

# ADSO Training 6

Temporització de tasques

# Índex

1. Objectiu	2
2. Abans de començar	2
3. Tasques puntuals	2
4. Tasques periòdiques	3

## 1. Objectiu

Ser capaç de programar tasques de manera puntual i periòdica.

## 2. Abans de començar

Hauríeu de tenir fets els següents scripts de les pràctiques anteriors:

- ocupacio.sh
- badusers.py

## 3. Tasques puntuals

Amb quina comanda es programen habitualment les tasques puntuals?

Per a tasques puntuals, tasques que s'han de realitzar una sola vegada en un moment específic en el futur, normalment s'utilitza la comanda `'at'`. Amb `'at'`, es pot programar una tasca perquè s'executi en un moment específic, i la tasca s'executarà una sola vegada.

Per instal·lar at usar aquestes 2 comandes:

- [`sudo apt-get update`](#)
- [`sudo apt-get install at`](#)

Exemple d'execució de la comanda `at`:

```
root@francesco0 (Mon Dec 11):<~># echo "hello" >> example.txt | at now
warning: commands will be executed using /bin/sh
job 11 at Mon Dec 11 16:00:00 2023
root@francesco0 (Mon Dec 11):<~># cat example.txt
hello
root@francesco0 (Mon Dec 11):<~># █
```

Si no deixa utilitzar el `at` executar la següent comanda:

```
root@francesco0 (Mon Dec 11):<~># sudo systemctl enable --now atd
Synchronizing state of atd.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable atd
```

Fent servir aquesta comanda programeu les següents tasques:

Un esborrat dels fitxers del directori `/tmp` a les 2 de la nit d'avui.

Escrivint la següent comanda al terminal ja quedaria programat l'esborrat de fitxers:  
`echo "find /tmp -type f -delete" | at 2:00`

```

root@March (Thu Nov 23):~# echo "find /tmp -type f -delete" | at 2:00
warning: commands will be executed using /bin/sh
job 1 at Fri Nov 24 02:00:00 2023

```

Podem veure el llistat de les tasques programades amb la comanda "atq", i eliminar una tasca en concret amb la comanda "atrm", seguit del número de la tasca que volem eliminar.

Comanda atq per veure les tasques:

```

root@March (Thu Nov 23):~# atq
1      Fri Nov 24 02:00:00 2023 a root

```

Comanda atrm + comanda atq per veure que la tasca s'ha eliminat correctament:

```

root@March (Thu Nov 23):~# atrm 1
root@March (Thu Nov 23):~# atq
root@March (Thu Nov 23):~#

```

Fer un llistat dels usuaris connectats al servidor dintre de 10 minuts. Aquest llistat s'ha de deixar al home del root amb el nom usuaris-data.txt , on data serà la data del moment en que s'ha fet el llistat.

Escrivint la següent comanda al terminal s'ens creara el llistat d'usuaris :

echo "who > /root/usuaris-\$(date +%Y%m%d%H%M%S).txt" | at now + 10 minutes

```

root@marionaF (Tue Dec 12):<~# echo "who > /root/usuaris-$(date +%Y%m%d%H%M%S).txt" | at now + 10 minutes
warning: commands will be executed using /bin/sh
job 13 at Tue Dec 12 17:21:00 2023

```

Al cap de 10 minuts a la carpeta home del root podem veure el fitxer creat amb la data establerta: any 2023 més 12 del dia 12 a la hora 17 minut 21 i segon 00

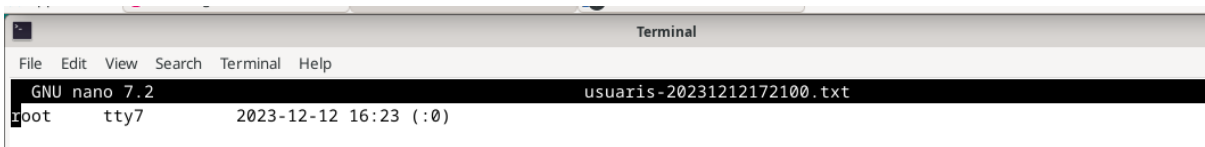
```

root@marionaF (Tue Dec 12):<~# ls
-hola.txt  Downloads  Public      ascii2binary_install  private.key  usuaris-20231212171129.txt
Desktop   Music      Templates  autoInstall.sh        sudoers      usuaris-20231212172100.txt
Documents Pictures  Videos    backups                sudousers

```

(el fitxer seleccionat)

Podem entrar al fitxer i veure el llistat d'usuaris connectats al servidor:



```

GNU nano 7.2 usuaris-20231212172100.txt
root      tty7      2023-12-12 16:23 (:0)

```

Actualment només hi ha el usuari root

Un shutdown del vostre servidor al final de la classe. Volem que avisi als usuaris del sistema 5 minuts abans.

La instrucció per un shutdown directe es:

```
echo "shutdown -h now" | at now
```

Per posar el avis als usuaris creem un missatge per tots i seguit de la instrucció shutdown amb un temporitzador de 5 minuts:

```
echo "echo 'Avis: El sistema s'apagarà en 5 minuts.'; shutdown -h now" | at now + 5 minutes
```

```
root@marionaF (Tue Dec 12):<~># echo "echo 'Avis: El sistema sapagarà en 5 minuts.'; shutdown -h now" | at now + 5 minutes
warning: commands will be executed using /bin/sh
job 32 at Tue Dec 12 18:21:00 2023
root@marionaF (Tue Dec 12):<~>#
```

A partir de l'execució de la instrucció, es farà un shutdown de l'usuari després de 5 minuts.

Volem restringir aquest servei a l'usuari root.

Quin fitxer hem de crear? Amb quin contingut?

Per restringir l'ús de la comanda at només a l'usuari root, s'han de gestionar els fitxers at.allow i at.deny, que controlen quins usuaris poden utilitzar at.

Si el fitxer at.allow existeix, només els usuaris llistats en aquest fitxer poden utilitzar at, i el fitxer at.deny és ignorat. Si at.allow no existeix, at llavors consulta at.deny per determinar quins usuaris estan exclosos d'utilitzar at. Si cap dels dos fitxers existeix, només l'usuari root podrà utilitzar at.

Passos per restringir at només a root:

- Crear o editar el fitxer at.deny: aquest fitxer conté la llista d'usuaris que no poden utilitzar at. Si no volem permetre a cap usuari (excepte root) utilitzar at, aquest fitxer hauria de contenir els noms de tots els usuaris excepte root. No obstant això, una manera més senzilla és simplement no utilitzar at.deny i en lloc d'això utilitzar at.allow.
- Crear el fitxer at.allow: aquest fitxer hauria de contenir els noms d'usuari dels que poden utilitzar at. Si només volem que root tingui accés, simplement crea aquest fitxer i afegeix només el nom d'usuari root.

Per crear el fitxer at.allow amb només root com a usuari permès:

- echo "root" | sudo tee /etc/at.allow

Això escriurà la paraula "root" en el fitxer /etc/at.allow.

```

root@francesco0 (Mon Dec 11):<~># cd /etc/
root@francesco0 (Tue Dec 12):</etc># ls
ImageMagick-6          console
PackageKit             console-setup
UPower                 cron.d
X11                    cron.daily
adduser.conf           cron.hourly
adduser.conf.update-old cron.monthly
aliases                cron.weekly
alsa                   cron.yearly
alternatives           crontab
apache2                dbus-1
apparmor               dconf
apparmor.d             debconf.conf
appstream.conf         debian_version
apt                    default
at.deny                deluser.conf

root@francesco0 (Tue Dec 12):</etc># echo "root" | sudo tee /etc/at.allow
root
root@francesco0 (Tue Dec 12):</etc># ls
ImageMagick-6          calendar          ethertypes       init.d
PackageKit            console          exim4            initramfs
UPower                console-setup   firefox-esr      inputrc
X11                   cron.d          fonts            ipp-usb
adduser.conf          cron.daily      fstab            iproute2
adduser.conf.update-old cron.hourly      fuse.conf        issue
aliases               cron.monthly    gai.conf         issue.net
alsa                  cron.weekly     ghostscript      kernel
alternatives          cron.yearly     glvnd            ld.so.cache
apache2               crontab         gprofng.rc       ld.so.conf
apparmor              dbus-1          groff            ld.so.conf
apparmor.d            dconf           group            ldap
appstream.conf        debconf.conf    group-           libaudit
apt                   debian_version  group.org        libblock
at.allow              default         grub.d           libpaper
at.deny               deluser.conf    gshadow          lightdm

```

A la imatge de l'esquerra podem veure com només m'apareix el at.deny, de manera que haurem de crear el fitxer at.allow i afegir l'usuari root (que és la imatge de la dreta).

Proveu de fer servir la comanda at amb un altre usuari i comproveu que efectivament no us deixa.

A la següent imatge podem observar com quan utilitzem la mateixa comanda que hem utilitzat al principi amb root ens diu que no tenim permís per utilitzar-la:

```

aso@francesco0 (Tue Dec 12):</>$ echo "hello" >> example_aso.txt | at now
bash: example_aso.txt: Permission denied
You do not have permission to use at.
aso@francesco0 (Tue Dec 12):</>$

```

## 4. Tasques periòdiques

Mireu el manual de les comandes cron i crontab i contesteu a les següents preguntes:

Com podem veure quin és el contingut del cron d'un usuari?

Per veure el contingut, hem d'utilitzar `'crontab -l'`. Aquesta mostra les tasques

programades actuals per a l'usuari que executa la comanda. Com a administrador es pot veure el crontab d'un altre usuari:

'crontab -u nom\_usuari -l'.

El crontab es veu d'aquesta manera:

```
aso@francesco0 (Tue Dec 12):</>$ su root
Password:
root@francesco0 (Tue Dec 12):</># crontab -u aso -l
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
*/5 * * * * date '+%F %T' > /tmp/hora_actual.txt
root@francesco0 (Tue Dec 12):</># █
```

Com podem afegir-hi noves tasques periòdiques?

Per afegir noves tasques periòdiques al crontab d'un usuari, ho podem fer amb la comanda 'crontab -e'. Aquesta comanda obre l'editor de text on es poden afegir noves línies per a cada tasca que es vulguin programar. Exemple d'una línia de crontab: 'minut hora dia mes mes dia setmana comanda a executar'.

```
root@francesco0 (Tue Dec 12):</etc># crontab -e
no crontab for root - 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

Choose 1-3 [1]: 1
No modification made
root@francesco0 (Tue Dec 12):</etc>#
```

Quan obrim el fitxer veiem:

```
# 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
```

Accedim amb l'usuari aso i fem que cada 5 minuts imprimeixi en un fitxer la data actual:

'\*/5 \* \* \* \* date '+\%F \%T' > /tmp/hora\_actual.txt'

```
root@francesco0 (Wed Dec 13):</tmp># ls
hora_actual.txt
systemd-private-5ba7635fc3734d20a9ed0da23069cdf5-colord.service-mcuWdC
systemd-private-5ba7635fc3734d20a9ed0da23069cdf5-systemd-logind.service-84GBXI
systemd-private-5ba7635fc3734d20a9ed0da23069cdf5-systemd-timesyncd.service-hwWxLi
systemd-private-5ba7635fc3734d20a9ed0da23069cdf5-upower.service-DwJ5sR
root@francesco0 (Wed Dec 13):</tmp># cat hora_actual.txt
2023-12-13 10:45:01
root@francesco0 (Wed Dec 13):</tmp>#
```

Com podem limitar qui té accés al servei de cron?

Per limitar l'accés, farem el mateix que amb la comanda 'at', utilitzarem els fitxers cron.allow i cron.deny per limitar els que poden utilitzar-la.

```
root@francesco0 (Wed Dec 13):</etc># echo "root" | sudo tee /etc/cron.allow
root
root@francesco0 (Wed Dec 13):</etc># ls
ImageMagick-6      calendar          environment.d     hosts.der
PackageKit         console          ethertypes       init.d
UPower            console-setup    exim4            initramfs
X11               cron.allow       firefox-esr      inputrc

aso@francesco0 (Wed Dec 13):</root>$ crontab -e
You (aso) are not allowed to use this program (crontab)
See crontab(1) for more information
aso@francesco0 (Wed Dec 13):</root>$
```

Un problema a l'hora de fer servir cron és assegurar-nos que hem especificat correctament la freqüència sense haver d'esperar i veure que quan arriba el moment no s'executa (en particular quan falta molt de temps).

Que podríem fer per comprovar en el mateix moment que els nostres fitxers de crontab són correctes?

Quins efectes col·laterals tindria això?

Estratègies per assegurar que les entrades de crontab són correctes (juntament amb els efectes col·laterals):

### 1. Provar amb una programació a curt termini

Per comprovar si una entrada de crontab està ben configurada podem programar una tasca perquè s'executi d'aquí a pocs minuts. Per exemple, si són les 12:00, programar la tasca perquè s'executi a les 12:05. Així podrem veure ràpidament si s'executa correctament.

Efectes col·laterals: això pot provocar l'execució de tasques no desitjades, especialment si provem scripts o comandes que afecten el sistema o les dades.

### 2. Validar la sintaxi amb eines en línia

Hi ha eines en línia que permeten validar la sintaxi de les entrades de crontab, com els generadors de sintaxi cron o els verificadors cron. Aquests serveis permeten introduir l'expressió cron i informen si hi ha errors en la sintaxi.

Efectes col·laterals: aquest mètode només comprova la sintaxi, no l'execució real ni els permisos necessaris per executar la tasca.

### 3. Escriptura de logs

Modificant els scripts o comandes perquè escriguin en un fitxer de log quan s'executen, ens permetrà comprovar si la comanda s'ha executat i en quin moment.

Efectes col·laterals: requereix modificar els scripts o les comandes per incloure l'escriptura de logs, la qual cosa pot ser laboriosa.

### 4. Utilitzar echo per a proves



En lloc d'executar la comanda real, es pots substituir temporalment per un echo que imprimeixi un missatge. És útil per veure si cron està executant les tasques en els temps esperats.

Efectes col·laterals: no estarem executant la tasca real durant aquestes proves, així que no obtindrem una validació completa.

## 5. Revisió manual

Revisar manualment les entrades de crontab per assegurar que la sintaxi és correcta i que els temps estan ben especificats. També es pot demanar a algú altre que revisi les entrades.

Efectes col·laterals: requereix un bon coneixement de la sintaxi cron, i per tant, pot ser susceptible a errors humans.

Volem controlar l'ocupació d'una sèrie d'usuaris (fent servir l'script ocupacio.sh que vareu fer) que estaran especificats al fitxer /etc/ocupacio.users. El control es farà tots els diumenges.

Quina entrada afegireu al crontab? De quin usuari?

```
aso@AdrianGasco-client: ~
File Edit View Search Terminal Help
GNU nano 7.2 /tmp/crontab.yEd62H/crontab *
# 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 0 * * 0 /etc/ocupacio.users/ocupacio.sh
```

Obrim el crontab amb l'usuari root, que té permisos per executar scrpts.

Afegim la següent linia:

0 0 \* \* 0 /etc/ocupacio.users/ocupacio.sh

```
crontab: installing new crontab
root@AdrianG (Tue Dec 12):/home/homeB/aso#
```

Comprovem que s'hagi instalat correctament.

També volem controlar els usuaris innecessaris del sistema fent servir l'script badusers.py. Aquest control es realitzarà el primer dia de cada mes.

Quina entrada afegireu al crontab? De quin usuari?

```
# m h dom mon dow command
0 0 * * 0 /etc/ocupacio.users/ocupacio.sh
0 0 1 * * /usr/local/bin/badusers.sh
```

Afegim la linia següent al crontab del root:

```
0 0 1 * * /usr/local/bin/badusers.sh
```

La direcció del script pot variar.

Que hem de fer perquè el llistat d'usuaris que genera l'script ens arribi per correu?

```
.
# m h dom mon dow  command
) 0 * * 0 /etc/ocupacio.users/ocupacio.sh
) 0 1 * * /usr/local/bin/badusers.sh | mail -s "Llistat usuaris" correu@exemple.com
```

Modifiquem la línia anterior per a que l'informació de l'script l'envii per correu.

```
0 0 1 * * /usr/local/bin/badusers.sh | mail -s "Llistat usuaris"
correu@exemple.com
```

D'aquesta manera tenim un correu amb l'assumpte Llistat usuaris.

L'usuari sergiS vol fer backups del seu home. Feu servir cron perquè es compleixin les següents condicions:

- És farà un backup total cada primer dilluns de mes
- És farà un backup incremental tots els dimecres i dissabtes respecte al backup total.
- Els backups es guardaran al mateix home de l'usuari sergiS a un directori backups. Evidentment no s'ha de fer backup dels backups.
- El nom dels backups ha d'incloure la data en la que es varen fer i el tipus de backup.
- Per reduir l'espai que ocupen el primer dia de cada mes es comprovarà si la mida total dels backups és superior a 100 Mb. Si és així s'esborraran tants backups com calgui fins estar per sota del límit en ordre cronològic ascendent (primer els més vells).
- L'usuari sergiS només vol rebre missatges de cron si hi ha errors.

Primer el que hem de fer es crear un arxiu .sh que serà l'encarregat de realitzar els backups de cada primer dilluns de mes. En el codi també incloem el control de mida.

### **backup\_total.sh**

```
#!/bin/bash

# Definir las variables
BACKUP_DIR="/home/SergiS/SergiS_total_backups/"
INC_DIR="/home/SergiS/SergiS_incremental_backups"
SOURCE_DIR="/home/SergiS"
DATE=$(date +%Y%m%d%H%M)
BACKUP_FILE="$BACKUP_DIR/total_backup_$(date +%Y%m%d%H%M).tar.gz"

tar cvf $BACKUP_FILE --exclude="$BACKUP_DIR" --exclude="$INC_DIR" $SOURCE_DIR

# Verificar el tamaño total de los backups
TOTAL_SIZE=$(du -s $BACKUP_DIR | cut -f1)

# Si el tamaño es mayor a 100 MB, eliminar los archivos más antiguos
while [ $TOTAL_SIZE -gt 102400 ]; do
    OLDEST_FILE=$(ls -t $BACKUP_DIR | tail -1)
    rm -f $BACKUP_DIR/$OLDEST_FILE
    TOTAL_SIZE=$(du -s $BACKUP_DIR | cut -f1)
done
```

**TOTAL\_SIZE=\$(du -s \$BACKUP\_DIR | cut -f1)**: Calcula el tamaño total del directorio de respaldo (**\$BACKUP\_DIR**) utilizando **du -s** (que devuelve el tamaño sumario en kilobytes) y extrae solo la cifra del tamaño usando **cut -f1**.

El bucle **while** se ejecuta mientras el tamaño total del directorio de respaldo sea mayor a **100 MB** (102400 KB).

**OLDEST\_FILE=\$(ls -t \$BACKUP\_DIR | tail -1)** Dentro del bucle, identifica el archivo más antiguo en el directorio de respaldo.

**rm -f \$BACKUP\_DIR/\$OLDEST\_FILE**: Elimina el archivo más antiguo.

El tamaño total se recalcula después de cada eliminación, y el bucle continúa hasta que el tamaño total sea menor o igual a 100 MB.

```

root@MarcR (Wed Dec 13):</media/usb># mkdir /home/SergiS/Downloads
root@MarcR (Wed Dec 13):</media/usb># mkdir /home/SergiS/Fotos
root@MarcR (Wed Dec 13):</media/usb># mkdir /home/SergiS/CosasTurbias
root@MarcR (Wed Dec 13):</media/usb># ./backup_total.sh
tar: Removing leading `/' from member names
/home/SergiS/
/home/SergiS/Fotos/
/home/SergiS/CosasTurbias/
/home/SergiS/Downloads/
root@MarcR (Wed Dec 13):</media/usb>#

```

Una vegada tenim aquest arxiu fem el que s'encarregarà de fer les còpies incrementals. **backup\_incremental.sh**

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

```
# Definir las variables
```

```
BACKUP_DIR="/home/SergiS/SergiS_incremental_backups/"
```

```
TOT_DIR="/home/SergiS/SergiS_total_backups"
```

```
SOURCE_DIR="/home/SergiS"
```

```
DATE=$(date +%Y%m%d)
```

```
BACKUP_FILE="$BACKUP_DIR/incremental_backup_$DATE.tar.gz"
```

```
SNAPSHOT_FILE="$BACKUP_DIR/snapshot.file"
```

```
# Crear el archivo de copia de seguridad incremental
```

```
tar -cvf $BACKUP_FILE --exclude="$BACKUP_DIR" --exclude="$TOT_DIR"
```

```
--listed-incremental=$SNAPSHOT_FILE $SOURCE_DIR
```

```

root@MarcR (Wed Dec 13):</media/usb># mkdir /home/SergiS/FotosBaño
root@MarcR (Wed Dec 13):</media/usb># mkdir /home/SergiS/CuentasCocaina
root@MarcR (Wed Dec 13):</media/usb># mkdir /home/SergiS/PonerCeroATodos
root@MarcR (Wed Dec 13):</media/usb># ./backup_incremental.sh
tar: Removing leading `/' from member names
root@MarcR (Wed Dec 13):</media/usb># cd /home/SergiS/SergiS_incremental_backups/
root@MarcR (Wed Dec 13):</home/SergiS/SergiS_incremental_backups># ls
incremental_backup_20231213194635.tar.gz snapshot.file
root@MarcR (Wed Dec 13):</home/SergiS/SergiS_incremental_backups># tar -zxvf incrementa
l_backup_20231213194635.tar.gz
home/SergiS/
home/SergiS/CosasTurbias/
home/SergiS/CuentasCocaina/
home/SergiS/Downloads/
home/SergiS/Fotos/
home/SergiS/FotosBaño/
home/SergiS/PonerCeroATodos/
root@MarcR (Wed Dec 13):</home/SergiS/SergiS_incremental_backups># cd home/SergiS/
root@MarcR (Wed Dec 13):</home/SergiS/SergiS_incremental_backups/home/SergiS># ls
CosasTurbias CuentasCocaina Downloads Fotos FotosBaño PonerCeroATodos
root@MarcR (Wed Dec 13):</home/SergiS/SergiS_incremental_backups/home/SergiS># █

```

Creem dos arxius per a provar:

```
root@MarcR (Wed Dec 13):</media/usb># nano /home/SergiS/CosasTurbias/equisde.txt
root@MarcR (Wed Dec 13):</media/usb># nano /home/SergiS/CuentasCocaina/cuentas.txt
```

Descomprimir el tar per a veure que hi ha els arxius:

```
root@MarcR (Wed Dec 13):</home/SergiS/SergiS_incremental_backups># tar -zxvf incrementa
l_backup_20231213200147.tar.gz
home/SergiS/
home/SergiS/CosasTurbias/
home/SergiS/CuentasCocaina/
home/SergiS/Downloads/
home/SergiS/Fotos/
home/SergiS/FotosBaño/
home/SergiS/PonerCeroATodos/
home/SergiS/CosasTurbias/equisde.txt
home/SergiS/CuentasCocaina/cuentas.txt
root@MarcR (Wed Dec 13):</home/SergiS/SergiS_incremental_backups># cd home/
root@MarcR (Wed Dec 13):</home/SergiS/SergiS_incremental_backups/home># ls
SergiS
root@MarcR (Wed Dec 13):</home/SergiS/SergiS_incremental_backups/home># cd SergiS/
root@MarcR (Wed Dec 13):</home/SergiS/SergiS_incremental_backups/home/SergiS># ls
CosasTurbias CuentasCocaina Downloads Fotos FotosBaño PonerCeroATodos
root@MarcR (Wed Dec 13):</home/SergiS/SergiS_incremental_backups/home/SergiS># cd Cosas
Turbias/
root@MarcR (Wed Dec 13):</home/SergiS/SergiS_incremental_backups/home/SergiS/CosasTurbi
as># ls
equisde.txt
```

Creem un altre arxiu:

```
root@MarcR (Wed Dec 13):</media/usb># nano /home/SergiS/FotosBaño/pervertido.txt
```

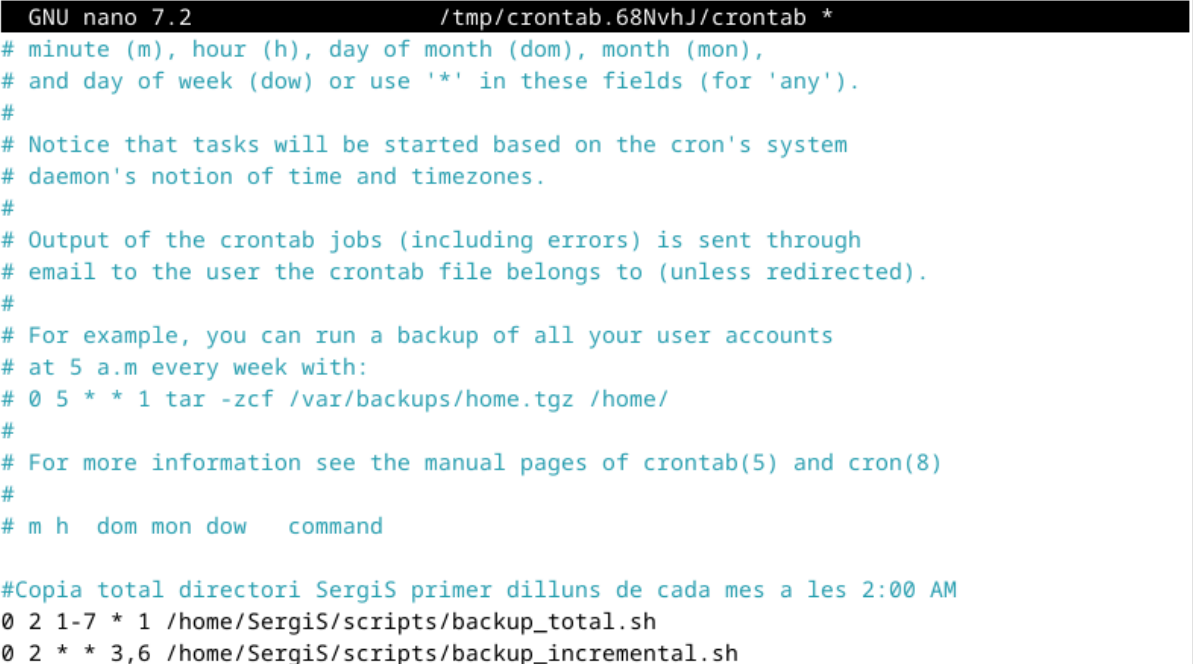
Descomprimim el nou tar per a veure que els dos arxius anteriors no estan, y només està el nou

```
root@MarcR (Wed Dec 13):</home/SergiS/SergiS_incremental_backups># tar -zxvf incrementa
l_backup_20231213200357.tar.gz
home/SergiS/
home/SergiS/CosasTurbias/
home/SergiS/CuentasCocaina/
home/SergiS/Downloads/
home/SergiS/Fotos/
home/SergiS/FotosBaño/
home/SergiS/PonerCeroATodos/
home/SergiS/FotosBaño/pervertido.txt
root@MarcR (Wed Dec 13):</home/SergiS/SergiS_incremental_backups># cd home/SergiS/C
CosasTurbias/ CuentasCocaina/
root@MarcR (Wed Dec 13):</home/SergiS/SergiS_incremental_backups># cd home/SergiS/Cosas
Turbias/
root@MarcR (Wed Dec 13):</home/SergiS/SergiS_incremental_backups/home/SergiS/CosasTurbi
as># ls
root@MarcR (Wed Dec 13):</home/SergiS/SergiS_incremental_backups/home/SergiS/CosasTurbi
as>#
```

Ara utilitzarem la comanda **crontab -e** i afegirem les següents instruccions

```
# Copia de seguridad total el primer lunes de cada mes a las 2:00 AM
0 2 1-7 * 1 /path/to/backup_total.sh
```

```
# Copia de seguridad incremental los miércoles y sábados a las 2:00 AM
0 2 * * 3,6 /path/to/backup_incremental.sh
```



```
GNU nano 7.2 /tmp/crontab.68NvhJ/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

#Copia total directori SergiS primer dilluns de cada mes a les 2:00 AM
0 2 1-7 * 1 /home/SergiS/scripts/backup_total.sh
0 2 * * 3,6 /home/SergiS/scripts/backup_incremental.sh
```

Posteriorment sense tancar el crontab afegirem també les següents instruccions per a configurar les notificacions del cron.

```
0 2 1-7 * * [ "$(date '+%u')" -eq 1 ] && /path/to/backup_total.sh > /dev/null
0 2 * * 3,6 /path/to/backup_incremental.sh > /dev/null
0 2 * * 3,6 /path/to/backup_incremental.sh > backup_output.txt
```

**0 2 1-7 \* \*** Programa el cron job para las 2:00 AM en días del 1 al 7 de cada mes.

**[ "\$(date '+%u')" -eq 1 ]** Comprueba si el día actual es lunes (%u devuelve el día de la semana como un número, donde 1 es lunes).

**&& /path/to/backup\_total.sh** Ejecuta el script si la condición anterior es verdadera.

**> /dev/null** Redirige la salida a /dev/null, descartando cualquier salida del script.

```
GNU nano 7.2 /tmp/crontab.68NvhJ/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

#Copia total directori SergiS primer dilluns de cada mes a les 2:00 AM
0 2 1-7 * 1 /home/SergiS/scripts/backup_total.sh > /dev/null
0 2 * * 3,6 /home/SergiS/scripts/backup_incremental.sh > /dev/null
```