

# Programación de Tareas con Cron y Scripts en Linux

## Tarea 1: Investigación sobre Cron y Scripts en Bash

### 1.1 ¿Qué es cron y cómo funciona en Linux?

**cron** es un planificador de tareas basado en tiempo en sistemas operativos tipo Unix (como Linux). Permite a los usuarios ejecutar comandos o scripts automáticamente en un horario fijo. Imagina que es como un despertador para tu ordenador, pero en lugar de sonar, ejecuta una acción específica.

#### ¿Cómo funciona?

- **Demonio crond:** En segundo plano, un proceso llamado **crond** (el demonio de cron) se ejecuta continuamente.
- **Tabla de Cron (crontab):** crond lee las instrucciones de un archivo llamado **crontab** (cron table). Este archivo contiene una lista de tareas (llamadas "trabajos de cron") y los horarios en los que deben ejecutarse.
- **Programación:** Cuando el tiempo del sistema coincide con el horario especificado para un trabajo en la **crontab**, **crond** ejecuta el comando o script asociado a ese trabajo.

### 1.2 Sintaxis de las expresiones de tiempo en cron (ejemplo: \* \* \* \* \*)

Cada línea en el archivo **crontab** representa un trabajo de cron y sigue una sintaxis específica para definir el horario de ejecución. La sintaxis básica es:

**minuto hora día\_del\_mes mes día\_de\_la\_semana comando**

Cada uno de estos campos puede tomar diferentes valores:

- **Minuto (0-59):** El minuto de la hora en que se ejecutará la tarea.
- **Hora (0-23):** La hora del día (en formato de 24 horas).
- **Día del mes (1-31):** El día del mes en que se ejecutará la tarea.
- **Mes (1-12 o JAN-DEC):** El mes del año en que se ejecutará la tarea.
- **Día de la semana (0-7 o SUN-SAT):** El día de la semana (0 o 7 representa el domingo).

**Comodines:**

- \*: Significa "todos los valores posibles". Por ejemplo, \* en el campo de los minutos significa que la tarea se ejecutará cada minuto.
- ,: Se utiliza para especificar una lista de valores. Ejemplo: 1,15,30 en el campo de los minutos significa que la tarea se ejecutará en los minutos 1, 15 y 30 de cada hora.
- -: Se utiliza para especificar un rango de valores. Ejemplo: 10-15 en el campo de los minutos significa que la tarea se ejecutará en los minutos 10, 11, 12, 13, 14 y 15 de cada hora.
- /: Se utiliza para especificar un paso o intervalo. Ejemplo: \*/5 en el campo de los minutos significa que la tarea se ejecutará cada 5 minutos (empezando desde el minuto 0).

### Ejemplos:

- \* \* \* \* \* **comando:** Ejecutar comando cada minuto.
- 0 \* \* \* \* **comando:** Ejecutar comando al inicio de cada hora.
- 0 2 \* \* \* **comando:** Ejecutar comando a las 2:00 AM todos los días.
- 0 2 \* \* 0 **comando:** Ejecutar comando a las 2:00 AM todos los domingos.
- 0 2 1 \* \* **comando:** Ejecutar comando a las 2:00 AM el primer día de cada mes.

### 1.3 ¿Cómo se listan, crean y eliminan trabajos de cron con crontab?

La herramienta principal para gestionar los trabajos de cron es **crontab**.

- **Listar trabajos de cron:** Para ver los trabajos de cron programados para el usuario actual, abre la terminal y ejecuta el siguiente comando:

**crontab -l**

Si no hay trabajos programados, no se mostrará nada o un mensaje indicándolo.

- **Crear o editar trabajos de cron:** Para crear o editar la tabla de cron del usuario actual, ejecuta el siguiente comando:

**crontab -e**

Esto abrirá un editor de texto (generalmente nano o vi). Cada línea en este archivo representa un trabajo de cron con la sintaxis que vimos anteriormente. Después de

agregar o modificar los trabajos, guarda el archivo y cierra el editor. El nuevo crontab se activará automáticamente.

Si es la primera vez que editamos el crontab, preguntará qué editor quieres usar.

```
salva@salva-ubuntu: ~  
salva@salva-ubuntu:~$ crontab -l  
no crontab for salva  
salva@salva-ubuntu:~$ crontab -e  
no crontab for salva - using an empty one  
  
Select an editor. To change later, run 'select-editor'.  
 1. /bin/nano          <---- easiest  
 2. /usr/bin/vim.basic  
 3. /usr/bin/vim.tiny  
 4. /bin/ed  
  
Choose 1-4 [1]:  
No modification made
```

```
GNU nano 6.2 /tmp/crontab.Fqz73o/crontab  
# 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/  
#  
[ 23 líneas leídas ]  
^G Ayuda      ^O Guardar    ^W Buscar     ^K Cortar     ^T Ejecutar   ^C Ubicación  
^X Salir      ^R Leer fich. ^\ Reemplazar  ^U Pegar      ^J Justificar ^/ Ir a línea
```

- **Eliminar todos los trabajos de cron:** Para eliminar todos los trabajos de cron programados para el usuario actual, ejecutamos el siguiente comando:

**crontab -r**

Este comando eliminará todos tus trabajos de cron sin preguntar por confirmación.

## 1.4 ¿Cómo se crean scripts en Bash y cómo se ejecutan desde cron?

### Creación de scripts en Bash:

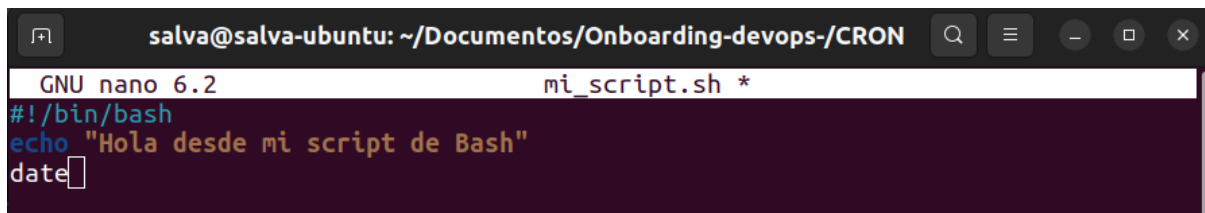
Un script en Bash es un archivo de texto que contiene una serie de comandos que se ejecutan secuencialmente por el intérprete de comandos Bash.

1. **Creamos un archivo de texto:** Abrimos un editor de texto (como nano, vim, gedit, etc.) y creamos un nuevo archivo. Por convención, los scripts de Bash suelen tener la extensión **.sh**. Por ejemplo, podríamos llamar al script **mi\_script.sh**.
2. **Agrega el shebang:** La primera línea de un script de Bash generalmente es el "shebang", que indica al sistema qué intérprete debe usar para ejecutar el script. Para Bash, esta línea es:

```
#!/bin/bash
```

3. **Escribir los comandos:** A partir de la segunda línea, escribimos los comandos que queremos que el script ejecute. Por ejemplo:

```
#!/bin/bash
echo "Hola desde mi script de Bash"
date
```



```
salva@salva-ubuntu: ~/Documentos/Onboarding-devops-/CRON
GNU nano 6.2 mi_script.sh *
#!/bin/bash
echo "Hola desde mi script de Bash"
date
```

4. Guardamos el archivo con la extensión **.sh**.
5. **Hacer el script ejecutable:** Para poder ejecutar el script, necesitamos darle permisos de ejecución. Abrimos la terminal, navegamos al directorio donde guardamos el script y ejecutamos el siguiente comando:

```
chmod +x mi_script.sh
```

Esto cambia los permisos del archivo para que pueda ejecutarlo.

### Ejecución de scripts desde cron:

Para ejecutar un script de Bash desde cron, lo incluimos como el comando en una entrada de la crontab. Es importante usar la ruta completa al script para que cron pueda encontrarlo.

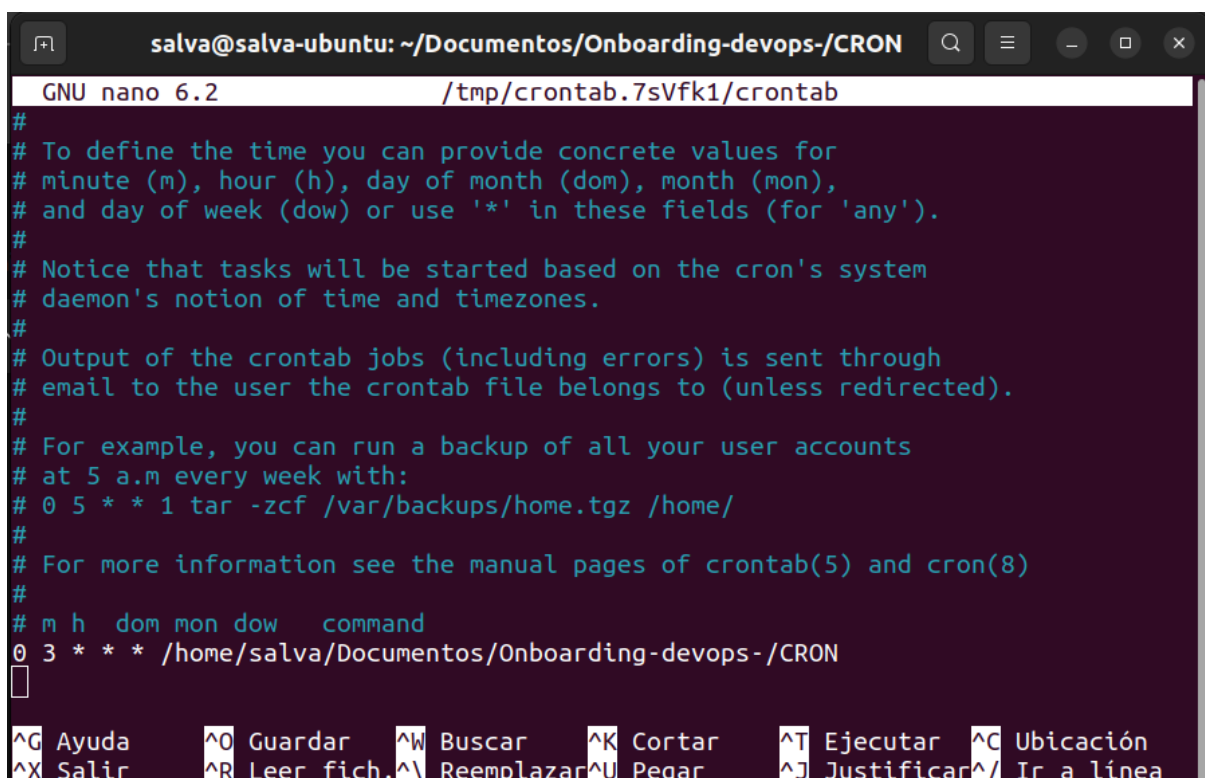
1. **Obtener la ruta completa al script:** Abro la terminal, navego al directorio donde guardé el script y ejecuto el siguiente comando:

**pwd**

Esto mostrará la ruta completa al directorio actual.

2. Ejecuto **crontab -e**.
3. **Agregar una entrada para ejecutar tu script:** Por ejemplo, para ejecutar el script todos los días a las 3:00 AM, agrego la siguiente línea:

**0 3 \* \* \* /home/salva/Documentos/Onboarding-devops-/CRON**



```
salva@salva-ubuntu: ~/Documentos/Onboarding-devops-/CRON
GNU nano 6.2 /tmp/crontab.7sVfk1/crontab
#
# 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 3 * * * /home/salva/Documentos/Onboarding-devops-/CRON
^G Ayuda  ^O Guardar  ^W Buscar  ^K Cortar  ^T Ejecutar  ^C Ubicación
^X Salir  ^R Leer fich. ^\ Reemplazar ^U Pegar    ^J Justificar ^_ Ir a línea
```

## Tarea 2: Automatización de Limpieza de Archivos y Logs

**2.1 Crea un script en Bash que elimine automáticamente archivos temporales en /tmp que tengan más de 7 días.**

Creo un archivo llamado `limpiar_tmp.sh` con el siguiente contenido:



```
salva@salva-ubuntu: ~/Documentos/Onboarding-devops-/CRON
GNU nano 6.2 limpiar_tmp.sh
#!/bin/bash

# Define el directorio a limpiar
directorio="/tmp"

# Define la antigüedad en días
antigüedad=7

# Encuentra y elimina los archivos con la antigüedad especificada
find "$directorio" -type f -mtime +"$antigüedad" -delete

# Registra la acción en un archivo de logs
fecha=$(date +%Y-%m-%d_%H:%M:%S)
echo "[$fecha] Se eliminaron archivos temporales con más de $antigüedad días en"

exit 0
```

[ 16 líneas escritas ]

<b>^G</b> Ayuda	<b>^O</b> Guardar	<b>^W</b> Buscar	<b>^K</b> Cortar	<b>^T</b> Ejecutar	<b>^C</b> Ubicación
<b>^X</b> Salir	<b>^R</b> Leer fich.	<b>^L</b> Reemplazar	<b>^U</b> Pegar	<b>^J</b> Justificar	<b>^_</b> Ir a línea

### Explicación del script:

- `#!/bin/bash`: Shebang para indicar que se ejecute con Bash.
- `directorio="/tmp"`: Define la variable `directorio` con la ruta `/tmp`.
- `antigüedad=7`: Define la variable `antigüedad` con el valor 7 (días).
- `find "$directorio" -type f -mtime +"$antigüedad" -delete`: Este es el comando clave:
  - `find "$directorio"`: Busca dentro del directorio especificado.
  - `-type f`: Busca solo archivos regulares (no directorios).
  - `-mtime +"$antigüedad"`: Busca archivos cuya última modificación fue hace más de (el signo +) el número de días especificado en la variable `antigüedad`.
  - `-delete`: Elimina los archivos encontrados. **Ten cuidado al usar este comando! Asegúrate de que la búsqueda sea correcta antes de eliminar.**
- `fecha=$(date +%Y-%m-%d_%H:%M:%S)`: Obtiene la fecha y hora actual en un formato específico y la guarda en la variable `fecha`.
- `echo "[$fecha] Se eliminaron archivos temporales con más de $antigüedad días en $directorio." >> /var/log/limpieza_tmp.log`: Escribe un mensaje de log en el archivo `/var/log/limpieza_tmp.log`.

indicando la fecha, hora y la acción realizada. El >> añade la línea al final del archivo.

- `exit 0`: Indica que el script se ejecutó correctamente.

## 2.2 Programa una tarea en cron que ejecute este script una vez al día.

Hacemos el script ejecutable:

```
chmod +x limpiar_tmp.sh
```

Edito crontab:

```
crontab -e
```

Agrego la siguiente línea para ejecutar el script todos los días a las 3:30 AM.

```
30 3 * * * /home/usuario/scripts/limpiar_tmp.sh
```

Esto ejecutará el script a las 03:30 de cada día.



```
salva@salva-ubuntu: ~/Documentos/Onboarding-devops-/CRON
GNU nano 6.2 /tmp/crontab.hD8opC/crontab
# 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 3 * * * /home/salva/Documentos/Onboarding-devops-/CRON
30 3 * * * /home/salva/Documentos/Onboarding-devops-/CRON
[ 25 líneas escritas ]
^G Ayuda  ^O Guardar  ^W Buscar  ^K Cortar  ^T Ejecutar  ^C Ubicación
^X Salir  ^R Leer fich. ^\ Reemplazar ^U Pegar  ^J Justificar ^_ Ir a línea
```

He probado a ejecutar el script manualmente, pero me da un error porque no tengo permisos. Voy a ejecutar:

```
sudo chown $USER /var/log/limpieza_tmp.log
```

Como todavía no se ha ejecutado nunca el script, el archivo no existe, así que voy a crearlo para poder darle los permisos.

```
sudo touch /var/log/limpieza_tmp.log
```

Ejecuto el script de limpieza otra vez:

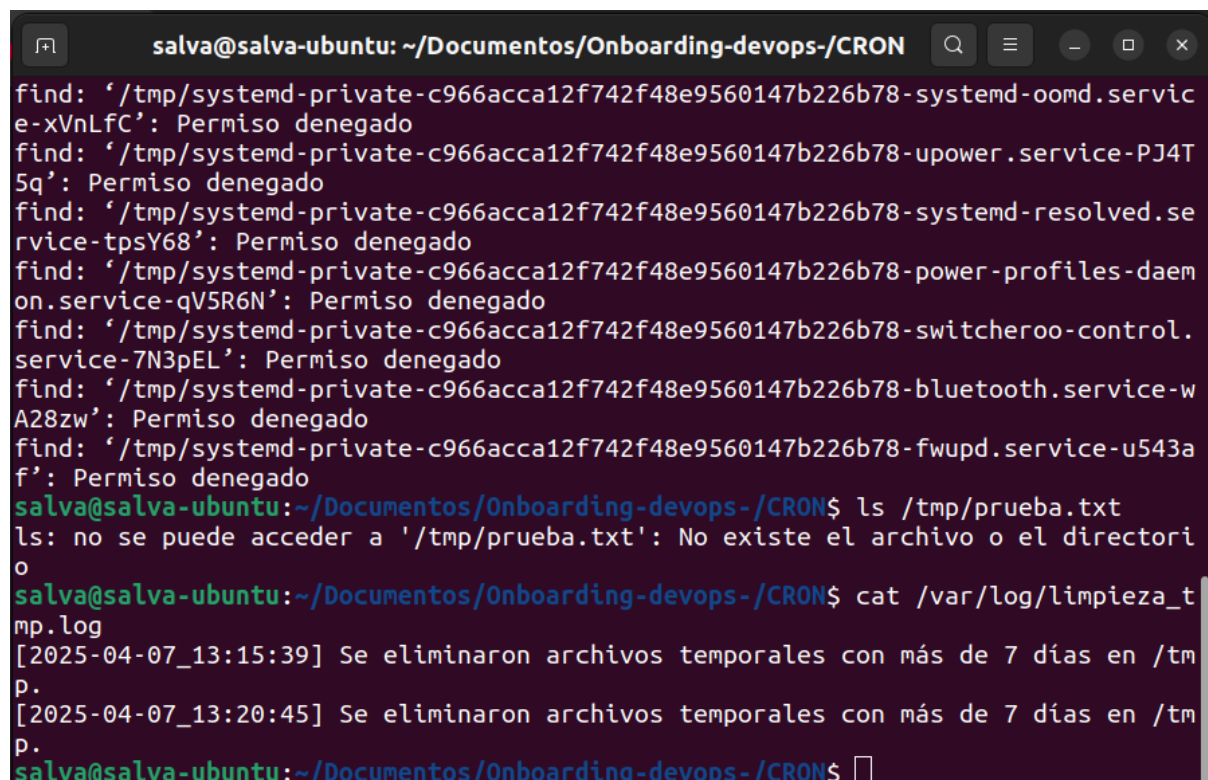
```
/home/salva/Documentos/Onboarding-devops-/CRON/limpiar_tmp.sh
```

Después de ejecutarlo, verifico si el archivo prueba.txt todavía existe:

```
ls /tmp/prueba.txt
```

Finalmente, reviso el archivo de log nuevamente para ver si se ha añadido una nueva entrada correspondiente a esta última ejecución del script:

```
cat /var/log/limpieza_tmp.log
```



```
salva@salva-ubuntu: ~/Documentos/Onboarding-devops-/CRON
find: '/tmp/systemd-private-c966acca12f742f48e9560147b226b78-systemd-oomd.servic
e-xVnLfc': Permiso denegado
find: '/tmp/systemd-private-c966acca12f742f48e9560147b226b78-upower.service-PJ4T
5q': Permiso denegado
find: '/tmp/systemd-private-c966acca12f742f48e9560147b226b78-systemd-resolved.se
rvice-tpsY68': Permiso denegado
find: '/tmp/systemd-private-c966acca12f742f48e9560147b226b78-power-profiles-daem
on.service-qV5R6N': Permiso denegado
find: '/tmp/systemd-private-c966acca12f742f48e9560147b226b78-switcheroo-control.
service-7N3pEL': Permiso denegado
find: '/tmp/systemd-private-c966acca12f742f48e9560147b226b78-bluetooth.service-w
A28zw': Permiso denegado
find: '/tmp/systemd-private-c966acca12f742f48e9560147b226b78-fwupd.service-u543a
f': Permiso denegado
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$ ls /tmp/prueba.txt
ls: no se puede acceder a '/tmp/prueba.txt': No existe el archivo o el directori
o
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$ cat /var/log/limpieza_t
mp.log
[2025-04-07_13:15:39] Se eliminaron archivos temporales con más de 7 días en /tm
p.
[2025-04-07_13:20:45] Se eliminaron archivos temporales con más de 7 días en /tm
p.
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$
```



## Tarea 3: Programación de Backups Automáticos

### 3.1 Crea un script que realice un backup de un directorio específico y lo comprima en formato .tar.gz.

Creo un nuevo archivo llamado **backup\_cron\_dir.sh** con el siguiente contenido:

```
#!/bin/bash

# Directorio a respaldar
directorio_a_respalidar="/home/salva/Documentos/Onboarding-devops-
/CRON/"

# Directorio donde guardar los backups
directorio_backup="/home/salva/backups/"

# Asegurarse de que el directorio de backup exista
mkdir -p "$directorio_backup"

# Nombre base del archivo de backup
nombre_base="backup_cron_dir"

# Fecha para el nombre del archivo
fecha=$(date +%Y-%m-%d_%H:%M:%S)

# Nombre completo del archivo de backup
nombre_archivo="$directorio_backup/$nombre_base-$fecha.tar.gz"

# Realizar el backup y comprimir
tar -czvf "$nombre_archivo" "$directorio_a_respalidar"

# Registrar la acción en un archivo de logs
fecha_log=$(date +%Y-%m-%d_%H:%M:%S)
echo "[$fecha_log] Se creó un backup de $directorio_a_respalidar en
$nombre_archivo." >> /var/log/backup_cron.log

exit 0
```

**Explicación del script:**

- `#!/bin/bash`: Shebang para Bash.
- `directorio_a_respalidar="/home/salva/Documentos/Onboarding-devops-/CRON/"`: Define el directorio que vamos a respaldar.
- `directorio_backup="/home/salva/backups/"`: Define el directorio donde se guardarán los archivos de backup. He creado un directorio llamado backups dentro de tu directorio personal (/home/salva/).
- `mkdir -p "$directorio_backup"`: Este comando se asegura de que el directorio de backup exista. Si no existe, lo crea. La opción `-p` crea los directorios padre si es necesario y no muestra un error si el directorio ya existe.
- `nombre_base="backup_cron_dir"`: Define un nombre base para los archivos de backup.
- `fecha=$(date +%Y-%m-%d_%H:%M:%S)`: Obtiene la fecha y hora para incluirla en el nombre del archivo.
- `nombre_archivo="$directorio_backup/$nombre_base-$fecha.tar.gz"`: Construye el nombre completo del archivo de backup.
- `tar -czvf "$nombre_archivo" "$directorio_a_respalidar"`: Ejecuta el comando tar para crear el backup comprimido:
  - `-c`: Crea un archivo.
  - `-z`: Comprime el archivo con gzip.
  - `-v`: Muestra los archivos que se están procesando (opcional).
  - `-f "$nombre_archivo"`: Especifica el nombre del archivo a crear.
  - `"$directorio_a_respalidar"`: Especifica el directorio que se va a incluir en el backup.
- `fecha_log=$(date +%Y-%m-%d_%H:%M:%S)`: Obtiene la fecha y hora para el log.
- `echo "[$fecha_log] Se creó un backup de $directorio_a_respalidar en $nombre_archivo." >> /var/log/backup_cron.log`: Escribe un mensaje de log en el archivo /var/log/backup\_cron.log.
- `exit 0`: Indica que el script se ejecutó correctamente.

```
salva@salva-ubuntu: ~/Documentos/Onboarding-devops-/CRON
GNU nano 6.2 backup_cron_dir.sh
mkdir -p "$directorio_backup"

# Nombre base del archivo de backup
nombre_base="backup_cron_dir"

# Fecha para el nombre del archivo
fecha=$(date +%Y-%m-%d_%H:%M:%S)

# Nombre completo del archivo de backup
nombre_archivo="$directorio_backup/$nombre_base-$fecha.tar.gz"

# Realizar el backup y comprimir
tar -czvf "$nombre_archivo" "$directorio_a_respalidar"

# Registrar la acción en un archivo de logs
fecha_log=$(date +%Y-%m-%d_%H:%M:%S)
echo "[$fecha_log] Se creó un backup de $directorio_a_respalidar en $nombre_archivo"

exit 0
```

[ 28 líneas escritas ]

^G Ayuda	^O Guardar	^W Buscar	^K Cortar	^T Ejecutar	^C Ubicación
^X Salir	^R Leer fich.	^_ Reemplazar	^U Pegar	^J Justificar	^_ Ir a línea

### 3.2 Configura cron para ejecutar este script automáticamente cada día a las 02:00 AM.

1. **Hago el script ejecutable:** Abro la terminal y ejecuto el siguiente comando para darle permisos de ejecución al script `backup_cron_dir.sh`:

```
chmod +x /home/salva/Documentos/Onboarding-devops-  
/CRON/backup_cron_dir.sh
```

2. **Edito crontab:** Ejecuto el siguiente comando para abrir el archivo de configuración de cron:

```
crontab -e
```

3. **Agrego la entrada para ejecutar el script:** Añado la siguiente línea al final del archivo crontab. Esta línea programará la ejecución del script todos los días a las 02:00 AM:

```
0 2 * * * /home/salva/Documentos/Onboarding-devops-  
/CRON/backup_cron_dir.sh
```

- a. 0: Representa el minuto 0 de la hora.

- b. 2: Representa la hora 2 (en formato de 24 horas, que es las 02:00 AM).
- c. \*: Representa cualquier día del mes.
- d. \*: Representa cualquier mes del año.
- e. \*: Representa cualquier día de la semana.
- f. /home/salva/Documentos/Onboarding-devops-  
/CRON/backup\_cron\_dir.sh: Es la ruta completa al script que creaste.

```
salva@salva-ubuntu: ~/Documentos/Onboarding-devops-/CRON
GNU nano 6.2 /tmp/crontab.xFWxEO/crontab
# 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 3 * * * /home/salva/Documentos/Onboarding-devops-/CRON
30 3 * * * /home/salva/Documentos/Onboarding-devops-/CRON
0 2 * * * /home/salva/Documentos/Onboarding-devops-/CRON/backup_cron_dir.sh
[ 26 líneas escritas ]
^G Ayuda  ^O Guardar  ^W Buscar  ^K Cortar  ^T Ejecutar  ^C Ubicación
^X Salir  ^R Leer fich. ^\ Reemplazar ^U Pegar  ^J Justificar ^_ Ir a línea
```

**3.3 Asegúrate de que el script almacene un registro de las copias de seguridad en un archivo de log.**

**Ejecuto el script manualmente** para probarlo ahora mismo:

**/home/salva/Documentos/Onboarding-devops-/CRON/backup\_cron\_dir.sh**

Como me ocurrió antes, el archivo no está creado y no tengo permisos. Así que voy a crearlo y a darle los permisos.

```
sudo touch /var/log/backup_cron.log
sudo chown $USER /var/log/backup_cron.log
```

```
salva@salva-ubuntu: ~/Documentos/Onboarding-devops-/CRON
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$ nano
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$ chmod +x /home/salva/Documentos/Onboarding-devops-/CRON/backup_cron_dir.sh

salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$ crontab -e
crontab: installing new crontab
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$ /home/salva/Documentos/Onboarding-devops-/CRON/backup_cron_dir.sh
tar: Eliminando la '/' inicial de los nombres
/home/salva/Documentos/Onboarding-devops-/CRON/
/home/salva/Documentos/Onboarding-devops-/CRON/limpiar_tmp.sh
/home/salva/Documentos/Onboarding-devops-/CRON/mi_script.sh
/home/salva/Documentos/Onboarding-devops-/CRON/backup_cron_dir.sh
/home/salva/Documentos/Onboarding-devops-/CRON/readme.md
/home/salva/Documentos/Onboarding-devops-/CRON/backup_cron_dir.sh: línea 26: /var/log/backup_cron.log: Permiso denegado
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$ sudo touch /var/log/backup_cron.log
sudo chown $USER /var/log/backup_cron.log
[sudo] contraseña para salva:
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$
```

Ejecuto el script de backup nuevamente:

`/home/salva/Documentos/Onboarding-devops-/CRON/backup_cron_dir.sh`

```
salva@salva-ubuntu: ~/Documentos/Onboarding-devops-/CRON
crontab: installing new crontab
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$ /home/salva/Documentos/Onboarding-devops-/CRON/backup_cron_dir.sh
tar: Eliminando la '/' inicial de los nombres
/home/salva/Documentos/Onboarding-devops-/CRON/
/home/salva/Documentos/Onboarding-devops-/CRON/limpiar_tmp.sh
/home/salva/Documentos/Onboarding-devops-/CRON/mi_script.sh
/home/salva/Documentos/Onboarding-devops-/CRON/backup_cron_dir.sh
/home/salva/Documentos/Onboarding-devops-/CRON/readme.md
/home/salva/Documentos/Onboarding-devops-/CRON/backup_cron_dir.sh: línea 26: /var/log/backup_cron.log: Permiso denegado
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$ sudo touch /var/log/backup_cron.log
sudo chown $USER /var/log/backup_cron.log
[sudo] contraseña para salva:
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$ /home/salva/Documentos/Onboarding-devops-/CRON/backup_cron_dir.sh
tar: Eliminando la '/' inicial de los nombres
/home/salva/Documentos/Onboarding-devops-/CRON/
/home/salva/Documentos/Onboarding-devops-/CRON/limpiar_tmp.sh
/home/salva/Documentos/Onboarding-devops-/CRON/mi_script.sh
/home/salva/Documentos/Onboarding-devops-/CRON/backup_cron_dir.sh
/home/salva/Documentos/Onboarding-devops-/CRON/readme.md
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$
```

Ahora sí se ejecuta correctamente.

Verifico el contenido del archivo `/var/log/backup_cron.log` para asegurarme de que se haya registrado la creación del backup:

```
cat /var/log/backup_cron.log
```

```
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$ cat /var/log/backup_cron.log
[2025-04-07_13:55:36] Se creó un backup de /home/salva/Documentos/Onboarding-devops-/CRON/ en /home/salva/backups//backup_cron_dir-2025-04-07_13:55:36.tar.gz.
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$
```

## Tarea 4: Reinicio de Servicios Automático

### 4.1 Escribe un script que verifique si un servicio (por ejemplo, nginx o apache2) está en ejecución y lo reinicie si está caído.

Creo un nuevo archivo llamado `verificar_nginx.sh`

```
#!/bin/bash

# Nombre del servicio a verificar
servicio="nginx"

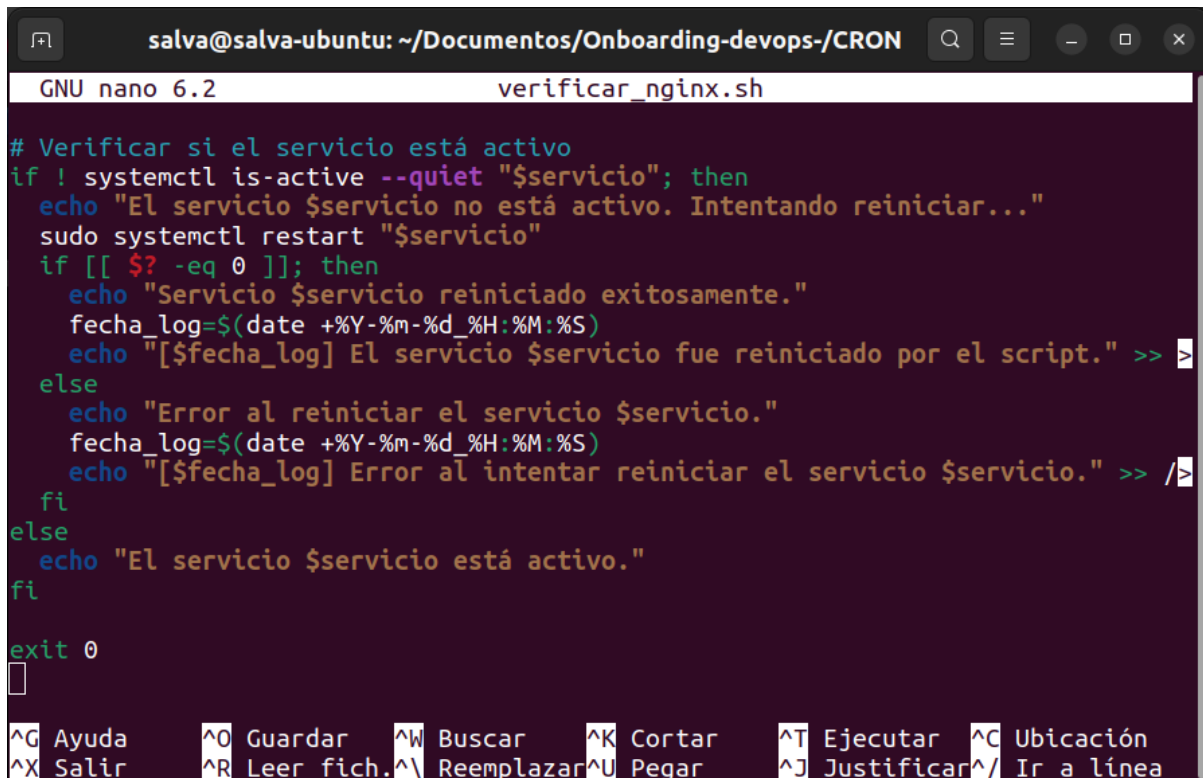
# Verificar si el servicio está activo
if ! systemctl is-active --quiet "$servicio"; then
    echo "El servicio $servicio no está activo. Intentando reiniciar..."
    sudo systemctl restart "$servicio"
    if [[ $? -eq 0 ]]; then
        echo "Servicio $servicio reiniciado exitosamente."
        fecha_log=$(date +%Y-%m-%d_%H:%M:%S)
        echo "[$fecha_log] El servicio $servicio fue reiniciado por el script." >> /var/log/reinicio_nginx.log
    else
        echo "Error al reiniciar el servicio $servicio."
        fecha_log=$(date +%Y-%m-%d_%H:%M:%S)
        echo "[$fecha_log] Error al intentar reiniciar el servicio $servicio." >> /var/log/reinicio_nginx.log
    fi
else
    echo "El servicio $servicio está activo."
```

fi

exit 0

### Explicación del script:

- `#!/bin/bash`: Shebang para Bash.
- `servicio="nginx"`: Define la variable `servicio` con el nombre del servicio a verificar (nginx).
- `if ! systemctl is-active --quiet "$servicio"; then`: Utiliza `systemctl is-active --quiet` para verificar si el servicio está activo. La opción `--quiet` no produce salida, y el `!` niega el resultado, por lo que el bloque `then` se ejecuta si el servicio *no* está activo.
- `echo "El servicio $servicio no está activo. Intentando reiniciar..."`: Muestra un mensaje si el servicio no está activo.
- `sudo systemctl restart "$servicio"`: Intenta reiniciar el servicio usando `sudo`.
- `if [[ $? -eq 0 ]]`: Verifica si el comando anterior (`sudo systemctl restart`) se ejecutó con éxito (código de salida 0).
- Los bloques `echo` y `fecha_log` registran la acción en el archivo `/var/log/reinicio_nginx.log`.
- `else`: Si el servicio ya estaba activo, se muestra un mensaje.
- `exit 0`: Indica éxito.



```
salva@salva-ubuntu: ~/Documentos/Onboarding-devops-/CRON
GNU nano 6.2                                verificar_nginx.sh

# Verificar si el servicio está activo
if ! systemctl is-active --quiet "$servicio"; then
    echo "El servicio $servicio no está activo. Intentando reiniciar..."
    sudo systemctl restart "$servicio"
    if [[ $? -eq 0 ]]; then
        echo "Servicio $servicio reiniciado exitosamente."
        fecha_log=$(date +%Y-%m-%d_%H:%M:%S)
        echo "[$fecha_log] El servicio $servicio fue reiniciado por el script." >> /dev/null
    else
        echo "Error al reiniciar el servicio $servicio."
        fecha_log=$(date +%Y-%m-%d_%H:%M:%S)
        echo "[$fecha_log] Error al intentar reiniciar el servicio $servicio." >> /dev/null
    fi
else
    echo "El servicio $servicio está activo."
fi

exit 0

^G Ayuda      ^O Guardar    ^W Buscar     ^K Cortar     ^T Ejecutar   ^C Ubicación
^X Salir      ^R Leer fich. ^\ Reemplazar  ^U Pegar      ^J Justificar ^/ Ir a línea
```

#### 4.2 Programa una tarea en cron para ejecutar este script cada 5 minutos.

Hago el script ejecutable:

```
chmod +x /home/salva/Documentos/Onboarding-devops-  
/CRON/verificar_nginx.sh
```

Edito crontab:

```
crontab -e
```

Agrego la siguiente línea para ejecutar el script cada 5 minutos:

```
*/5 * * * * /home/salva/Documentos/Onboarding-devops-  
/CRON/verificar_nginx.sh
```

\*/5 en el campo de los minutos significa que el script se ejecutará en los minutos 0, 5, 10, 15, etc., de cada hora.



```
salva@salva-ubuntu: ~/Documentos/Onboarding-devops-/CRON
GNU nano 6.2 /tmp/crontab.Ujt0dq/crontab
# 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 3 * * * /home/salva/Documentos/Onboarding-devops-/CRON
30 3 * * * /home/salva/Documentos/Onboarding-devops-/CRON
0 2 * * * /home/salva/Documentos/Onboarding-devops-/CRON/backup_cron_dir.sh
*/5 * * * * /home/salva/Documentos/Onboarding-devops-/CRON/verificar_nginx.sh
[ 27 líneas escritas ]
^G Ayuda      ^O Guardar    ^W Buscar     ^K Cortar     ^T Ejecutar    ^C Ubicación
^X Salir      ^R Leer fich. ^\ Reemplazar  ^U Pegar      ^J Justificar ^/ Ir a línea
```

4.3 Prueba el script deteniendo el servicio manualmente y verifica que cron lo reinicia.

```
sudo systemctl start nginx
sudo systemctl status nginx
```

```
salva@salva-ubuntu: ~/Documentos/Onboarding-devops-/CRON
sudo systemctl status nginx
[sudo] contraseña para salva:
● nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/lib/systemd/system/nginx.service; disabled; vendor preset: enabled)
   Active: active (running) since Mon 2025-04-07 14:24:41 CEST; 34ms ago
     Docs: man:nginx(8)
  Process: 20788 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=0, status=exited,
 Process: 20789 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=0, status=exited,
 Main PID: 20790 (nginx)
    Tasks: 7 (limit: 14181)
   Memory: 8.6M
      CPU: 67ms
   CGroup: /system.slice/nginx.service
           └─20790 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
             └─20791 "nginx: worker process"
               └─20792 "nginx: worker process"
                 └─20793 "nginx: worker process"
                   └─20794 "nginx: worker process"
                     └─20795 "nginx: worker process"
                       └─20797 "nginx: worker process"

abr 07 14:24:41 salva-ubuntu systemd[1]: Starting A high performance web server: nginx.service.
abr 07 14:24:41 salva-ubuntu systemd[1]: Started A high performance web server: nginx.service.
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$
```

**Detengo el servicio manualmente:**

**sudo systemctl stop nginx**

```
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$ sudo systemctl status nginx
○ nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/lib/systemd/system/nginx.service; disabled; vendor preset: enabled)
   Active: inactive (dead)
     Docs: man:nginx(8)

abr 07 14:24:41 salva-ubuntu systemd[1]: Starting A high performance web server: nginx.service.
abr 07 14:24:41 salva-ubuntu systemd[1]: Started A high performance web server: nginx.service.
abr 07 14:26:25 salva-ubuntu systemd[1]: Stopping A high performance web server: nginx.service.
abr 07 14:26:25 salva-ubuntu systemd[1]: nginx.service: Deactivated successfully.
abr 07 14:26:25 salva-ubuntu systemd[1]: Stopped A high performance web server: nginx.service.
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$
```

**Espero 5 minutos.** La tarea de cron debería ejecutarse dentro de este tiempo.

**Verificar si el servicio se ha reiniciado automáticamente:**

**sudo systemctl status nginx**

```
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$ sudo systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/lib/systemd/system/nginx.service; disabled; vendor preset: enabled)
   Active: active (running) since Mon 2025-04-07 14:38:48 CEST; 18s ago
     Docs: man:nginx(8)
  Process: 21214 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=0)
  Process: 21215 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=0)
 Main PID: 21216 (nginx)
    Tasks: 7 (limit: 14181)
   Memory: 6.3M
      CPU: 53ms
   CGroup: /system.slice/nginx.service
           └─21216 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
           └─21217 "nginx: worker process" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" ""
```

Vemos que el estado del servicio ha vuelto a ser active (running).

**Reviso el archivo de log:** Verifico el archivo `/var/log/reinicio_nginx.log` para ver si hay entradas indicando que el script detectó que el servicio estaba inactivo y lo reinició. Creo este archivo de log con los permisos correctos, de la misma manera que hice con los logs anteriores:

```
sudo touch /var/log/reinicio_nginx.log
sudo chown $USER /var/log/reinicio_nginx.log
```

Reviso el archivo de log para ver si el script registró su ejecución y el intento de reinicio:

```
cat /var/log/reinicio_nginx.log
```

```
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$ cat /var/log/reinicio_nginx.log
[2025-04-07_14:38:48] El servicio nginx fue reiniciado por el script.
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$
```

El comando **sudo systemctl restart nginx** dentro del script me está fallando porque cron no puede proporcionar una contraseña para sudo.

```
salva@salva-ubuntu: ~/Documentos/Onboarding-devo...
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$ cat /home/salva/Documentos/Onboarding-devops-/CRON/cron_nginx.log
El servicio nginx no está activo. Intentando reiniciar...
sudo: a terminal is required to read the password; either use the -S option to read from standard input or configure an askpass helper
sudo: se requiere una contraseña
Error al reiniciar el servicio nginx.
El servicio nginx no está activo. Intentando reiniciar...
sudo: a terminal is required to read the password; either use the -S option to read from standard input or configure an askpass helper
sudo: se requiere una contraseña
Error al reiniciar el servicio nginx.
El servicio nginx no está activo. Intentando reiniciar...
sudo: a terminal is required to read the password; either use the -S option to read from standard input or configure an askpass helper
sudo: se requiere una contraseña
Error al reiniciar el servicio nginx.
El servicio nginx no está activo. Intentando reiniciar...
sudo: a terminal is required to read the password; either use the -S option to read from standard input or configure an askpass helper
sudo: se requiere una contraseña
Error al reiniciar el servicio nginx.
El servicio nginx no está activo. Intentando reiniciar...
sudo: a terminal is required to read the password; either use the -S option to read from standard input or configure an askpass helper
sudo: se requiere una contraseña
Error al reiniciar el servicio nginx.
El servicio nginx no está activo. Intentando reiniciar...
sudo: a terminal is required to read the password; either use the -S option to read from standard input or configure an askpass helper
sudo: se requiere una contraseña
Error al reiniciar el servicio nginx.
```

Así que voy a ejecutar la tarea de cron como el usuario root. Esto evitará la necesidad de usar sudo dentro del script.

**sudo crontab -e**

Y agrego la línea para el script.

```
salva@salva-ubuntu: ~/Documentos/Onboarding-devo...
GNU nano 6.2 /tmp/crontab.7NOFVn/crontab
#
# 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
*/5 * * * * /home/salva/Documentos/Onboarding-devops-/CRON/verificar_>
[ 25 líneas escritas ]
^G Ayuda      ^O Guardar    ^W Buscar     ^K Cortar     ^T Ejecutar
^X Salir      ^R Leer fich. ^\ Reemplazar ^U Pegar      ^J Justificar
```

Ahora sí, Nginx se activa correctamente.

```
salva@salva-ubuntu: ~/Documentos/Onboarding-devo...
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$ sudo systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/lib/systemd/system/nginx.service; disabled; vendor preset: enabled)
   Active: active (running) since Mon 2025-04-07 16:20:01 CEST; 1min 1s ago
     Docs: man:nginx(8)
   Process: 23338 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on;
   Process: 23339 ExecStart=/usr/sbin/nginx -g daemon on; master_process on;
  Main PID: 23340 (nginx)
    Tasks: 7 (limit: 14181)
   Memory: 6.3M
      CPU: 52ms
   CGroup: /system.slice/nginx.service
           └─23340 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
              └─23341 "nginx: worker process"
                 └─23342 "nginx: worker process"
                    └─23343 "nginx: worker process"
                       └─23344 "nginx: worker process"
                          └─23345 "nginx: worker process"
                             └─23346 "nginx: worker process"

abr 07 16:20:01 salva-ubuntu systemd[1]: Starting A high performance web server: nginx.
abr 07 16:20:01 salva-ubuntu systemd[1]: Started A high performance web server: nginx.
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$
```

Elimino crontab de usuario y edito sudo crontab con todas las tareas para que no haya errores:

**crontab -r**

**sudo crontab -e**



```
salva@salva-ubuntu: ~/Documentos/Onboarding-devo...
GNU nano 6.2 /tmp/crontab.A5IBYc/crontab
# 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 3 * * * /home/salva/Documentos/Onboarding-devops-/CRON
30 3 * * * /home/salva/Documentos/Onboarding-devops-/CRON
0 2 * * * /home/salva/Documentos/Onboarding-devops-/CRON/backup_cron_>
*/5 * * * * /home/salva/Documentos/Onboarding-devops-/CRON/verificar_>
[ 28 líneas escritas ]
^G Ayuda      ^O Guardar    ^W Buscar     ^K Cortar     ^T Ejecutar
^X Salir      ^R Leer fich. ^\ Reemplazar ^U Pegar      ^J Justificar
```

```
salva@salva-ubuntu: ~/Documentos/Onboarding-devo...
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$ sudo cat /var
/log/reinicio_nginx.log
[2025-04-07_14:38:48] El servicio nginx fue reiniciado por el script.
[2025-04-07_14:50:01] Error al intentar reiniciar el servicio nginx.
[2025-04-07_14:55:01] Error al intentar reiniciar el servicio nginx.
[2025-04-07_15:00:01] Error al intentar reiniciar el servicio nginx.
[2025-04-07_15:05:01] Error al intentar reiniciar el servicio nginx.
[2025-04-07_15:10:01] Error al intentar reiniciar el servicio nginx.
[2025-04-07_15:15:02] Error al intentar reiniciar el servicio nginx.
[2025-04-07_15:20:01] Error al intentar reiniciar el servicio nginx.
[2025-04-07_15:25:01] Error al intentar reiniciar el servicio nginx.
[2025-04-07_15:30:01] Error al intentar reiniciar el servicio nginx.
[2025-04-07_15:35:01] Error al intentar reiniciar el servicio nginx.
[2025-04-07_15:40:01] Error al intentar reiniciar el servicio nginx.
[2025-04-07_15:45:01] Error al intentar reiniciar el servicio nginx.
[2025-04-07_15:50:01] Error al intentar reiniciar el servicio nginx.
[2025-04-07_15:55:01] Error al intentar reiniciar el servicio nginx.
[2025-04-07_16:00:01] Error al intentar reiniciar el servicio nginx.
[2025-04-07_16:05:01] Error al intentar reiniciar el servicio nginx.
[2025-04-07_16:10:01] Error al intentar reiniciar el servicio nginx.
[2025-04-07_16:15:01] Error al intentar reiniciar el servicio nginx.
[2025-04-07_16:20:01] Error al intentar reiniciar el servicio nginx.
[2025-04-07_16:20:01] El servicio nginx fue reiniciado por el script.
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$
```

## Tarea 5: Despliegue y Monitoreo de Contenedores con Cron y Docker

- **Explica cómo interactúan cron y Docker en sistemas Linux.**

**cron** es un planificador de tareas en sistemas tipo **Unix** como **Linux**. Permite ejecutar comandos o scripts automáticamente en momentos específicos o a intervalos regulares. Docker es una plataforma para desarrollar, enviar y ejecutar aplicaciones en contenedores.

La interacción entre **cron y Docker** se da cuando cron se utiliza para automatizar tareas relacionadas con la gestión de contenedores Docker. Por ejemplo, **cron** puede ejecutar un script que:

- Inicie o detenga contenedores en horarios específicos.
- Verifique el estado de los contenedores y realice acciones si es necesario (como reiniciar un contenedor que se ha detenido).
- Ejecute backups de volúmenes de Docker.
- Realice tareas de limpieza de contenedores o imágenes no utilizadas.

En esencia, **cron** actúa como el disparador para ejecutar comandos de Docker o scripts que interactúan con el daemon de Docker.

- **Investiga cómo se puede comprobar el estado de un contenedor con docker ps.**

El comando **docker ps** se utiliza para listar los contenedores que se están ejecutando actualmente. Algunas opciones útiles son:

- `docker ps`: Muestra solo los contenedores en ejecución.
- `docker ps -a`: Muestra todos los contenedores (en ejecución y detenidos).
- `docker ps -f "name=<nombre_del_contenedor>"`: Filtra los contenedores por nombre.
- `docker ps --format "{{.ID}}\t{{.Names}}\t{{.State}}\t{{.Status}}"`: Permite personalizar el formato de la salida, mostrando información como el ID, nombre, estado y estatus del contenedor.

El estado de un contenedor puede ser `running`, `restarting`, `exited`, `created`, etc. Para verificar si un contenedor está activo, generalmente buscamos el estado `running`.



- **Busca cómo reiniciar un contenedor si ha dejado de ejecutarse.**

Para reiniciar un contenedor que se ha detenido, se utiliza el comando `docker start <nombre_o_ID_del_contenedor>`. Si el contenedor ya se estaba ejecutando, este comando no tendrá ningún efecto.

También existe el comando `docker restart <nombre_o_ID_del_contenedor>`, que primero intenta detener el contenedor y luego lo inicia.

### Creación del entorno:

- **Usa Docker para crear y ejecutar un contenedor con un servicio web (por ejemplo, un contenedor de nginx).**

Ejecutar el siguiente comando para crear y ejecutar un contenedor nginx en segundo plano:

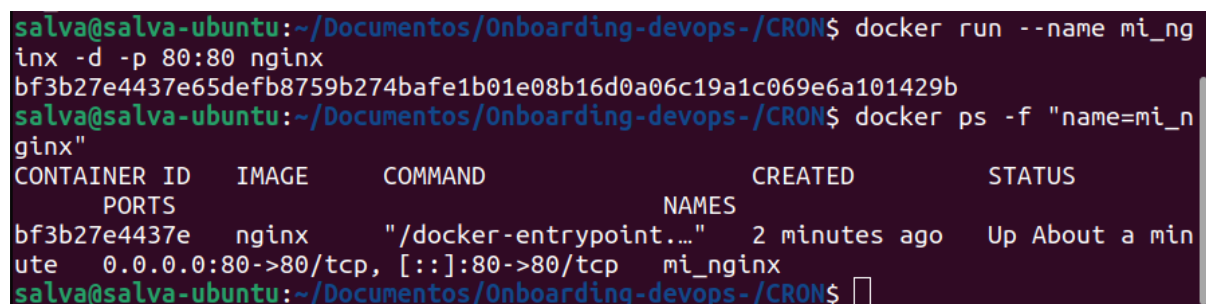
```
docker run --name mi_nginx -d -p 80:80 nginx
```

- `docker run`: Comando para ejecutar un nuevo contenedor.
- `--name mi_nginx`: Asigna el nombre `mi_nginx` a este contenedor. Es importante darle un nombre para poder referenciarlo fácilmente.
- `-d`: Ejecuta el contenedor en segundo plano (detached mode).
- `-p 80:80`: Mapea el puerto 80 del host (tu máquina) al puerto 80 del contenedor (donde nginx está escuchando).
- `nginx`: La imagen de Docker que se utilizará para crear el contenedor.

Después de ejecutar este comando, puedes verificar que el contenedor se está ejecutando con:

Bash

```
docker ps -f "name=mi_nginx"
```



```
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$ docker run --name mi_nginx -d -p 80:80 nginx
bf3b27e4437e65defb8759b274baf1b01e08b16d0a06c19a1c069e6a101429b
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$ docker ps -f "name=mi_nginx"
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS
bf3b27e4437e	nginx	"/docker-entrypoint..."	2 minutes ago	Up About a minute

```
0.0.0.0:80->80/tcp, [::]:80->80/tcp mi_nginx
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$
```

### Automatización con Cron:

## 1. Escribe un script en Bash que verifique si el contenedor sigue activo.

Creo un nuevo archivo llamado **monitorear\_nginx.sh** en el directorio `/home/salva/Documentos/Onboarding-devops-/CRON/` y pego el siguiente contenido:

```
#!/bin/bash

nombre_contenedor="mi_nginx"
log_file="/var/log/monitoreo_nginx.log"
fecha=$(date +%Y-%m-%d_%H:%M:%S)

# Verificar si el contenedor está en ejecución
if docker ps -q --filter "name=$nombre_contenedor" | grep -q .; then
    echo "[$fecha] El contenedor $nombre_contenedor está en ejecución." >> "$log_file"
else
    echo "[$fecha] El contenedor $nombre_contenedor NO está en ejecución. Intentando reiniciarlo..." >> "$log_file"
    docker start "$nombre_contenedor"
    if [[ $? -eq 0 ]]; then
        echo "[$fecha] El contenedor $nombre_contenedor ha sido reiniciado exitosamente." >> "$log_file"
    else
        echo "[$fecha] Error al intentar reiniciar el contenedor $nombre_contenedor." >> "$log_file"
    fi
fi

exit 0
```



```
salva@salva-ubuntu: ~/Documentos/Onboarding-devops-/CRON
GNU nano 6.2      monitorear_nginx.sh

nombre_contenedor="mi_nginx"
log_file="/var/log/monitoreo_nginx.log"
fecha=$(date +%Y-%m-%d_%H:%M:%S)

# Verificar si el contenedor está en ejecución
if docker ps -q --filter "name=$nombre_contenedor" | grep -q .; then
    echo "[$fecha] El contenedor $nombre_contenedor está en ejecución." >> "$log_>
else
    echo "[$fecha] El contenedor $nombre_contenedor NO está en ejecución. Intenta>
    docker start "$nombre_contenedor"
    if [[ $? -eq 0 ]]; then
        echo "[$fecha] El contenedor $nombre_contenedor ha sido reiniciado exitosam>
    else
        echo "[$fecha] Error al intentar reiniciar el contenedor $nombre_contenedor>
    fi
fi

exit 0

```

### Explicación del script:

- nombre\_contenedor="mi\_nginx": Define el nombre del contenedor que vamos a monitorear.
- log\_file="/var/log/monitoreo\_nginx.log": Define la ruta del archivo de log.
- fecha=\$(date +%Y-%m-%d\_%H:%M:%S): Obtiene la fecha y hora para el log.
- docker ps -q --filter "name=\$nombre\_contenedor": Lista solo los IDs de los contenedores con el nombre especificado.
- grep -q .: Verifica si la salida del comando anterior contiene alguna línea (es decir, si el contenedor existe y está en algún estado).
- El bloque if verifica si el contenedor está en ejecución. Si no lo está, intenta reiniciarlo con docker start "\$nombre\_contenedor".
- Se registra la actividad en el archivo de log /var/log/monitoreo\_nginx.log.

### 2. Hago el script ejecutable:

```
chmod +x /home/salva/Documentos/Onboarding-devops-  
/CRON/monitorear_nginx.sh
```

### 3. Configura cron para ejecutar este script cada 2 minutos.

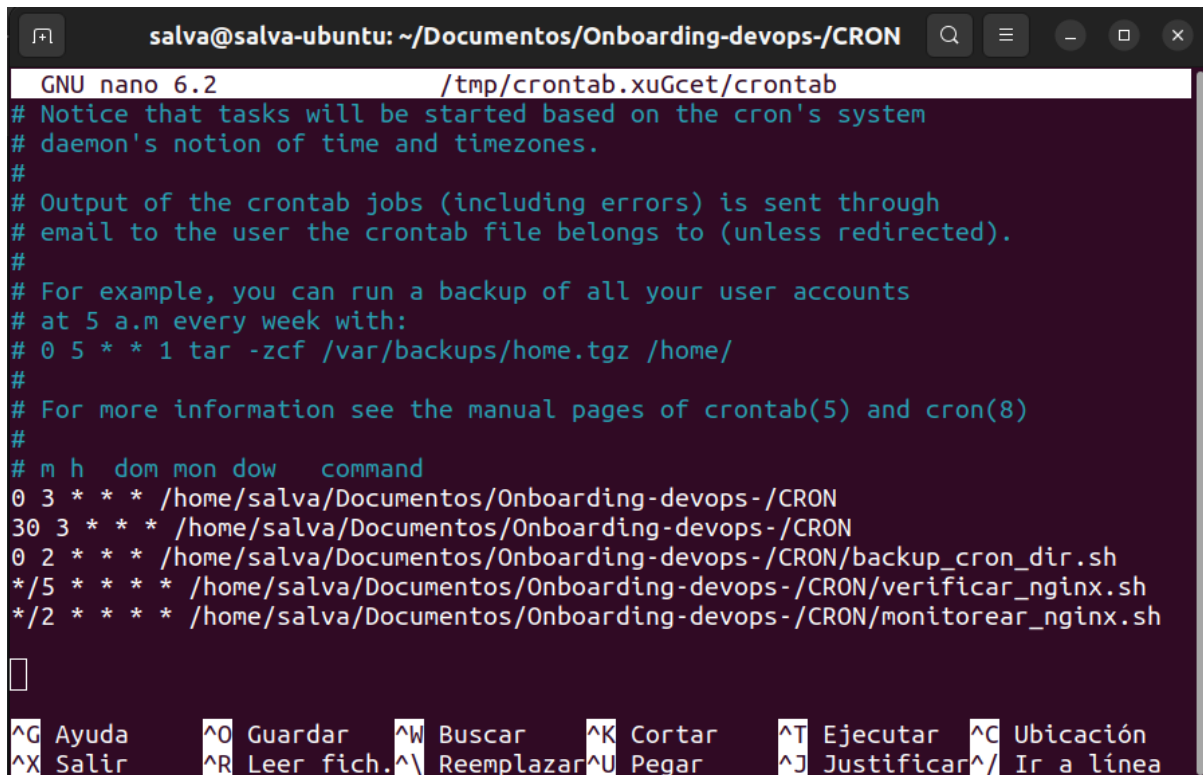
Edito la crontab del usuario root:

```
sudo crontab -e
```

Agrego la siguiente línea al final del archivo:

```
*/2 * * * * /home/salva/Documentos/Onboarding-devops-  
/CRON/monitorear_nginx.sh
```

Esto ejecutará el script `monitorear_nginx.sh` cada 2 minutos.



```
salva@salva-ubuntu: ~/Documentos/Onboarding-devops-/CRON
GNU nano 6.2 /tmp/crontab.xuGcet/crontab
# 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 3 * * * /home/salva/Documentos/Onboarding-devops-/CRON
30 3 * * * /home/salva/Documentos/Onboarding-devops-/CRON
0 2 * * * /home/salva/Documentos/Onboarding-devops-/CRON/backup_cron_dir.sh
*/5 * * * * /home/salva/Documentos/Onboarding-devops-/CRON/verificar_nginx.sh
*/2 * * * * /home/salva/Documentos/Onboarding-devops-/CRON/monitorear_nginx.sh

```

**Pruebas:**

#### 1. Detén el contenedor `mi_nginx` manualmente:

```
docker stop mi_nginx
```

2. **Espero al menos 3-4 minutos.** Esto asegurará que la tarea de cron se ejecute al menos una o dos veces.

### 3. Verifica el estado del contenedor mi\_nginx:

`docker ps -a`

```
salva@salva-ubuntu: ~/Documentos/Onboarding-devops-/CRON
NAME                  NAMES
bf3b27e4437e         nginx                "/docker-entrypoint..." 15 minutes ago
Exited (0) 16 seconds ago
e6801effa9a1         portainer/portainer-ce "/portainer"              6 days ago
Up 9 hours           0.0.0.0:8000->8000/tcp, [::]:8000->8000/tcp, 0.0.0.0:9443->9443/tcp, [::]:9443->9443/tcp, 9000/tcp
19ab297ad19d         hello-world          "/hello"                  6 days ago
Exited (0) 6 days ago
beautiful_jemison

salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$ docker ps -a
CONTAINER ID   IMAGE                                PORTS
STATUS        PORTS
bf3b27e4437e   nginx                                "/docker-entrypoint..." 17 minutes ago
Up About a minute 0.0.0.0:80->80/tcp, [::]:80->80/tcp
e6801effa9a1   portainer/portainer-ce             "/portainer"              6 days ago
Up 9 hours     0.0.0.0:8000->8000/tcp, [::]:8000->8000/tcp, 0.0.0.0:9443->9443/tcp, [::]:9443->9443/tcp, 9000/tcp
19ab297ad19d   hello-world                         "/hello"                  6 days ago
Exited (0) 6 days ago
beautiful_jemison

salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$
```

Después de la espera, veo que el contenedor `mi_nginx` aparece en la lista con un estado `Up` (en ejecución).

Reviso el archivo de log `/var/log/monitoreo_nginx.log`:

`cat /var/log/monitoreo_nginx.log`

```
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$ cat /var/log/monitoreo_nginx.log
[2025-04-07_17:30:01] El contenedor mi_nginx está en ejecución.
[2025-04-07_17:32:01] El contenedor mi_nginx está en ejecución.
[2025-04-07_17:34:01] El contenedor mi_nginx NO está en ejecución. Intentando reiniciarlo...
[2025-04-07_17:34:01] El contenedor mi_nginx ha sido reiniciado exitosamente.
[2025-04-07_17:36:02] El contenedor mi_nginx está en ejecución.
[2025-04-07_17:38:01] El contenedor mi_nginx está en ejecución.
salva@salva-ubuntu:~/Documentos/Onboarding-devops-/CRON$
```

Vemos en la salida de `docker ps -a` que después de detener el contenedor `mi_nginx`, volvió a estar en estado `Up` en aproximadamente un minuto.

Y el contenido del archivo de log **/var/log/monitoreo\_nginx.log** confirma el comportamiento esperado:

- El script se ejecutó a las 17:30:01 y detectó que el contenedor estaba en ejecución.
- A las 17:32:01 también estaba en ejecución.
- A las 17:34:01, después de que lo detuve manualmente, el script detectó que no estaba en ejecución, y lo reinició.
- Luego, en las ejecuciones posteriores (17:36:02 y 17:38:01), el script verificó y encontró que el contenedor ya estaba en ejecución.