

Bloque 2: Día 8

Fase: 1

Para empezar crearemos tres usuarios llamados dev_user, sysadmin y intern_user, a cada usuario le asignaremos un grupo especial llamados developers, admins e interns. Y configuraremos los permisos de a carpetas básicas a cada grupo.

Empezaremos por crear los tres usuarios con el comando sudo adduser. Y después creamos los grupos con el comando sudo groupadd.

```
LinuxServer [Corriendo] - Oracle VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
cethfisto@srv-base-cesar:~$ getent group users
users:x:100:administrador,desarrollador,dev_user,sysadmin,intern_user
cethfisto@srv-base-cesar:~$
```

```
LinuxServer [Corriendo] - Oracle VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
cethfisto@srv-base-cesar:~$ sudo groupadd developers
cethfisto@srv-base-cesar:~$ sudo groupadd admins
cethfisto@srv-base-cesar:~$ sudo groupadd interns
cethfisto@srv-base-cesar:~$ _
```

```
LinuxServer [Corriendo] - Oracle VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
cethfisto@srv-base-cesar:~$ sudo usermod -aG admins sysadmin
cethfisto@srv-base-cesar:~$ sudo usermod -aG developers dev_user
cethfisto@srv-base-cesar:~$ sudo usermod -aG interns intern_user
cethfisto@srv-base-cesar:~$ getent group admins
admins:x:1007:sysadmin
cethfisto@srv-base-cesar:~$ getent group developers
developers:x:1006:dev_user
cethfisto@srv-base-cesar:~$ getent group interns
interns:x:1008:intern_user
cethfisto@srv-base-cesar:~$
```

Ahora configuraremos los permisos de las carpetas, crearemos las carpetas si es necesario con mkdir, asignamos el root con chown y los permisos con chmod.

La configuración sería:

- /opt/dev_projects : acceso solo para developers.
- /var/log/dev_logs : acceso solo para admins.
- /home : para todos los usuarios.
- /home/shared_docs : acceso de lectura solo para interns.

```
LinuxServer [Corriendo] - Oracle VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
cethfisto@srv-base-cesar:~$ sudo mkdir -p /opt/dev_projects
cethfisto@srv-base-cesar:~$ sudo chown root:developers /opt/dev_projects
cethfisto@srv-base-cesar:~$ sudo chmod 770 /opt/dev_projects
chmod: cannot access '/opt/dev_projects': No such file or directory
cethfisto@srv-base-cesar:~$ sudo chmod 770 /opt/dev_projects
cethfisto@srv-base-cesar:~$ _
```

```
LinuxServer [Corriendo] - Oracle VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
cethfisto@srv-base-cesar:~$ sudo mkdir -p /var/log/dev_logs
cethfisto@srv-base-cesar:~$ sudo chown root:admins /var/log/dev_logs
chown: cannot access '/var/log/dev_logs': No such file or directory
cethfisto@srv-base-cesar:~$ sudo chown root:admins /var/log/dev_logs
cethfisto@srv-base-cesar:~$ sudo chmod 750 /var/log/dev_logs
cethfisto@srv-base-cesar:~$ _
```

```
LinuxServer [Corriendo] - Oracle VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
cethfisto@srv-base-cesar:~$ sudo mkdir -p /home/shared_docs
cethfisto@srv-base-cesar:~$ sudo chown root:interns /home/shared_docs
cethfisto@srv-base-cesar:~$ sudo chmod 750 /home/shared_docs
cethfisto@srv-base-cesar:~$ _
```

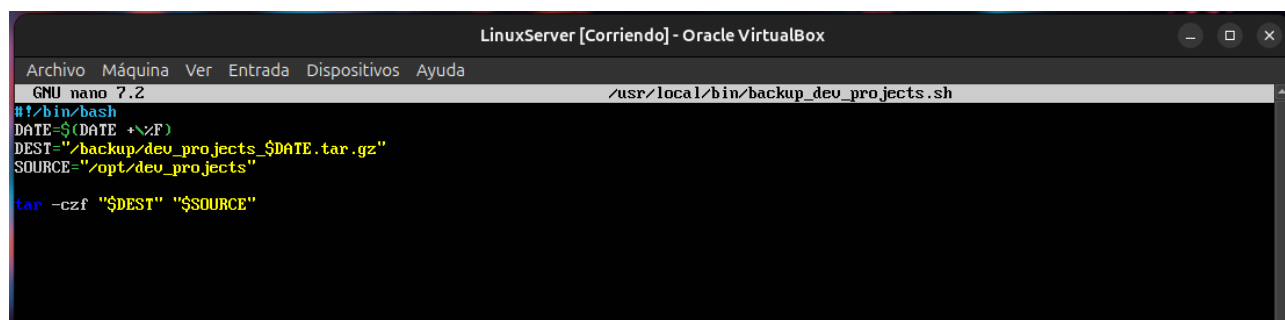
Fase: 2

Programaremos un cron para realizar un backup automático de un directorio. Y crearemos un script que envíe notificaciones de actividad del servidor.

Vamos a crear un tarea en cron para hacer backups del directorio /opt/dev_projects de forma automática. Primero crearemos el directorio de backups para el grupo de developers con sus correspondientes permisos .

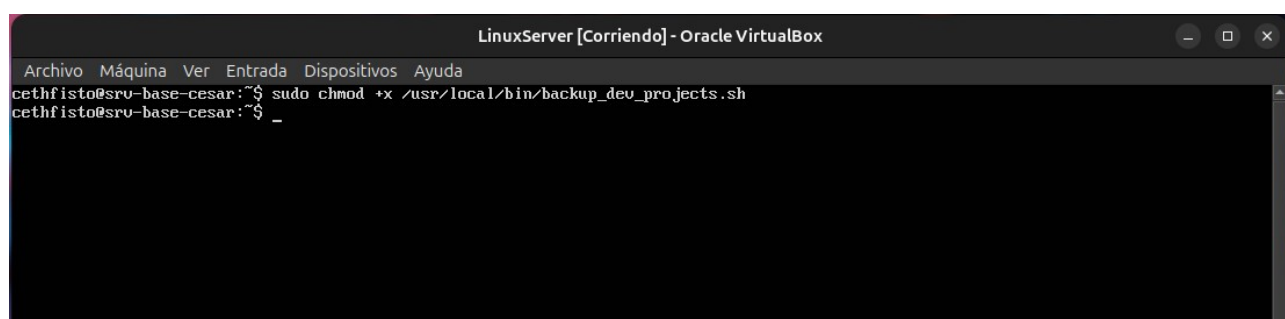
```
LinuxServer [Corriendo] - Oracle VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
cethfisto@srv-base-cesar:~$ sudo mkdir -p /backups
[sudo] password for cethfisto:
cethfisto@srv-base-cesar:~$ sudo chown root:developers /backups
cethfisto@srv-base-cesar:~$ sudo chmod 770 /backups
cethfisto@srv-base-cesar:~$ _
```

Después crearemos y daremos el permiso de ejecución a un script que cogerá el contenido del directorio /dev_projects, lo comprimirá en formato .tar .gz y los guardará en el directorio /backups.



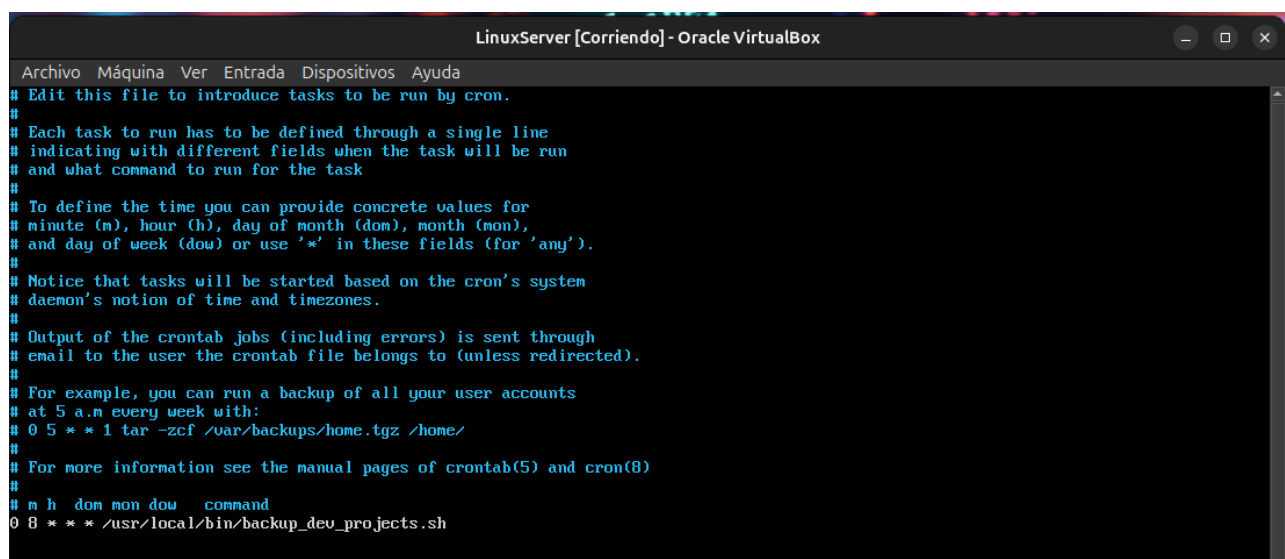
```
LinuxServer [Corriendo] - Oracle VirtualBox
GNU nano 7.2 /usr/local/bin/backup_dev_projects.sh
#!/bin/bash
DATE=$(date +%F)
DEST="/backup/dev_projects_${DATE}.tar.gz"
SOURCE="/opt/dev_projects"
tar -czf "$DEST" "$SOURCE"
```

Estos comandos indican que se hará un backup en el directorio /backups el contenido de /opt/dev_projects.



```
LinuxServer [Corriendo] - Oracle VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
cethfisto@srv-base-cesar:~$ sudo chmod +x /usr/local/bin/backup_dev_projects.sh
cethfisto@srv-base-cesar:~$ _
```

Y para completar la mitad de esta fase, crearemos la tarea cron para que haga los backups a las 8 am, es decir, ejecutará el script cada día a las 8 am. Para ello usaremos el comando sudo crontab -e.



```
LinuxServer [Corriendo] - Oracle VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
# 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
0 8 * * * /usr/local/bin/backup_dev_projects.sh
```

Añadiremos el comando de abajo que indica que todos los días a las 8 am se ejecutará el script creado para hacer los backups.

Ahora crearemos un script que envíe notificaciones sobre la actividad del servidor y lo guarde en un archivo log. Por ejemplo: usuarios conectados, uso CPU, etc. Y después de crear el script le daremos permiso de ejecución.

```
GNU nano 2.2 /usr/local/bin/monitor_server.sh
#!/bin/bash
LOG="/var/log/server_activity.log"
DATE=$(date +%Y-%m-%d %H:%M:%S)

echo "[${DATE}] === ACTIVIDAD DEL SERVER ===" >> $LOG
echo "Usuarios conectados:" >> $LOG
who >> $LOG
```

```
LinuxServer [Corriendo] - Oracle VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
cethfisto@srv-base-cesar:~$ sudo chmod +x /usr/local/bin/monitor_server.sh
[sudo] password for cethfisto:
cethfisto@srv-base-cesar:~$ _
```

Después añadimos el script al archivo cron para que se ejecute cada hora.

```
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow  command
0 8 * * * /usr/local/bin/backup_dev_projects.sh
0 * * * * /usr/local/bin/monitor_server.sh

crontab: installing new crontab
cethfisto@srv-base-cesar:~$ _
```

Podremos ver el registro de cada hora haciendo el comando `sudo nano /var/log/server_activity.log`

```
GNU nano 7.2 server_activity.log
[2025-06-19 19:00:01] === ACTIVIDAD DEL SERVER ===
Usuarios conectados:
cethfisto tty1      2025-06-19 16:40
```

Fase: 3

Aquí, identificaremos procesos en ejecución y ajustaremos su prioridad en caso necesario. Y configuraremos logs de auditoria para registrar accesos y acciones de los usuarios.

Para identificar los procesos usaremos el comando `top` o `htop`. Nos saldrá una lista de comandos en ejecución.

PID	USER	PRI	NI	VIRT	RES	SHR	S	CPU%	MEM%	TIME+	Command
1729	cethfisto	20	0	8684	4736	3584	R	0.6	0.1	0:00.04	htop
1	root	20	0	22024	12904	9320	S	0.0	0.3	0:00.70	/sbin/init splash noprompt noshell automatic-ubiquity
302	root	19	-1	50444	16164	15140	S	0.0	0.4	0:00.18	/usr/lib/systemd/systemd-journald
352	root	RT	0	282M	27136	8704	S	0.0	0.7	0:00.07	/sbin/multipathd -d -s
365	root	20	0	29056	7600	4992	S	0.0	0.2	0:00.00	/usr/lib/systemd/systemd-udevd
366	root	20	0	282M	27136	8704	S	0.0	0.7	0:00.00	/sbin/multipathd -d -s
367	root	RT	0	282M	27136	8704	S	0.0	0.7	0:00.00	/sbin/multipathd -d -s
368	root	RT	0	282M	27136	8704	S	0.0	0.7	0:00.00	/sbin/multipathd -d -s
369	root	RT	0	282M	27136	8704	S	0.0	0.7	0:00.00	/sbin/multipathd -d -s
370	root	RT	0	282M	27136	8704	S	0.0	0.7	0:00.07	/sbin/multipathd -d -s
371	root	RT	0	282M	27136	8704	S	0.0	0.7	0:00.00	/sbin/multipathd -d -s
429	systemd-re	20	0	21584	12800	10624	S	0.0	0.3	0:00.06	/usr/lib/systemd/systemd-resolved
434	systemd-ti	20	0	91020	7808	6912	S	0.0	0.2	0:00.04	/usr/lib/systemd/systemd-timesyncd
523	systemd-ti	20	0	91020	7808	6912	S	0.0	0.2	0:00.00	/usr/lib/systemd/systemd-timesyncd
592	systemd-ne	20	0	18996	9216	8192	S	0.0	0.2	0:00.03	/usr/lib/systemd/systemd-networkd
645	messagebus	20	0	9788	5376	4608	S	0.0	0.1	0:00.05	dbus-daemon --system --address=systemd: --nofork --nopidfile --systemd-activation
654	polkitd	20	0	300M	7808	7040	S	0.0	0.2	0:00.03	/usr/lib/polkit-1/polkitd --no-debug
671	root	20	0	18124	8576	7680	S	0.0	0.2	0:00.06	/usr/lib/systemd/systemd-logind
682	root	20	0	457M	13568	11520	S	0.0	0.3	0:00.03	/usr/libexec/udisks2/udisksd
720	root	20	0	107M	22908	13568	S	0.0	0.6	0:00.07	/usr/bin/python3 /usr/share/unattended-upgrades/unattended-upgrade-shutdown --wait-
722	root	20	0	457M	13568	11520	S	0.0	0.3	0:00.07	/usr/libexec/udisks2/udisksd
723	root	20	0	457M	13568	11520	S	0.0	0.3	0:00.00	/usr/libexec/udisks2/udisksd
727	syslog	20	0	217M	6016	4608	S	0.0	0.2	0:00.01	/usr/sbin/rsyslogd -n -iNONE
733	root	20	0	457M	13568	11520	S	0.0	0.3	0:00.00	/usr/libexec/udisks2/udisksd
753	polkitd	20	0	300M	7808	7040	S	0.0	0.2	0:00.11	/usr/lib/polkit-1/polkitd --no-debug
755	polkitd	20	0	300M	7808	7040	S	0.0	0.2	0:00.00	/usr/lib/polkit-1/polkitd --no-debug
757	polkitd	20	0	300M	7808	7040	S	0.0	0.2	0:00.00	/usr/lib/polkit-1/polkitd --no-debug
767	root	20	0	382M	12672	10752	S	0.0	0.3	0:00.07	/usr/sbin/ModemManager
773	syslog	20	0	217M	6016	4608	S	0.0	0.2	0:00.01	/usr/sbin/rsyslogd -n -iNONE
774	syslog	20	0	217M	6016	4608	S	0.0	0.2	0:00.00	/usr/sbin/rsyslogd -n -iNONE
775	syslog	20	0	217M	6016	4608	S	0.0	0.2	0:00.00	/usr/sbin/rsyslogd -n -iNONE
783	root	20	0	457M	13568	11520	S	0.0	0.3	0:00.00	/usr/libexec/udisks2/udisksd
785	root	20	0	382M	12672	10752	S	0.0	0.3	0:00.00	/usr/sbin/ModemManager
788	root	20	0	382M	12672	10752	S	0.0	0.3	0:00.00	/usr/sbin/ModemManager
791	root	20	0	382M	12672	10752	S	0.0	0.3	0:00.00	/usr/sbin/ModemManager
824	root	20	0	457M	13568	11520	S	0.0	0.3	0:00.00	/usr/libexec/udisks2/udisksd
877	root	20	0	107M	22908	13568	S	0.0	0.6	0:00.00	/usr/bin/python3 /usr/share/unattended-upgrades/unattended-upgrade-shutdown --wait-
892	root	20	0	6824	2688	2560	S	0.0	0.1	0:00.01	/usr/sbin/cron -f -P
900	root	20	0	12020	7936	6912	S	0.0	0.2	0:00.00	sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups
903	root	20	0	11160	1588	640	S	0.0	0.0	0:00.00	nginx: master process /usr/sbin/nginx -g daemon on; master_process on;
904	www-data	20	0	12880	4404	3200	S	0.0	0.1	0:00.00	nginx: worker process

Para cambiar la prioridad debemos ajustar los valores (que van desde -20 muy alta al +19 muy baja). Para ello usaremos el comando `renice`. Por ejemplo: `sudo renice +X -p XXXX`.

```
LinuxServer [Corriendo] - Oracle VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda

cethfisto@srv-base-cesar:~$ sudo renice -20 -p 302
302 (process ID) old priority -1, new priority -20
cethfisto@srv-base-cesar:~$
```

```
302 root      0 -20 50444 16164 15140 S   0.0  0.4  0:00.19 /usr/lib/systemd/systemd-journald
```

Para configurar los logs de auditoria, primero debemos instalar el sistema de auditoria. Usaremos el comando `sudo apt install auditd audispd-plugins`.

Una vez instalado crearemos una regla de auditoria con el comando `auditctl`. Vamos a registrar qué usuarios acceden a `/opt/dev_projects` por ejemplo.

```
cethfisto@srv-base-cesar:~$ sudo auditctl -w /opt/dev_projects -p rwx -k dev_projects_watch
cethfisto@srv-base-cesar:~$ _
```

Podremos revisar quien accedo al directorio con el comando `ausearch`.

```
cethfisto@srv-base-cesar:~$ sudo ausearch -k dev_projects_watch
[sudo] password for cethfisto:
-----
time->Thu Jun 19 19:12:11 2025
type=PROCTITLE msg=audit(1750353131.651:142): proctitle=617564697463746C002D77002F6F70742F6465765F70726F6A65637473002D700072777861002D6B006465765F70726F6A656374735F7761746368
type=SYSCALL msg=audit(1750353131.651:142): arch=c000003e syscall=44 success=yes exit=1092 a0=4 a1=7fffd78d78fb0 a2=444 a3=0 items=0 ppid=1396 pid=1397 auid=1000 uid=0 gid=0 euid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts0 ses=1 comm="auditctl" exe="/usr/sbin/auditctl" subj=unconfined key=(null)
type=CONFIG_CHANGE msg=audit(1750353131.651:142): auid=1000 ses=1 subj=unconfined op=add_rule key="dev_projects_watch" list=4 res=1
cethfisto@srv-base-cesar:~$
```

Fase: 4

Explicaremos lo importante que es administrar usuarios y seguridad en servidores Linux. Y porqué nuestra configuración mejora la gestión interna de CodeArts.

Respecto a la administración de usuarios y seguridad en servidores Linux, nos dará control personalizado a cada usuario según las necesidades. También tendremos seguridad frente amenazas, ya que limitaremos los ataques al limitar permisos y registramos la actividad. Y ganaremos organización a la hora de agrupar los usuarios según el grupo que le corresponda.

¿Por qué nuestra configuración mejora la gestión de CodeArts? Conseguimos separar los roles de cada empleado limitando su acceso y permisos. Tendremos seguridad en los datos, ya que con los logs de auditoria tendremos un registro de accesos y modificación dentro del servidor. Y ganaremos escalabilidad , ya que tener todo organizado y estructurado nos permitirá añadir más directorio o usuarios.