la cuenta del usuario juan para que no pueda iniciar sesión.

```
servidor@servidor:~$ sudo passwd -l juan
passwd: password changed.
```

- 12) Intentar Cambiar a Usuario Bloqueado: Intenta convertirte en el usuario juan de nuevo. Debería fallar.

```
servidor@servidor:~$ su juan
Password:
su: Authentication failure
```

- 13) Desbloquear una Cuenta: Desbloquea la cuenta del usuario juan.

```
servidor@servidor:~$ sudo passwd -u juan
passwd: password changed.
```

- 14) Eliminar un Grupo: Elimina el grupo becarios. ¿Qué ocurre? (Nota: Fallará si algún usuario lo tiene como grupo primario).

```
servidor@servidor:~$ sudo groupdel becarios
groupdel: cannot remove the primary group of user 'juan'
```

- 15) Eliminar Usuario y su Directorio: Elimina al usuario juan y asegúrate de que su directorio personal (/home/juan) también se borre.

```
servidor@servidor:~$ sudo userdel -r juan
userdel: group juan not removed because it is not the primary group of user juan.
userdel: juan mail spool (/var/mail/juan) not found
userdel: juan home directory (/home/juan) not found
```

- 16) Crear Archivos de Prueba: Dentro de la carpeta proyectos, crea un archivo vacío llamado informe.txt. Dentro de scripts, crea otro archivo vacío llamado lanzar_app.sh.

```
servidor@servidor:~$ touch ~/practicas_linux/proyectos/informe.txt
servidor@servidor:~$ touch ~/practicas_linux/scripts/lanzar_app.sh
```

- 17) Ver Permisos: Muestra los permisos por defecto de los archivos y directorios que has creado. Anota quién es el propietario y el grupo.

Tanto el propietario como el grupo es “servidor”.

```
servidor@servidor:~$ ls -l ~/practicas_linux/proyectos/informe.txt
-rw-rw-r-- 1 servidor servidor 0 Oct 23 07:19 /home/servidor/practicas_linux/proyectos/informe.txt
servidor@servidor:~$ ls -l ~/practicas_linux/scripts/lanzar_app.sh
-rw-rw-r-- 1 servidor servidor 0 Oct 23 07:20 /home/servidor/practicas_linux/scripts/lanzar_app.sh
```

- 18) Cambiar Propietario: Cambia el propietario del archivo informe.txt para que pertenezca a la usuaria Ana.

```
servidor@servidor:~$ sudo chown ana ~/practicas_linux/proyectos/informe.txt
servidor@servidor:~$ ls -l ~/practicas_linux/proyectos/informe.txt
-rw-rw-r-- 1 ana servidor 0 Oct 23 07:19 /home/servidor/practicas_linux/proyectos/informe.txt
```

3) Bloque 3: Permisos y Propiedad de Archivos

- 1) Cambiar Grupo: Cambia el grupo del directorio proyectos para que pertenezca al grupo desarrolladores.

```
servidor@servidor:~$ sudo chown :desarrolladores ~/practicas_linux/proyectos
servidor@servidor:~$ ll practicas_linux/
total 20
drwxrwxr-x 5 servidor servidor 4096 Oct 23 06:52 ./
drwxr-x--- 8 servidor servidor 4096 Oct 23 07:29 ../
drwxrwxr-x 2 servidor servidor 4096 Oct 23 06:52 documentos/
drwxrwxr-x 2 servidor desarrolladores 4096 Oct 23 07:19 proyectos/
drwxrwxr-x 2 servidor servidor 4096 Oct 23 07:20 scripts/
```

- 2) Cambiar Propietario y Grupo: Cambia el propietario y el grupo del archivo lanzar_app.sh para que pertenezcan al usuario david y al grupo analistas, respectivamente, con un solo comando.

```
servidor@servidor:~$ sudo chown david:analistas ~/practicas_linux/scripts/lanzar_app.sh
servidor@servidor:~$ ll practicas_linux/scripts/lanzar_app.sh
-rw-rw-r-- 1 david analistas 0 Oct 23 07:20 practicas_linux/scripts/lanzar_app.sh
```

- 3) Permisos con Notación Octal (Archivo): Usa la notación numérica (octal) para asignar los siguientes permisos a informe.txt: el propietario (ana) puede leer y escribir; el grupo (desarrolladores) solo puede leer; y los otros no tienen ningún permiso.

```
servidor@servidor:~$ sudo chmod 640 ~/practicas_linux/proyectos/informe.txt
servidor@servidor:~$ ll ~/practicas_linux/proyectos/informe.txt
-rw-r----- 1 ana desarrolladores 0 Oct 23 07:19 /home/servidor/practicas_linux/proyectos/informe.txt
```

- 4) Permisos con Notación Octal (Directorio): Asigna permisos de lectura, escritura y ejecución para el propietario y solo de lectura y ejecución para los miembros del grupo al directorio documentos.

```
servidor@servidor:~$ sudo chmod 750 ~/practicas_linux/documentos
```

- 5) Verificar Permisos: Lista el contenido de practicas_linux para verificar que todos los cambios de propietario y permisos se han aplicado correctamente.

```
servidor@servidor:~$ ll practicas_linux/
total 20
drwxrwxr-x 5 servidor servidor 4096 Oct 23 06:52 ./
drwxr-x--- 8 servidor servidor 4096 Oct 23 07:29 ../
drwxr-x--- 2 servidor servidor 4096 Oct 23 06:52 documentos/
drwxrwxr-x 2 servidor desarrolladores 4096 Oct 23 07:19 proyectos/
drwxrwxr-x 2 servidor servidor 4096 Oct 23 07:20 scripts/
```

- 6) Permisos con Notación Simbólica (Añadir): Usa la notación simbólica para añadir el permiso de ejecución al propietario del script lanzar_app.sh.

```
servidor@servidor:~$ sudo chmod u+x ~/practicas_linux/scripts/lanzar_app.sh
servidor@servidor:~$ ll ~/practicas_linux/scripts/lanzar_app.sh
-rwxrw-r-- 1 david analistas 0 Oct 23 07:20 /home/servidor/practicas_linux/scripts/lanzar_app.sh*
```

- 7) Permisos con Notación Simbólica (Quitar): Quita el permiso de lectura al “resto del mundo” (otros) en el directorio proyectos.

```
servidor@servidor:~$ sudo chmod o-r ~/practicas_linux/proyectos
servidor@servidor:~$ ll practicas_linux/
total 20
drwxrwxr-x 5 servidor servidor 4096 Oct 23 06:52 ./
drwxr-x--- 8 servidor servidor 4096 Oct 23 07:29 ../
drwxr-x--- 2 servidor servidor 4096 Oct 23 06:52 documentos/
drwxrwx--x 2 servidor desarrolladores 4096 Oct 23 07:19 proyectos/
drwxrwxr-x 2 servidor servidor 4096 Oct 23 07:20 scripts/
```

- 8) Permisos Recursivos: Dentro de proyectos, crea una nueva carpeta version2 con un archivo notas.txt dentro. Luego, cambia el propietario de la carpeta proyectos y todo su contenido para que pertenezca a david con un solo comando recursivo.

```

servidor@servidor:~$ mkdir ~/practicas_linux/proyectos/version2
servidor@servidor:~$ touch ~/practicas_linux/proyectos/version2/notas.txt
servidor@servidor:~$ sudo chown -R david ~/practicas_linux/proyectos
servidor@servidor:~$ ll practicas_linux/
total 20
drwxrwxr-x 5 servidor servidor 4096 Oct 23 06:52 ./
drwxr-x--- 8 servidor servidor 4096 Oct 23 07:29 ../
drwxr-x--- 2 servidor servidor 4096 Oct 23 06:52 documentos/
drwxrwx--x 3 david desarrolladores 4096 Oct 23 07:48 proyectos/
drwxrwxr-x 2 servidor servidor 4096 Oct 23 07:20 scripts/

```

- 9) Permiso Especial SGID en Directorio: Establece el permiso especial SGID en el directorio documentos. Después, cambia a ser el usuario david (su david) y crea un nuevo archivo dentro de documentos. Verifica a qué grupo pertenece el nuevo archivo (debería heredar el del directorio documentos). Vuelve a tu usuario.

```

servidor@servidor:~/practicas_linux/proyectos$ su david
Password:
$ touch archivo_nuevo.txt
$ ls -l
total 4
-rw-r--r-- 1 david analistas 0 Oct 23 08:05 archivo_nuevo.txt
-rw-r----- 1 david desarrolladores 0 Oct 23 07:19 informe.txt
drwxrwxr-x 2 david servidor 4096 Oct 23 07:48 version2

```

- 10) Permiso Especial SUID: Establece el permiso SUID en el script lanzar_app.sh. (Nota: Explica a tus alumnos qué implicaría esto si fuera un programa compilado).

```

servidor@servidor:~/practicas_linux$ sudo chmod u+s ~/practicas_linux/scripts/lanzar_app.sh
servidor@servidor:~/practicas_linux$ ll scripts/
total 8
drwxrwxr-x 2 servidor servidor 4096 Oct 23 07:20 ./
drwxrwxrwx 5 servidor servidor 4096 Oct 23 06:52 ../
-rwsr--r-- 1 david analistas 0 Oct 23 07:20 lanzar_app.sh*

```

Hola mis alumnos, cuando un programa o script con permiso SUID se ejecuta, el proceso corre con los privilegios del propietario del archivo, no con los del usuario que lo ejecuta.

- 11) Comprobar umask: Muestra el valor umask actual de tu sesión.

```

servidor@servidor:~/practicas_linux$ umask
0002

```

- 12) Efecto de umask: Cambia temporalmente tu umask a 077. Crea un nuevo archivo llamado privado.txt. Comprueba sus permisos por defecto. Luego, restaura el umask a su valor original.

```

servidor@servidor:~/practicas_linux$ umask 077
servidor@servidor:~/practicas_linux$ touch privado.txt
servidor@servidor:~/practicas_linux$ ll privado.txt
-rw----- 1 servidor servidor 0 Oct 23 08:15 privado.txt
servidor@servidor:~/practicas_linux$ umask 002

```

- 13) Estado Detallado de un Servicio: Comprueba el estado completo del servicio cups.

El servicio cups no existe en mi máquina, lo he sustituido por SSH.

```
servidor@servidor:~/practicass_linux$ systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/usr/lib/systemd/system/ssh.service; enabled; preset: enabled)
   Active: active (running) since Thu 2025-10-23 06:53:44 UTC; 1h 25min ago
   TriggeredBy: ● ssh.socket
     Docs: man:sshd(8)
           man:sshd_config(5)
   Process: 726 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 735 (sshd)
     Tasks: 1 (limit: 2265)
    Memory: 5.0M (peak: 21.3M)
       CPU: 213ms
    CGroup: /system.slice/ssh.service
            └─735 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Oct 23 06:53:43 servidor systemd[1]: Starting ssh.service - OpenBSD Secure Shell server...
Oct 23 06:53:44 servidor sshd[735]: Server listening on 0.0.0.0 port 22.
Oct 23 06:53:44 servidor sshd[735]: Server listening on :: port 22.
Oct 23 06:53:44 servidor systemd[1]: Started ssh.service - OpenBSD Secure Shell server.
Oct 23 06:54:05 servidor sshd[822]: Accepted password for servidor from 10.140.42.214 port 62455 ssh2
Oct 23 06:54:05 servidor sshd[822]: pam_unix(sshd:session): session opened for user servidor(uid=1000) by servidor(ui>
Oct 23 08:03:24 servidor sshd[1582]: Accepted password for david from 10.140.42.214 port 50063 ssh2
Oct 23 08:03:24 servidor sshd[1582]: pam_unix(sshd:session): session opened for user david(uid=1003) by david(uid=0)
Oct 23 08:04:25 servidor sshd[1719]: Accepted password for servidor from 10.140.42.214 port 52300 ssh2
Oct 23 08:04:25 servidor sshd[1719]: pam_unix(sshd:session): session opened for user servidor(uid=1000) by servidor(ui>
```

- 14) Analiza la salida: ¿está activo (active), cargado (loaded) y habilitado (enabled)? Anota las últimas líneas de su registro (log) que aparecen.

Está activo (active). Las ultimas líneas son:

Oct 23 06:53:43 servidor systemd[1]: Starting ssh.service - OpenBSD Secure Shell server...

Oct 23 06:53:44 servidor sshd[735]: Server listening on 0.0.0.0 port 22.

Oct 23 06:53:44 servidor sshd[735]: Server listening on :: port 22.

Oct 23 06:53:44 servidor systemd[1]: Started ssh.service - OpenBSD Secure Shell server.

Oct 23 06:54:05 servidor sshd[822]: Accepted password for servidor from 10.140.42.214 port 62455 ssh2

Oct 23 06:54:05 servidor sshd[822]: pam_unix(sshd:session): session opened for user servidor(uid=1000) by servidor(ui>

Oct 23 08:03:24 servidor sshd[1582]: Accepted password for david from 10.140.42.214 port 50063 ssh2

Oct 23 08:03:24 servidor sshd[1582]: pam_unix(sshd:session): session opened for user david(uid=1003) by david(uid=0)

Oct 23 08:04:25 servidor sshd[1719]: Accepted password for servidor from 10.140.42.214 port 52300 ssh2

Oct 23 08:04:25 servidor sshd[1719]: pam_unix(sshd:session): session opened for user servidor(uid=1000) by servidor(u>

- 15) Comprobación Rápida: Utiliza un comando más directo para verificar si el servicio cups está actualmente en ejecución (activo). La salida de este comando debería ser simplemente active o inactive.

```
servidor@servidor:~/practicass_linux$ systemctl is-active cups
inactive
```

- 16) Ver Archivo de Unidad: Muestra el contenido del archivo de unidad del servicio cups (cups.service). Esto te permitirá ver cómo está definido el servicio.

```
servidor@servidor:~/practicass_linux$ systemctl cat ssh.service
# /usr/lib/systemd/system/ssh.service
[Unit]
Description=OpenBSD Secure Shell server
Documentation=man:sshd(8) man:sshd_config(5)
After=network.target auditd.service
ConditionPathExists=!/etc/ssh/ssh_not_to_be_run

[Service]
EnvironmentFile=-/etc/default/ssh
ExecStartPre=/usr/sbin/sshd -t
ExecStart=/usr/sbin/sshd -D $SSHD_OPTS
ExecReload=/usr/sbin/sshd -t
ExecReload=/bin/kill -HUP $MAINPID
KillMode=process
Restart=on-failure
RestartPreventExitStatus=255
Type=notify
RuntimeDirectory=sshd
RuntimeDirectoryMode=0755

[Install]
WantedBy=multi-user.target
Alias=sshd.service
```

4) Bloque 4: Gestión de Servicios con systemctl

- 1) Detener un Servicio: Detén la ejecución del servicio cups. Comprueba su estado de nuevo para confirmar que está inactive (dead).

Vamos a utilizar cron en vez de cups.

```
servidor@servidor:~$ sudo systemctl stop cron
```

- 2) Iniciar un Servicio: Vuelve a iniciar el servicio cups. Verifica una vez más que ha vuelto al estado active (running).

```
servidor@servidor:~$ sudo systemctl start cron
servidor@servidor:~$ sudo systemctl is-active cron
active
```

- 3) Reiniciar un Servicio: El comando restart es muy común tras un cambio de configuración. Ejecútalo para el servicio cups.

```
servidor@servidor:~$ sudo systemctl restart cron
servidor@servidor:~$ sudo systemctl is-active cron
active
```

- 4) Habilitar para el Arranque: Asegúrate de que el servicio cups esté configurado para iniciarse automáticamente cada vez que el sistema arranque.

```
servidor@servidor:~$ sudo systemctl enable cron
Synchronizing state of cron.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable cron
servidor@servidor:~$ sudo systemctl state cron
Unknown command verb 'state', did you mean 'start'?
servidor@servidor:~$ sudo systemctl status cron
● cron.service - Regular background program processing daemon
   Loaded: loaded (/usr/lib/systemd/system/cron.service; enabled; preset: enabled)
   Active: active (running) since Thu 2025-10-23 08:56:40 UTC; 1min 46s ago
     Docs: man:cron(8)
    Main PID: 2170 (cron)
      Tasks: 1 (limit: 2265)
    Memory: 344.0K (peak: 604.0K)
       CPU: 5ms
    CGroup: /system.slice/cron.service
            └─2170 /usr/sbin/cron -f -P

Oct 23 08:56:40 servidor systemd[1]: Started cron.service - Regular background program processing daemon
Oct 23 08:56:40 servidor (cron)[2170]: cron.service: Referenced but unset environment variable evaluates
Oct 23 08:56:40 servidor cron[2170]: (CRON) INFO (pidfile fd = 3)
Oct 23 08:56:40 servidor cron[2170]: (CRON) INFO (Skipping @reboot jobs -- not system startup)
```

- 5) Verificar si está Habilitado: Usa un comando específico para preguntar si cups está habilitado. La salida debería ser enabled o disabled.

```
servidor@servidor:~$ systemctl is-enabled cron
enabled
```

- 6) Deshabilitar para el Arranque: Ahora, desactiva el servicio cups para que no se inicie automáticamente.

```
servidor@servidor:~$ sudo systemctl disable cron
Synchronizing state of cron.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install disable cron
Removed "/etc/systemd/system/multi-user.target.wants/cron.service".
servidor@servidor:~$ systemctl is-enabled cron
disabled
```

- 7) Enmascarar un Servicio: El enmascaramiento es una forma más contundente de deshabilitar, ya que impide cualquier tipo de inicio (manual o automático). Enmascara el servicio cups. Intenta iniciarlo después. Debería fallar. No olvides desenmascararlo (unmask) al terminar el ejercicio.

```
servidor@servidor:~$ sudo systemctl mask cron
Created symlink /etc/systemd/system/cron.service → /dev/null.
servidor@servidor:~$ sudo systemctl start cron
Failed to start cron.service: Unit cron.service is masked.
servidor@servidor:~$ sudo systemctl unmask cron
Removed "/etc/systemd/system/cron.service".
servidor@servidor:~$ sudo systemctl start cron
```

- 8) Comprobar Estado y Activar UFW: Primero, ejecuta un comando para verificar el estado actual del firewall. Probablemente estará inactivo. A continuación, activa UFW. Presta atención al mensaje de advertencia, especialmente si estás conectado por SSH.

```
servidor@servidor:~$ sudo ufw status
Status: inactive
servidor@servidor:~$ sudo ufw enable
Command may disrupt existing ssh connections. Proceed with operation (y|n)? y
Firewall is active and enabled on system startup
```

- 9) Permitir un Servicio Web (HTTP): Imagina que tu servidor necesita alojar una página web. Añade una regla para permitir todas las conexiones entrantes para el servicio http. Verifica el estado del firewall de nuevo para confirmar que la regla (y el puerto 80) se ha añadido correctamente.

```
servidor@servidor:~$ sudo ufw allow 80/tcp
Rule added
Rule added (v6)
servidor@servidor:~$ sudo ufw status
Status: active
```

To	Action	From
--	-----	----
80/tcp	ALLOW	Anywhere
80/tcp (v6)	ALLOW	Anywhere (v6)

- 10) Abrir un Puerto Específico: Imagina que estás ejecutando un servidor de aplicaciones web en el puerto 8080. Añade una regla para permitir las conexiones entrantes TCP a ese puerto.

```
servidor@servidor:~$ sudo ufw allow 8080/tcp
Rule added
Rule added (v6)
servidor@servidor:~$ sudo ufw status
Status: active
```

To	Action	From
--	-----	----
80/tcp	ALLOW	Anywhere
8080/tcp	ALLOW	Anywhere
80/tcp (v6)	ALLOW	Anywhere (v6)
8080/tcp (v6)	ALLOW	Anywhere (v6)

5) Bloque 5: Gestión de ufw

- 1) Permitir un Rango de Puertos: Supón que una aplicación FTP necesita un rango de puertos pasivos. Añade una regla para permitir las conexiones TCP en el rango de puertos desde el 3000 al 3100.

```
servidor@servidor:~$ sudo ufw allow 3000:3100/tcp
Rule added
Rule added (v6)
servidor@servidor:~$ sudo ufw status
Status: active
```

To	Action	From
--	-----	----
80/tcp	ALLOW	Anywhere
8080/tcp	ALLOW	Anywhere
3000:3100/tcp	ALLOW	Anywhere
80/tcp (v6)	ALLOW	Anywhere (v6)
8080/tcp (v6)	ALLOW	Anywhere (v6)
3000:3100/tcp (v6)	ALLOW	Anywhere (v6)

- 2) Bloquear una Dirección IP: Por seguridad, has detectado actividad sospechosa desde la IP 192.168.100.50. Añade una regla para denegar todas las conexiones provenientes de esa dirección IP.

```
servidor@servidor:~$ sudo ufw deny from 192.168.100.50
Rule added
servidor@servidor:~$ sudo ufw status
Status: active
```

To	Action	From
--	-----	----
80/tcp	ALLOW	Anywhere
8080/tcp	ALLOW	Anywhere
3000:3100/tcp	ALLOW	Anywhere
Anywhere	DENY	192.168.100.50
80/tcp (v6)	ALLOW	Anywhere (v6)
8080/tcp (v6)	ALLOW	Anywhere (v6)
3000:3100/tcp (v6)	ALLOW	Anywhere (v6)

- 3) Listar Reglas para Borrar: Muestra todas las reglas activas del firewall, pero esta vez de forma numerada, para prepararte para eliminar una de ellas.

```
servidor@servidor:~$ sudo ufw status numbered
Status: active
```

	To	Action	From
	--	-----	----
[1]	80/tcp	ALLOW IN	Anywhere
[2]	8080/tcp	ALLOW IN	Anywhere
[3]	3000:3100/tcp	ALLOW IN	Anywhere
[4]	Anywhere	DENY IN	192.168.100.50
[5]	80/tcp (v6)	ALLOW IN	Anywhere (v6)
[6]	8080/tcp (v6)	ALLOW IN	Anywhere (v6)
[7]	3000:3100/tcp (v6)	ALLOW IN	Anywhere (v6)

- 4) Eliminar una Regla: Basándote en la lista del ejercicio anterior, elimina la regla que creaste para el puerto 8080. Vuelve a listar las reglas (de forma normal o numerada) para confirmar que la regla ha sido eliminada correctamente.

```

servidor@servidor:~$ sudo ufw status numbered
Status: active

      To      Action      From
      --      -
[ 1] 80/tcp    ALLOW IN    Anywhere
[ 2] 3000:3100/tcp ALLOW IN    Anywhere
[ 3] Anywhere  DENY IN     192.168.100.50
[ 4] 8080/tcp   ALLOW IN    Anywhere
[ 5] 3000:3100/tcp (v6) ALLOW IN    Anywhere (v6)
[ 6] 8080/tcp (v6) ALLOW IN    Anywhere (v6)

servidor@servidor:~$ sudo ufw delete 4
Deleting:
  allow 8080/tcp
Proceed with operation (y|n)? y
Rule deleted
servidor@servidor:~$ sudo ufw delete 6
ERROR: Could not find rule '6'
servidor@servidor:~$ sudo ufw delete 5
Deleting:
  allow 8080/tcp
Proceed with operation (y|n)? y
Rule deleted (v6)
servidor@servidor:~$ sudo ufw status numbered
Status: active

      To      Action      From
      --      -
[ 1] 80/tcp    ALLOW IN    Anywhere
[ 2] 3000:3100/tcp ALLOW IN    Anywhere
[ 3] Anywhere  DENY IN     192.168.100.50
[ 4] 3000:3100/tcp (v6) ALLOW IN    Anywhere (v6)

```