

# DÍA 9: GESTIÓN DE REDES Y CONECTIVIDAD EN LINUX SERVER

Por Daniel Ariza

20/06/2025

## Fase 1: Creación y organización de usuarios

- Crear los siguientes usuarios:
  - ana, carlos, elena

Para esto ejecutaremos los siguientes comandos:

**sudo adduser ana**

**sudo adduser carlos**

**sudo adduser elena**

```
daniel@ubuntucodearts:~$ sudo add user Ana
[sudo] password for daniel:
sudo: add: orden no encontrada
daniel@ubuntucodearts:~$ sudo adduser Ana
adduser: Escriba un nombre de usuario que coincida con la expresión regular configurada
a través de la variable de configuración NAME_REGEX[SYSTEM]. Use la opción «--force-badname»
para relajar esta comprobación o reconfigurar NAME_REGEX.
daniel@ubuntucodearts:~$ sudo adduser ana
Añadiendo el usuario 'ana' ...
Añadiendo el nuevo grupo 'ana' (1003) ...
Añadiendo el nuevo usuario 'ana' (1003) con grupo 'ana' ...
Creando el directorio personal '/home/ana' ...
Copiando los ficheros desde '/etc/skel' ...
Introduzca la nueva contraseña de UNIX:
Vuelva a escribir la nueva contraseña de UNIX:
passwd: contraseña actualizada correctamente
Cambiando la información de usuario para ana
Introduzca el nuevo valor, o presione INTRO para el predeterminado
Nombre completo []:
Número de habitación []:
Teléfono del trabajo []:
Teléfono de casa []:
Otro []:
¿Es correcta la información? [S/n] s
daniel@ubuntucodearts:~$
```

Adjunto la captura solo con el usuario ana ya que con los demás el proceso es exactamente el mismo.

- **Crear los grupos:**
  - **webdev, infra, docs**

Los grupos los crearemos con los siguientes comandos:

**sudo groupadd webdev**

**sudo groupadd infra**

**sudo groupadd docs**

```

Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
daniel@ubuntucodearts:~$ sudo groupadd webdev
daniel@ubuntucodearts:~$ sudo groupadd infra
daniel@ubuntucodearts:~$ sudo groupadd docs
daniel@ubuntucodearts:~$

```

- **Asignar:**
  - **ana al grupo webdev**
  - **carlos a infra**
  - **elena a docs**

Asignaremos los usuarios a los grupos con los comandos:

**sudo usermod -aG webdev ana**

**sudo usermod -aG infra carlos**

**sudo usermod -aG docs elena**

La opción **-aG agrega** al grupo sin quitar los grupos existentes del usuario.

```

Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
daniel@ubuntucodearts:~$ sudo groupadd webdev
daniel@ubuntucodearts:~$ sudo groupadd infra
daniel@ubuntucodearts:~$ sudo groupadd docs
daniel@ubuntucodearts:~$ sudo usermod -aG webdev ana
daniel@ubuntucodearts:~$ sudo usermod -aG infra carlos
daniel@ubuntucodearts:~$ sudo usermod -aG docs elena
daniel@ubuntucodearts:~$ groups ana
ana : ana webdev
daniel@ubuntucodearts:~$ groups carlos
carlos : carlos infra
daniel@ubuntucodearts:~$ groups elena
elena : elena docs
daniel@ubuntucodearts:~$ _

```

Hemos verificado que se han añadido bien con el comando `groups`

- **Establecer contraseñas seguras para cada usuario.**

Al crear los nuevos usuarios ya establecimos contraseñas para cada uno de ellos como puede verse en la captura de más arriba.

## **Fase 2: Estructura de directorios y control de acceso**

- **Crear las siguientes carpetas:**
  - `/grupos/web`, `/grupos/infra`, `/grupos/docs`

Con el comando `sudo mkdir -p /grupos/web /grupos/infra /grupos/docs` crearemos las carpetas y con `ls -l /grupos` verificamos que se han creado correctamente.

```

daniel@ubuntu:~$ sudo mkdir -p /grupos/web /grupos/infra /grupos/docs
[sudo] password for daniel:
Lo sentimos, vuelva a intentarlo.
[sudo] password for daniel:
daniel@ubuntu:~$ ls -l /grupos
total 12
drwxr-xr-x 2 root root 4096 jun 23 07:13 docs
drwxr-xr-x 2 root root 4096 jun 23 07:13 infra
drwxr-xr-x 2 root root 4096 jun 23 07:13 web
daniel@ubuntu:~$

```

- **Asignar propietario al grupo correspondiente y cambiar permisos:**
  - **Solo los miembros del grupo pueden leer y escribir en su carpeta.**

Con estos comandos cambiamos el grupo propietario de cada carpeta: **sudo chown :webdev /grupos/web**

**sudo chown :infra /grupos/infra**

**sudo chown :docs /grupos/docs**

```

daniel@ubuntu:~$ sudo chown :webdev /grupos/web
daniel@ubuntu:~$ sudo chown :infra /grupos/infra
daniel@ubuntu:~$ sudo chown :docs /grupos/docs
daniel@ubuntu:~$

```

- **Otros usuarios no deben tener acceso.**

Con estos comandos: **sudo chmod 770 /grupos/web**

**sudo chmod 770 /grupos/infra**

**sudo chmod 770 /grupos/docs** estableceremos los permisos 770 para cada carpeta. Teniendo en cuenta que:

**7** para el **propietario** (root): **rw** = leer, escribir, ejecutar.

7 para el **grupo** (webdev): **rwX** = leer, escribir, ejecutar.

0 para **otros usuarios**: **---** = sin permisos.

Y para verificar que todo se ha configurado bien ejecutaremos **ls -ld /grupos/\***

```
daniel@ubuntucodearts:~$ sudo chmod 770 /grupos/web
daniel@ubuntucodearts:~$ sudo chmod 770 /grupos/infra
daniel@ubuntucodearts:~$ sudo chmod 770 /grupos/docs
daniel@ubuntucodearts:~$ ls -ls /grupos/*
ls: no se puede abrir el directorio '/grupos/docs': Permiso denegado
ls: no se puede abrir el directorio '/grupos/infra': Permiso denegado
ls: no se puede abrir el directorio '/grupos/web': Permiso denegado
daniel@ubuntucodearts:~$ sudo ls -ld /grupos/*
drwxrwx--- 2 root docs 4096 jun 23 07:13 /grupos/docs
drwxrwx--- 2 root infra 4096 jun 23 07:13 /grupos/infra
drwxrwx--- 2 root webdev 4096 jun 23 07:13 /grupos/web
```

- **Aplicar chmod y chown correctamente para:**
  - **Establecer permisos 770.**

Esto se realizó en el paso anterior.

- **Activar setgid para que los nuevos archivos hereden el grupo.**

Esto hará que todo lo que se cree dentro de cada carpeta herede el grupo de la carpeta, no el grupo del usuario que crea el archivo. Utilizaremos los siguientes comandos:

**sudo chmod g+s /grupos/web**

**sudo chmod g+s /grupos/infra**

**sudo chmod g+s /grupos/docs**

**+s** → activa el bit **setgid** (Set Group ID)

Esto cambia el comportamiento de herencia del grupo en esa carpeta.

Y verificaremos con: **ls -ld /grupos/\***

```
daniel@ubuntucodearts:~$ sudo chmod g+s /grupos/web
daniel@ubuntucodearts:~$ sudo chmod g+s /grupos/infra
daniel@ubuntucodearts:~$ sudo chmod g+s /grupos/docs
daniel@ubuntucodearts:~$ sudo ls -ld /grupos/*
drwxrws--- 2 root docs  4096 jun 23 07:13 /grupos/docs
drwxrws--- 2 root infra 4096 jun 23 07:13 /grupos/infra
drwxrws--- 2 root webdev 4096 jun 23 07:13 /grupos/web
daniel@ubuntucodearts:~$ _
```

Esa **s** indica que el bit **setgid** está activo, y todo lo nuevo que se cree dentro de esas carpetas heredará automáticamente el grupo.

### Fase 3: Configuración avanzada de permisos y restricciones

#### ● Crear en /grupos/docs un archivo llamado plan.txt.

Utilizaremos el siguiente comando para crear el archivo mediante el usuario Elena que es quién está en el grupo docs: **sudo -u elena touch /grupos/docs/plan.txt**

Y con **sudo ls -l /grupos/docs** verificamos que el archivo se ha creado y tiene el grupo correcto.

```
daniel@ubuntucodearts:~$ sudo -u elena touch /grupos/docs/plan.txt
[sudo] password for daniel:
Lo sentimos, vuelva a intentarlo.
[sudo] password for daniel:
Lo sentimos, vuelva a intentarlo.
[sudo] password for daniel:
daniel@ubuntucodearts:~$ ls -l /grupos/docs
ls: no se puede abrir el directorio '/grupos/docs': Permiso denegado
daniel@ubuntucodearts:~$ sudo ls -l /grupos/docs
total 0
-rw-r--r-- 1 elena docs 0 jun 23 08:22 plan.txt
daniel@ubuntucodearts:~$ _
```

- Permitir que solo elena pueda modificarlo, pero que ana y carlos puedan leerlo sin poder editarlo.

Actualmente el archivo pertenece al grupo **docs**, y solo **elena** es miembro de ese grupo.

Como **ana**, **carlos** y **elena** no comparten un mismo grupo, la solución más adecuada es crear un **grupo nuevo compartido**. A este grupo lo llamaremos **lectores** y añadiremos carlos y ana. Para ello utilizaremos los siguientes comandos:

```
daniel@ubuntucodearts:~$ sudo groupadd lectores
daniel@ubuntucodearts:~$ sudo usermod -aG lectores ana
daniel@ubuntucodearts:~$ sudo usermod -aG lectores carlos
daniel@ubuntucodearts:~$
```

**sudo groupadd lectores**

**sudo usermod -aG lectores ana**

**sudo usermod -aG lectores carlos**

Ahora cambiaremos el grupo del archivo con:

**sudo chown elena:lectores /grupos/docs/plan.txt**

```
daniel@ubuntucodearts:~$ sudo chown elena:lectores /grupos/docs/plan.txt
daniel@ubuntucodearts:~$ _
```

Y ahora ajustaremos los permisos del archivo:

Queremos:

**elena: rw-** (lectura y escritura)

**Grupo (lectores): r--** (solo lectura)

**Otros: ---** (sin permisos)

Para esto ejecutaremos: **sudo chmod 640 /grupos/docs/plan.txt**

Y verificaremos con : **ls -l /grupos/docs/plan.txt**

```
daniel@ubuntucodearts:~$ sudo chmod 640 /grupos/docs/plan.txt
daniel@ubuntucodearts:~$ sudo -l /grupos/docs/plan.txt
sudo: /grupos/docs/plan.txt: orden no encontrada
daniel@ubuntucodearts:~$ sudo ls -l /grupos/docs/plan.txt
-rw-r----- 1 elena lectores 0 jun 23 08:22 /grupos/docs/plan.txt
daniel@ubuntucodearts:~$
```

Esto confirma: **elena** puede leer y escribir  
**Ana y carlos** (como miembros del grupo **lectores**) pueden leer

Ningún otro usuario puede acceder.

- **Configurar un grupo compartido llamado lectura e incluir a los tres usuarios.**

Esto se realiza en el paso anterior.

- **Usar ACLs (Access Control Lists) para asignar permisos finos.**

Vamos a realizar lo siguiente en cuanto a permisos:

**elena:** lectura y escritura del archivo **plan.txt**

**ana y carlos:** solo lectura, todos los demás: sin acceso.

Primero vamos a asegurarnos de que tenemos habilitado ACL en el sistema, lo haremos con: **mount | grep 'on /'**

Podemos ver que si está habilitado.



```
daniel@ubuntucodearts:~$ mount | grep 'on /'
sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,relatime)
proc on /proc type proc (rw,nosuid,nodev,noexec,relatime)
udev on /dev type devtmpfs (rw,nosuid,relatime,size=1003956k,nr_inodes=250989,mode=755)
devpts on /dev/pts type devpts (rw,nosuid,noexec,relatime,gid=5,mode=620,ptmxmode=000)
tmpfs on /run type tmpfs (rw,nosuid,noexec,relatime,size=204820k,mode=755)
/dev/sda1 on / type ext4 (rw,relatime,errors=remount-ro,data=ordered)
securityfs on /sys/kernel/security type securityfs (rw,nosuid,nodev,noexec,relatime)
tmpfs on /dev/shm type tmpfs (rw,nosuid,nodev)
tmpfs on /run/lock type tmpfs (rw,nosuid,nodev,noexec,relatime,size=5120k)
tmpfs on /sys/fs/cgroup type tmpfs (ro,nosuid,nodev,noexec,mode=755)
cgroup on /sys/fs/cgroup/systemd type cgroup (rw,nosuid,nodev,noexec,relatime,xattr,release_agent=/lib/systemd/systemd-cgroups-agent,name=systemd)
pstore on /sys/fs/pstore type pstore (rw,nosuid,nodev,noexec,relatime)
cgroup on /sys/fs/cgroup/cpuset type cgroup (rw,nosuid,nodev,noexec,relatime,cpuset)
cgroup on /sys/fs/cgroup/net_cls,net_prio type cgroup (rw,nosuid,nodev,noexec,relatime,net_cls,net_prio)
cgroup on /sys/fs/cgroup/pids type cgroup (rw,nosuid,nodev,noexec,relatime,pids)
cgroup on /sys/fs/cgroup/devices type cgroup (rw,nosuid,nodev,noexec,relatime,devices)
cgroup on /sys/fs/cgroup/memory type cgroup (rw,nosuid,nodev,noexec,relatime,memory)
cgroup on /sys/fs/cgroup/hugetlb type cgroup (rw,nosuid,nodev,noexec,relatime,hugetlb)
cgroup on /sys/fs/cgroup/cpu,cpuacct type cgroup (rw,nosuid,nodev,noexec,relatime,cpu,cpuacct)
cgroup on /sys/fs/cgroup/freezer type cgroup (rw,nosuid,nodev,noexec,relatime,freezer)
cgroup on /sys/fs/cgroup/blkio type cgroup (rw,nosuid,nodev,noexec,relatime,blkio)
cgroup on /sys/fs/cgroup/perf_event type cgroup (rw,nosuid,nodev,noexec,relatime,perf_event)
systemd-1 on /proc/sys/fs/binfmt_misc type autofs (rw,relatime,fd=31,pgrp=1,timeout=0,minproto=5,maxproto=5,direct)
mqueue on /dev/mqueue type mqueue (rw,relatime)
debugfs on /sys/kernel/debug type debugfs (rw,relatime)
hugetlbfs on /dev/hugepages type hugetlbfs (rw,relatime)
fusectl on /sys/fs/fuse/connections type fusectl (rw,relatime)
tmpfs on /run/user/1000 type tmpfs (rw,nosuid,nodev,relatime,size=204820k,mode=700,uid=1000,gid=1000)
daniel@ubuntucodearts:~$
```

Ahora vamos a asegurarnos de que el archivo pertenezca a elena con: **sudo chown elena /grupos/docs/plan.txt**

Seguimos aplicando las ACL asignando los permisos con setfacl:

**# Dar a elena permisos de lectura y escritura**

**sudo setfacl -m u:elena:rw /grupos/docs/plan.txt**

**# Dar a ana y carlos permisos de solo lectura**

**sudo setfacl -m u:ana:r /grupos/docs/plan.txt**

**sudo setfacl -m u:carlos:r /grupos/docs/plan.txt**

**# Asegurar que los demás usuarios no tengan permisos**

`sudo setfacl -m o::--- /grupos/docs/plan.txt`

```
daniel@ubuntucodearts:~$ sudo setfacl -m u:elena:rw /grupos/docs/plan.txt
daniel@ubuntucodearts:~$ sudo setfacl -m u:ana:r /grupos/docs/plan.txt
[sudo] password for daniel:
Lo sentimos, vuelva a intentarlo.
[sudo] password for daniel:
daniel@ubuntucodearts:~$ sudo setfacl -m u:carlos:r /grupos/docs/plan.txt
daniel@ubuntucodearts:~$ sudo setfacl -m o::--- /grupos/docs/plan.txt
daniel@ubuntucodearts:~$ _
```

Por último verificamos la configuración de la ACL con:

`getfacl /grupos/docs/plan.txt`

```
daniel@ubuntucodearts:~$ sudo getfacl /grupos/docs/plan.txt
getfacl: Eliminando «/» inicial en nombres de ruta absolutos
# file: grupos/docs/plan.txt
# owner: elena
# group: lectores
user::rw-
user:ana:r--
user:carlos:r--
user:elena:rw-
group::r--
mask::rw-
other::---
```

Vemos

que la ACL se realizó correctamente.

## Fase 4: Buenas prácticas y seguridad básica

- Establecer una política de caducidad de contraseñas de 60 días para todos los usuarios.

Primero estableceremos la política por defecto para usuarios nuevos editando el archivo de configuración global `sudo nano /etc/login.defs` ajustando estas líneas o agregándolas si no existen:

`PASS_MAX_DAYS 60` la contraseña caduca a los 60 días.

**PASS\_MIN\_DAYS 0** se puede cambiar la contraseña en cualquier momento

**PASS\_WARN\_AGE 7** el sistema avisará 7 días antes de que caduque

```
# PASS_MAX_DAYS 60
# PASS_MIN_DAYS 0
# PASS_WARN_AGE 7_
```

Guardamo

s y salimos del archivo, Esto **solo se aplica a nuevos usuarios** creados después de este cambio. Para usuarios actuales lo haremos en el siguiente paso.

Ahora lo aplicaremos a los usuarios ya existentes con:

**sudo chage -M 60 -W 7 -m 0 ana**

**sudo chage -M 60 -W 7 -m 0 carlos**

**sudo chage -M 60 -W 7 -m 0 elena**

y verificaremos con: **sudo chage -l ana** incluyendo el nombre de los distintos usuarios donde está el de ana.

```
daniel@ubuntucodearts:~$ sudo chage -l ana
Último cambio de contraseña           : jun 20, 2025
La contraseña caduca                   : ago 19, 2025
Contraseña inactiva                    : nunca
La cuenta caduca                       : nunca
Número de días mínimo entre cambio de contraseña : 0
Número de días máximo entre cambio de contraseña : 60
Número de días de aviso antes de que caduque la contraseña : 7
daniel@ubuntucodearts:~$ _
```

Podemos ver que todo está bien.

## ● Bloquear el acceso SSH al usuario elena.

Para esto editaremos el archivo de configuración de SSH:

**sudo nano /etc/ssh/sshd\_config**

Añadiremos esta línea al final del archivo: **DenyUsers elena**

Reiniciamos SSh con: **sudo systemctl restart ssh**

```
GNU nano 2.5.3          Archivo: /etc/ssh/sshd_config

#UseLogin no

#MaxStartups 10:30:60
#Banner /etc/issue.net

# Allow client to pass locale environment variables
AcceptEnv LANG LC_*

Subsystem sftp /usr/lib/openssh/sftp-server

# Set this to 'yes' to enable PAM authentication, account processing,
# and session processing. If this is enabled, PAM authentication will
# be allowed through the ChallengeResponseAuthentication and
# PasswordAuthentication. Depending on your PAM configuration,
# PAM authentication via ChallengeResponseAuthentication may bypass
# the setting of "PermitRootLogin without-password".
# If you just want the PAM account and session checks to run without
# PAM authentication, then enable this but set PasswordAuthentication
# and ChallengeResponseAuthentication to 'no'.
UsePAM yes
DenyUsers elena
```

- **Crear un alias de shell en /etc/skel/.bashrc para que todos los nuevos usuarios vean un mensaje de bienvenida personalizado al iniciar sesión.**

Empezaremos editando el archivo de la plantilla con: **sudo nano /etc/skel/.bashrc**

Y al final del archivo añadiremos esta línea: `echo "¡Bienvenido al sistema, disfruta tu sesión!"` 🎉

```
GNU nano 2.5.3 Archivo: /etc/skel/.bashrc

if [ -f ~/.bash_aliases ]; then
    . ~/.bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
    if [ -f /usr/share/bash-completion/bash_completion ]; then
        . /usr/share/bash-completion/bash_completion
    elif [ -f /etc/bash_completion ]; then
        . /etc/bash_completion
    fi
fi
echo "¡Bienvenido al sistema, disfruta tu sesión!"
```

Para verificarlo crearemos un usuario de prueba llamado **prueba1**.

```
daniel@ubuntucodearts:~$ sudo adduser prueba1
Añadiendo el usuario 'prueba1' ...
Añadiendo el nuevo grupo 'prueba1' (1010) ...
Añadiendo el nuevo usuario 'prueba1' (1006) con grupo 'prueba1' ...
Creando el directorio personal '/home/prueba1' ...
Copiando los ficheros desde '/etc/skel' ...
Introduzca la nueva contraseña de UNIX:
Vuelva a escribir la nueva contraseña de UNIX:
passwd: contraseña actualizada correctamente
Cambiando la información de usuario para prueba1
Introduzca el nuevo valor, o presione INTRO para el predeterminado
Nombre completo []:
Número de habitación []:
Teléfono del trabajo []:
Teléfono de casa []:
Otro []:
¿Es correcta la información? [S/n]
daniel@ubuntucodearts:~$
daniel@ubuntucodearts:~$ su - prueba1
Contraseña:
¡Bienvenido al sistema, disfruta tu sesión!
prueba1@ubuntucodearts:~$
```

Como podemos ver se ha realizado correctamente.