

ADSO Training 5

Manteniment del sistema de
fitxers

Índex

1. Introducció	3
1.1. Objectiu	4
2. Abans de començar	4
3. Partició per emmagatzemar les còpies de seguretat.	4
Quina modificació és necessària fer perquè aquesta nova partició es munti automàticament durant el boot amb mode de només lectura?	4
3. Realització de còpies amb TAR	5
3.1. Realització de còpies completes	5
3.2. Realització de còpies incrementals	6
3.3. Restauració d'una còpia de seguretat	11
3.4. Restauració d'un fragment	13
4. Realització de còpies usant RSYNC	14
4.1. Realització de backups a través d'una xarxa	14
\$ echo "nou arxiu" > /root/arxiu_nou.txt	15
4.2. Realització de còpies incrementals inverses	18
4.3. Realització de còpies incrementals inverses tipus snapshot	20
Repàs d'enllaços durs	20
4.3.1 Backups tipus "snapshot" amb rsync i cp -al	21
4.4. Script per fer backups tipus snapshot	22
5. Referències	23
6. Apèndix. Codi de l'script per còpies tipus snapshot	23

1. Introducció

Una de les tasques més importants de l'administrador de sistemes és la realització de còpies de seguretat que permeten restaurar el sistema complet en una quantitat acceptable de temps quan es produeix una fallada del sistema amb pèrdua de dades. Aquestes pèrdues poden ser degudes a múltiples factors com poden ser fallades de hardware, de software, accions humanes (accidentals o premeditades) o desastres naturals .

Abans de començar a realitzar còpies de seguretat l'administrador del sistema ha de decidir una política tenint en compte aspectes com:

Seleccionar el tipus correcte de medi físic per fer les còpies de seguretat tenint en compte la grandària, el cost, la velocitat, la disponibilitat, l'usabilitat i la confiabilitat.

Decidir quins fitxers necessiten una còpia de seguretat i on són aquest fitxers. Són més importants el fitxers de configuració del sistema i els fitxers dels usuaris (normalment ubicats a /etc i /home respectivament) que el fitxers temporals o els binaris del sistema (/tmp i /bin)

Decidir la freqüència i el tipus de planificació de les còpies de seguretat. Això depèn de la variabilitat de les dades. Una base de dades pot necessitar múltiples còpies de seguretat diàries, mentre que un servidor web pot requerir només una còpia diària i altres sistemes de fitxers poden requerir només una còpia setmanal.

Analitzar altres aspectes com: on s'han d'emmagatzemar les còpies, per quan temps s'han de mantenir i amb quina rapidesa es necessita poder recuperar cada tipus de fitxer.

Utilitzant tota la informació anterior es pot decidir finalment una estratègia de còpies de seguretat. Això inclou decidir la freqüència de les còpies i el tipus. Una estratègia comú es fer còpies completes i còpies incrementals. D'aquesta manera es pot disminuir el temps i la grandària de les transferències de dades de les còpies però també s'incrementa la complexitat de la restauració de les dades.

Una estratègia típica consisteix en realitzar còpies completes (també conegudes com de nivell 0) setmanalment i còpies incrementals (conegudes com de nivell 1 o més gran) diàriament. Si el grau de variabilitat dels fitxers és molt gran es pot modificar l'anterior model setmanal per un model mensual on cada més es realitza una còpia de nivell 0, cada setmana una còpia de nivell 1 (incremental setmanal sobre el nivell 0) i cada dia es fa una còpia de nivell 2 (incremental diari sobre el nivell 1).

Per últim s'han de decidir les eines més adequades per implementar l'estratègia de còpies de seguretat que s'ha dissenyat. Com a primer pas en aquest objectiu usarem el mateix disc dur com medi físic per fer les còpies de seguretat tot i que no és el més convenient habitualment a causa del alt risc de que una pèrdua de dades del disc afecti també a les còpies de seguretat. Per fer les còpies utilitzarem dos tipus diferents d'aplicacions de còpia de seguretat: **tar**, que realitza les còpies a través del sistema de fitxers i **rsync** que permet sincronitzar discs amb moltes opcions de configuració i per tant permet implementar diferents estratègies de còpies de seguretat.

1.1. Objectiu

Aprendre a dissenyar i implementar sistemes de còpies de seguretat tot utilitzant eines bàsiques de UNIX.

2. Abans de començar

contesteu les següents preguntes abans de començar:

1. Com es pot empaquetar i desempaquetar un(s) fitxer(s) utilitzant la comanda tar?

Per empaquetar hem d'utilitzar:

```
tar -cvf nom_arxiu.tar fitxer1 fitxer2 ..
```

Aquí, -c indica crear un arxiu, -v és per veure el procés (verbose), i -f especifica el nom de l'arxiu resultant. Pots afegir múltiples fitxers i directoris separant-los amb espais.

Per desempaquetar hem d'utilitzar:

```
tar -xvf nom_arxiu.tar
```

On -x indica extreure el contingut de l'arxiu.

2. Què és un enllaç dur (hard link)? I quina diferència hi ha entre fer una còpia (cp file_a file_b) i fer un hard link (ln file_a file_b)?

Un enllaç dur és una referència directa al contingut físic d'un fitxer al disc. Quan crees un enllaç dur a un fitxer, estàs creant essencialment una altra entrada al mateix bloc de dades al disc. Això significa que el fitxer original i l'enllaç dur tenen el mateix identificador de node d'índex (inode) i qualsevol canvi en un es reflectirà en l'altre.

La diferència clau entre fer una còpia (cp file_a file_b) i crear un enllaç dur (ln file_a file_b) rau en com es tracta el fitxer en el sistema de fitxers. Amb cp, es crea una nova

instància del fitxer, que posseeix un inode únic; això significa que els fitxers resultants són independents, i els canvis en un no afecten l'altre. En contrast, quan es crea un enllaç dur, no es genera una còpia física del fitxer, sinó una nova entrada al sistema de fitxers que apunta al mateix inode que l'original. Això fa que el fitxer original i l'enllaç dur siguin intercanviables, compartint els mateixos continguts, de manera que qualsevol canvi en un es reflecteix automàticament en l'altre.

3. Partició per emmagatzemar les còpies de seguretat.

Creeu una nova partició a l'espai lliure que teniu al disc i doneu-li format extended3. Munteu aquesta partició sobre el directori /backup de forma que tan sols root tingui permisos d'accés. La resta d'usuaris no han de tenir ni tan sols permís de lectura al directori ja que el contingut de les còpies de seguretat podria ser informació confidencial.

Quines comandes heu utilitzat per crear la partició, donar format al sistema de fitxers, muntar la partició i canviar els permisos del directori backup?

Primer de tot entrem al fdisk. Per fer-ho utilitzem "fdisk /dev/sda".

```
root@AdrianG (Sun Dec 17):/home/homeB/aso# fdisk /dev/sda

Welcome to fdisk (util-linux 2.38.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

This disk is currently in use - repartitioning is probably a bad idea.
It's recommended to umount all file systems, and swapoff all swap
partitions on this disk.

Command (m for help): p
```

Creem una nova partició utilitzant "n". Li assignem el valor de partició primària, i li donem un espai de 100 MB. Si després volem ampliar l'espai ja ho farem. Finalment guardem els canvis amb "w".

```

Command (m for help): n
Partition type
  p   primary (2 primary, 1 extended, 1 free)
  l   logical (numbered from 5)
Select (default p): p

Selected partition 4
First sector (54528000-67108863, default 54528000):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (54528000-67108863, default 67108863): +100M

Created a new partition 4 of type 'Linux' and of size 100 MiB.

Command (m for help): w
The partition table has been altered.
Syncing disks.

root@AdrianG (Sun Dec 17):/home/homeB/aso# █

```

Donem el sistema de fitxers ext3 amb la següent comanda.

`mkfs -t ext3 /dev/sda4`

```

root@AdrianG (Sun Dec 17):/home/homeB/aso# mkfs -t ext3 /dev/sda4
mke2fs 1.47.0 (5-Feb-2023)
Creating filesystem with 102400 1k blocks and 25584 inodes
Filesystem UUID: cb8335e5-aa7c-44b5-a4d1-f6aea10e49da
Superblock backups stored on blocks:
    8193, 24577, 40961, 57345, 73729

Allocating group tables: done
Writing inode tables: done
Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done

root@AdrianG (Sun Dec 17):/home/homeB/aso# █

```

Ara hem de crear la carpeta del /backup amb mkdir. Una vegada creada utilitzem les següents comandes per donar-li només permisos al root per poder entrar a la carpeta.

`chown root:root /backup`

`chmod 700 /backup`

```

root@AdrianG (Sun Dec 17):/home/homeB/aso# mkdir /backup
root@AdrianG (Sun Dec 17):/home/homeB/aso# chown root:root /backup
root@AdrianG (Sun Dec 17):/home/homeB/aso# chmod 700 /backup
root@AdrianG (Sun Dec 17):/home/homeB/aso# █

```

Montem el directori al sda4, i el remontem per a que sigui read-write amb les següents comandes.

```
mount /dev/sda4 /backup
```

```
mount -o remount,rw /dev/sda4 /backup
```

```
root@AdrianG (Sun Dec 17):/home/homeB/aso# mount /dev/sda4 /backup/
root@AdrianG (Sun Dec 17):/home/homeB/aso# mount -o remount,rw /dev/sda4 /backup/
root@AdrianG (Sun Dec 17):/home/homeB/aso#
```

Per afegir més proteccions a aquest directori es pot muntar en mode de escriptura només quan s'escriguin els backups i la resta del temps muntar-lo en mode de només lectura. Normalment, s'hauria de desmuntar i tornar a muntar canviant les opcions per defecte. Però és possible canviar les opcions d'una partició sense desmuntar-la si feu servir l'opció **remount**.

Muntar només per lectura:

```
$ mount -o remount,ro /dev/usb4 /backup
```

Muntar per lectura i escriptura:

```
$ mount -o remount,rw /dev/usb4 /backup
```

Quina modificació és necessària fer perquè aquesta nova partició es munti automàticament durant el boot amb mode de només lectura?

Obrim el fstab fent "nano /etc/fstab", i afegim la següent línia.

```
/dev/sda4 /backup ext3 ro 0 2
```



```
aso@AdrianGasco-client: ~
File Edit View Search Terminal Help
GNU nano 7.2 /etc/fstab *
#<dispositiu> <directori> <tipus> <parametres> <dump> <pass>
/dev/sda1 / ext4 defaults 0 1
UUID="2df3bccf-15df-4da5-8923-fba57fc1585c" none swap defaults 0 0
/dev/sda5 /usr/local ext4 defaults 0 2
/dev/sda6 /home/homeB ext4 defaults 0 2
/dev/sdb1 /home/homeA ext4 defaults 0 2
/dev/sda4 /backup ext3 ro 0 2
```

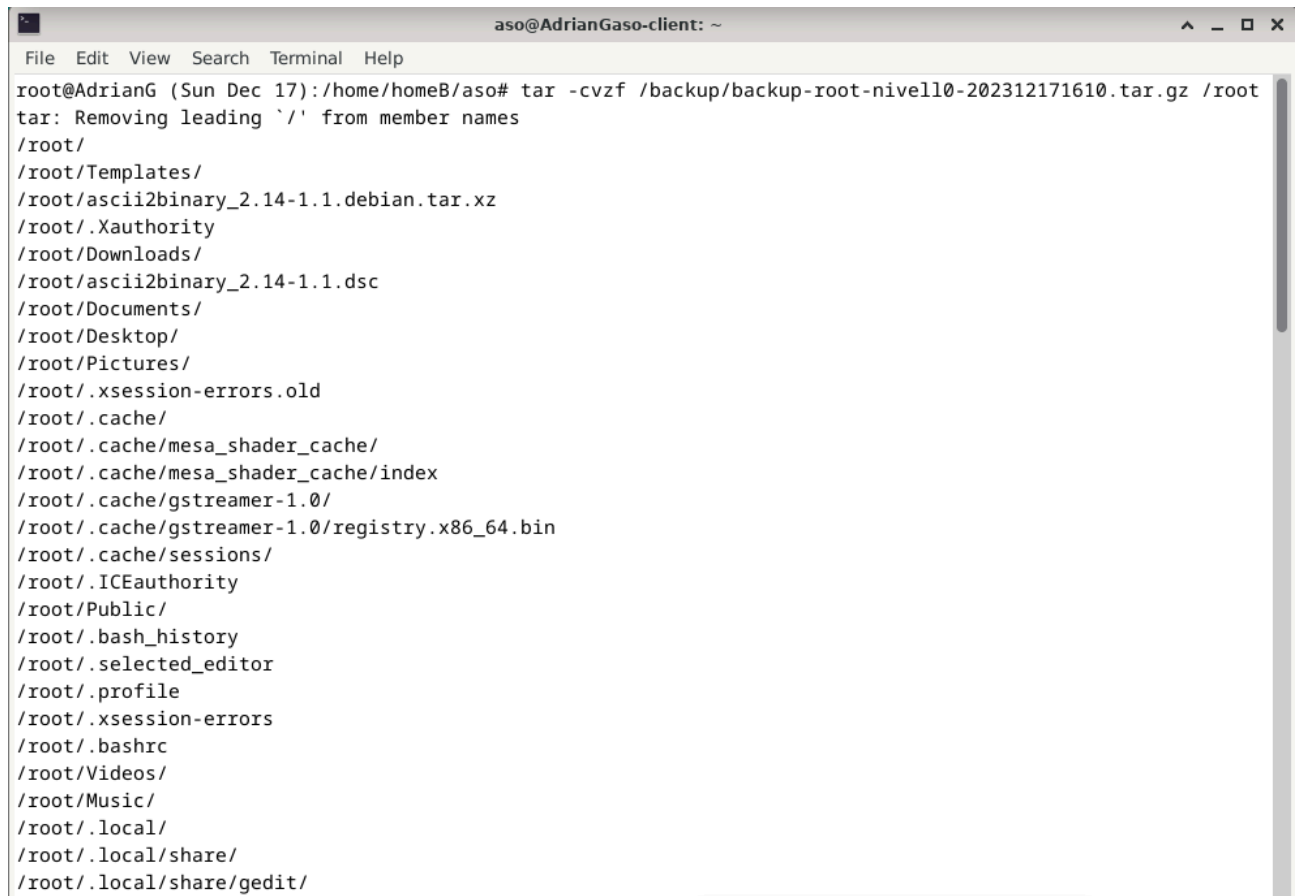
3. Realització de còpies amb TAR

3.1. Realització de còpies completes

Realitzeu una còpia completa del directori /root (comproveu que existeixen fitxers en aquest directori) amb la comanda **tar**. Useu noms significatius per als fitxers de *backup*:

feu que el nom del fitxer tingui informació sobre el contingut del fitxer, la data i hora en que es va fer la còpia, i si la còpia és completa o incremental, i el nivell de la còpia (0, 1, ...).

Utilitzem la comanda `"tar -cvzf /backup/backup-root-nivell0-202312171610.tar.gz (directori on ho volem guardar) /root (carpeta que volem comprimir)"`.



```
aso@AdrianGaso-client: ~  
File Edit View Search Terminal Help  
root@AdrianG (Sun Dec 17):/home/homeB/aso# tar -cvzf /backup/backup-root-nivell0-202312171610.tar.gz /root  
tar: Removing leading '/' from member names  
/root/  
/root/Templates/  
/root/ascii2binary_2.14-1.1.debian.tar.xz  
/root/.Xauthority  
/root/Downloads/  
/root/ascii2binary_2.14-1.1.dsc  
/root/Documents/  
/root/Desktop/  
/root/Pictures/  
/root/.xsession-errors.old  
/root/.cache/  
/root/.cache/mesa_shader_cache/  
/root/.cache/mesa_shader_cache/index  
/root/.cache/gstreamer-1.0/  
/root/.cache/gstreamer-1.0/registry.x86_64.bin  
/root/.cache/sessions/  
/root/.ICEauthority  
/root/Public/  
/root/.bash_history  
/root/.selected_editor  
/root/.profile  
/root/.xsession-errors  
/root/.bashrc  
/root/Videos/  
/root/Music/  
/root/.local/  
/root/.local/share/  
/root/.local/share/gedit/
```

Com es pot fer perquè el nom del fitxer de *backup* inclogui automàticament la data del *backup*? per exemple que sigui *backup-etc-nivell0-202112041030* (any mes dia hora minut segon). Utilitza la comanda **date**.

Utilitzem la comanda

`"tar -cvzf /backup/backup-root-nivell0-date-$(date +%y-%m-%d-%T).tar.gz /root"`.


```

root@AdrianG (Sun Dec 17):/home/homeB/aso# tar -cvzf /backup/backup-root-nivell0-date-$(date +%y-%m-%d-%T).tar.gz /root
bash: date +%y-%m-%d-%T: command not found
tar: Removing leading '/' from member names
/root/
/root/Templates/
/root/ascii2binary_2.14-1.1.debian.tar.xz
/root/.Xauthority
/root/Downloads/
/root/ascii2binary_2.14-1.1.dsc
/root/Documents/
/root/Desktop/
/root/Pictures/
/root/.xsession-errors.old
/root/.cache/
/root/.cache/mesa_shader_cache/
/root/.cache/mesa_shader_cache/index

root@AdrianG (Sun Dec 17):/backup# ls
backup-root-nivell0-202312171610.tar.gz  backup-root-nivell0-date-23-12-17-15:17:08.tar.gz  lost+found
root@AdrianG (Sun Dec 17):/backup#

```

Quina comanda heu fet servir per fer la còpia completa del directori /etc?

`tar -cvf /backup/backup-etc-nivell0-$(date +%y-%m-%d-%T).tar.gz /etc`

```

root@AdrianG (Sun Dec 17):/backup# tar -cvzf /backup/backup-etc-nivell0-$(date +%y-%m-%d-%T).tar.gz /etc
tar: Removing leading '/' from member names
/etc/
/etc/libn1-3/
/etc/libn1-3/pktloc
/etc/libn1-3/classid
/etc/xdg/
/etc/xdg/systemd/
/etc/xdg/systemd/user
/etc/xdg/user-dirs.defaults
/etc/xdg/tumbler/
/etc/xdg/tumbler/tumbler.rc
/etc/xdg/user-dirs.conf
/etc/xdg/Thunar/
/etc/xdg/Thunar/uca.xml
/etc/xdg/plasma-workspace/
/etc/xdg/plasma-workspace/env/
/etc/xdg/plasma-workspace/env/env.sh
/etc/xdg/kickoffrc

root@AdrianG (Sun Dec 17):/backup# ls
backup-etc-nivell0-23-12-17-15:19:08.tar.gz  backup-root-nivell0-date-23-12-17-15:17:08.tar.gz
backup-root-nivell0-202312171610.tar.gz      lost+found
root@AdrianG (Sun Dec 17):/backup# █

```

Per què no és aconsellable comprimir el fitxer de *backup*?

Comprimir un fitxer de backup no és aconsellable principalment perquè pot augmentar el risc de corrupció de dades: si l'arxiu comprimit es danya, es podrien perdre totes les dades que conté. A més, la compressió pot complicar la restauració selectiva de dades, ja que caldria descomprimir tot l'arxiu per accedir a parts específiques. També, els processos de compressió i descompressió requereixen temps i recursos addicionals, el que pot ser inconvenient en situacions d'urgència on el temps de recuperació és crític.

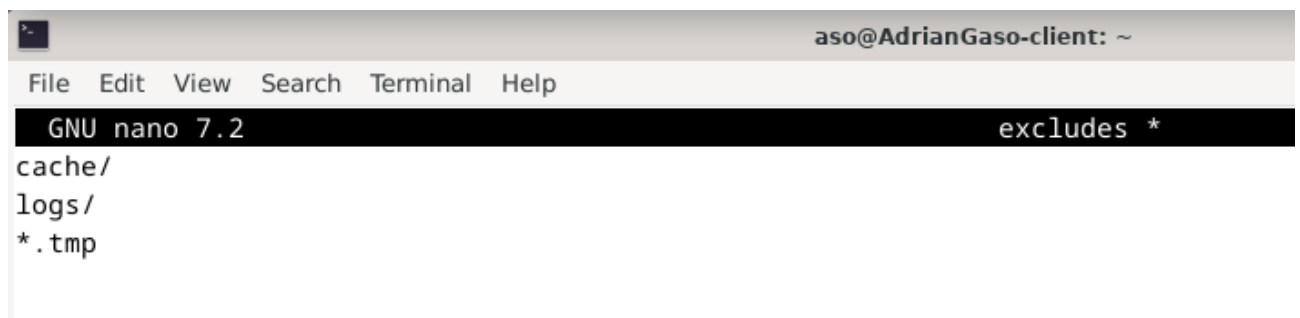
Si volguéssim comprimir el fitxer de *backup* quina opció afegiríem a la comanda *tar*?

Fariem `"tar -cvzf backup.tar.gz /backup"`. La *z* en permet comprimir en *gzip* el fitxer */backup*

A vegades quan es realitza la còpia completa es requereix excloure certs fitxers. Per això es pot construir un fitxer amb una llista de fitxers que no haurien de ser al *backup*.

Feu de nou la copia completa però aquesta ocasió excloent el fitxers que siguin al fitxer *excludes*. (poseu el nom d'alguns fitxers al fitxer *excludes*). Quina opció de **tar** permet excloure un llistat de fitxers del *backup*?

Creem el fitxer *excludes* i fiquem alguns arxius de prova.

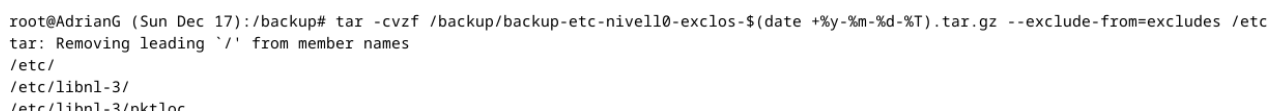


```
File Edit View Search Terminal Help
GNU nano 7.2 excludes *
cache/
logs/
*.tmp
```

Aquest llistat exclouria tots els continguts dins dels directoris *cache* i *logs*, i qualsevol fitxer que acabi amb *.tmp*.

Ara utilitzem la següent comanda.

`tar -cvzf /backup/backup-etc-nivell0-exclos-$(date +%y-%m-%d-%T).tar.gz. --exclude-from=excludes /etc`



```
root@AdrianG (Sun Dec 17):/backup# tar -cvzf /backup/backup-etc-nivell0-exclos-$(date +%y-%m-%d-%T).tar.gz --exclude-from=excludes /etc
tar: Removing leading '/' from member names
/etc/
/etc/libn1-3/
/etc/libn1-3/nk1nc
```

A més a més de protegir el directori de les còpies de seguretat és importat utilitzar algun mecanisme que permeti verificar que el fitxers de *backup* no hagin estat modificats després d'haver-los creat. Per això es comú utilitzar algun mecanisme de signatura digital, com son els *hashs* SHA, que permeten verificar la integritat d'un fitxer.

Una vegada hagueu realitzat la còpia, utilitzeu la comanda **sha512sum**

(utilitzeu el **man** per saber com fer servir aquesta comanda) amb la còpia del directori i guardeu-vos el resultat en un fitxer *<nomdelacopia>.asc*.

Utilitzem la comanda sha512sum i fem:

```
sha512sum backup-etc-nivell0-23-12-17-15\19\08.tar.gz > backup-etc-nivell0-23-12-17-15\19\08.tar.gz.asc
```

```
root@AdrianG (Sun Dec 17):/backup# sha512sum backup-etc-nivell0-23-12-17-15\19\08.tar.gz > backup-etc-nivell0-23-12-17-15\19\08.tar.gz.asc
root@AdrianG (Sun Dec 17):/backup# ls
backup-etc-nivell0-23-12-17-15:19:08.tar.gz      backup-root-nivell0-202312171610.tar.gz      lost+found
backup-etc-nivell0-23-12-17-15:19:08.tar.gz.asc  backup-root-nivell0-date-23-12-17-15:17:08.tar.gz
backup-etc-nivell0-exclos-23-12-17-15:31:29.tar.gz  excludes
root@AdrianG (Sun Dec 17):/backup#
```

3.2. Realització de còpies incrementals

Per tal de dur a terme còpies incrementals, serà necessari modificar alguns fitxers dins el directori /root :

Modificacions:

Genereu nous fitxers i subdirectoris

Modifiquen el contingut d'alguns fitxers

Useu la comanda **touch** per canviar la data de modificació d'alguns fitxers.

Per fer còpies incrementals amb tar tenim l'opció **--newer** que només inclou els fitxers que hagin estat modificats des d'una data determinada. Aquesta data es pot especificar de dues formes: la primera, posant-la directament, per exemple: "**--newer="2021-11-28 12:10"**". La segona manera consisteix en agafar la data d'un fitxer, això vol dir que la data serà la de l'última modificació del fitxer, per exemple: "**--newer=./file**".

Ara realitzeu una còpia incremental del directori /root respecte a la còpia completa que heu fet abans usant la comanda tar. Feu també un **sha512sum** i guardeu-lo en un arxiu.asc amb el nom diferent al nom que heu utilitzat abans.

Hem creat aquests arxius i fitxers:

```

root@francesco0 (Fri Dec 08):<~># mkdir prova_training5
root@francesco0 (Fri Dec 08):<~># cd Templates/
root@francesco0 (Fri Dec 08):<~/Templates># ls
root@francesco0 (Fri Dec 08):<~/Templates># nano new_file
root@francesco0 (Fri Dec 08):<~/Templates># ls
new_file
root@francesco0 (Fri Dec 08):<~/Templates># cd ..
root@francesco0 (Fri Dec 08):<~># nano file_new

```

```

root@francesco0 (Fri Dec 08):<~/Pictures># ls
firefox.jpeg

```

Hem modificat el contingut d'aquest arxiu:

```

root@francesco0 (Fri Dec 08):<~/Downloads># nano borrar_user.sh
root@francesco0 (Fri Dec 08):<~/Downloads># █

```

Hem canviat la data de modificació dels 2 arxius *.tar* amb la comanda touch:

```

root@francesco0 (Fri Dec 08):<~/Downloads># ls -la
total 124
drwxr-xr-x  2 root root  4096 Dec  8 11:42 .
drwx----- 17 root root  4096 Dec  8 09:42 ..
-rw-r--r--  1 root root 101885 Oct 18 22:38 ascii2binary_2.14.orig.tar.gz
-rw-r--r--  1 root root   584 Dec  8 11:42 borrar_user.sh
-rw-r--r--  1 root root 10940 Dec  8 11:45 calc_2.15.0.1-1.debian.tar.xz
root@francesco0 (Fri Dec 08):<~/Downloads># touch ascii2binary_2.14.orig.tar.gz
root@francesco0 (Fri Dec 08):<~/Downloads># ls -la
total 124
drwxr-xr-x  2 root root  4096 Dec  8 11:42 .
drwx----- 17 root root  4096 Dec  8 09:42 ..
-rw-r--r--  1 root root 101885 Dec  8 11:49 ascii2binary_2.14.orig.tar.gz
-rw-r--r--  1 root root   584 Dec  8 11:42 borrar_user.sh
-rw-r--r--  1 root root 10940 Dec  8 11:45 calc_2.15.0.1-1.debian.tar.xz
root@francesco0 (Fri Dec 08):<~/Downloads># █

```

Imatge de la comanda utilitzada per fer la còpia incremental:

```

root@francesco0 (Fri Dec 08):<~># tar -cvf incremental_backup_root.tar --newer="2023-12-8 10:00" /root/
tar: Option --after-date: Treating date '2023-12-8 10:00' as 2023-12-08 10:00:00
tar: Removing leading '/' from member names
/root/
tar: /root/incremental_backup_root.tar: file is the archive; not dumped
/root/Pictures/

```

Imatge de la còpia incremental:

```

root@francesco0 (Fri Dec 08):<~># ls
Desktop Documents Downloads Music Pictures Public Templates Videos file_new incremental_backup_root.tar prova_training5
root@francesco0 (Fri Dec 08):<~># █

```

Si volem guardar el resultat del checksum del backup.tar podem direccionar-lo amb '>' a l'arxiu que volguem:

```

root@francesco0 (Fri Dec 08):<~># sha512sum -t incremental_backup_root.tar > incremental_backup_root.asc
root@francesco0 (Fri Dec 08):<~># ls
Desktop  Downloads  Pictures  Templates  file_new                                incremental_backup_root.tar
Documents Music      Public    Videos    incremental_backup_root.asc             prova_training5
root@francesco0 (Fri Dec 08):<~># cat incremental_backup_root.asc
1d2f0c712525eb915c19ae4eb27fb9e4e5b300d44198577190ad2296a8bf888021046d353f875560135bebe351641e9d4d165e5aa887af337558c4b1fb36eedc incremental_backup_
root.tar
root@francesco0 (Fri Dec 08):<~># █

```

Quin problema potencial hi ha al utilitzar el fitxer .tar de la còpia completa per obtenir la data del backup per fer la còpia incremental? Com es pot resoldre aquest problema?

L'ús del fitxer .tar de la còpia completa per obtenir la data del backup per realitzar una còpia incremental pot presentar diversos problemes potencials:

1. Si el fitxer .tar es corromp o es modifica de qualsevol manera, això pot afectar la capacitat de determinar correctament la data de l'última còpia de seguretat.
2. Obtenir la data de la última còpia de seguretat a partir d'un fitxer .tar pot requerir que es descomprimeixi o es llegeixi el fitxer complet, la qual cosa pot ser molt ineficient, especialment si el fitxer és gran.
3. Si hi ha múltiples còpies de seguretat que s'executen en paral·lel o si el fitxer .tar s'actualitza o es modifica després de crear la còpia incremental, pot haver-hi inconsistències en les dades.

Aquests problemes es poden resoldre de diferents maneres:

1. Utilitzar les metadades (autor, data de creació, data de modificació, mida del fitxer) del sistema de fitxers per obtenir la data de la última modificació dels fitxers.
2. Mantenir un registre separat de la data de l'última còpia de seguretat en un fitxer o base de dades.
3. Utilitzar eines de backup que suportin còpies de seguretat incrementals de manera nativa i que puguin manejar la lògica de determinació de quins fitxers han canviat des de l'última còpia de seguretat.
4. Fer servir checksums per validar la integritat dels fitxers abans de realitzar una còpia incremental, assegurant-se que els fitxers no s'han corromput o modificat de manera inesperada.
5. Considerar la possibilitat d'utilitzar còpies de seguretat basades en blocs en lloc de còpies basades en fitxers, la qual cosa pot ser més eficient i precisa per a determinar canvis.

Realitzeu una segona ronda de modificacions al directori /root per tal de provocar una segona còpia incremental:

Modificacions:

Genereu nous fitxers

Modifiqueu el contingut d'alguns fitxers

Useu la comanda touch per canviar la data de modificació d'alguns fitxers.

Esborreu algun dels fitxers que heu generat per la primera còpia incremental anterior.

Realitzeu una segona còpia incremental del directori /root (respecte la primera còpia incremental) amb la comanda **tar**. També feu un **sha512sum** de la segona còpia incremental i poseu-li un nom apropiat.

Aquí hi ha una imatge de les modificacions que he fet per aquesta segona còpia incremental:

```
root@francesco0 (Sat Dec 09):<~># rm -rf prova_training5/
root@francesco0 (Sat Dec 09):<~># ls
Desktop Documents Downloads Music Pictures Public Templates Videos file_new incremental_backup_root.asc incremental_backup_root.tar
root@francesco0 (Sat Dec 09):<~># cd Documents/
root@francesco0 (Sat Dec 09):<~/Documents># ls
root@francesco0 (Sat Dec 09):<~/Documents># nano empty_file
root@francesco0 (Sat Dec 09):<~/Documents># ls
empty_file
root@francesco0 (Sat Dec 09):<~/Documents># cd ..
root@francesco0 (Sat Dec 09):<~># cd Pictures/
root@francesco0 (Sat Dec 09):<~/Pictures># ls
firefox.jpeg
root@francesco0 (Sat Dec 09):<~/Pictures># mkdir training5_photos
root@francesco0 (Sat Dec 09):<~/Pictures># ls training5_photos/
root@francesco0 (Sat Dec 09):<~/Pictures># nano text
root@francesco0 (Sat Dec 09):<~/Pictures># cd
root@francesco0 (Sat Dec 09):<~># cd Templates/
root@francesco0 (Sat Dec 09):<~/Templates># ls
new_file
root@francesco0 (Sat Dec 09):<~/Templates># nano new_file
root@francesco0 (Sat Dec 09):<~/Templates># cd
root@francesco0 (Sat Dec 09):<~># cd Pictures/
root@francesco0 (Sat Dec 09):<~/Pictures># ls
firefox.jpeg text training5_photos
root@francesco0 (Sat Dec 09):<~/Pictures># touch firefox.jpeg
root@francesco0 (Sat Dec 09):<~/Pictures># ls-la
bash: ls-la: command not found
root@francesco0 (Sat Dec 09):<~/Pictures># ls -la
total 24
drwxr-xr-x  3 root root 4096 Dec  9 11:47 .
drwx----- 16 root root 4096 Dec  9 11:44 ..
-rw-r--r--  1 root root 5881 Dec  9 11:48 firefox.jpeg
-rw-r--r--  1 root root   80 Dec  9 11:47 text
```

Per poder fer la còpia del root, sense que aquest tingui la còpia anterior, hem creat una carpeta backups fora del directori root, i hem mogut el .tar i .asc de la 1a còpia incremental:

```

root@francesco0 (Mon Dec 11):</># mkdir backups
root@francesco0 (Mon Dec 11):</># ls
backups  dev      initrd.img      lib32  lost+found  opt    run    sys    var
bin      etc      initrd.img.old  lib64  media      proc   sbin   tmp    vmlinuz
boot     home    lib             libx32 mnt         root   srv    usr    vmlinuz.old
root@francesco0 (Mon Dec 11):</># mv -f /root/incremental_backup_root.tar /backu
ps/
root@francesco0 (Mon Dec 11):</># mv -f /root/incremental_backup_root.asc /backu
ps/
root@francesco0 (Mon Dec 11):</># ls /backups/
incremental_backup_root.asc  incremental_backup_root.tar
root@francesco0 (Mon Dec 11):</>#

```

Segona còpia incremental:

```

root@francesco0 (Mon Dec 11):<~># tar -cvf incremental_backup2_root.tar --newer="2023-12-9 10:00" /root/
tar: Option --after-date: Treating date '2023-12-9 10:00' as 2023-12-09 10:00:00
tar: Removing leading '/' from member names
/root/
tar: /root/incremental_backup_root.tar: file is unchanged; not dumped
/root/Pictures/

```

Imatge dels nous fitxers, .tar i .asc (checksum): els movem a la carpeta de backups per poder verificar que el contingut sigui el mateix.

```

root@francesco0 (Mon Dec 11):<~># sha512sum incremental_backup2_root.tar > incremental_backup2_root.asc
root@francesco0 (Mon Dec 11):<~># ls
Desktop Documents Downloads Music Pictures Public Templates Videos file_new incremental_backup2_root.asc incremental_backup2_root.tar
root@francesco0 (Mon Dec 11):<~># cat incremental_backup2_root.asc
ea11ffe57ed4f25a7a95eb98ea20b472d98c46a5752d45398bbdbdea424c7f097ec6d3e2b1021cbbca03f1927560ee815d8410047025dcde81633a0ded45fa6e incremental_backup2
_root.tar
root@francesco0 (Mon Dec 11):<~># cd ..
root@francesco0 (Mon Dec 11):</># mv -f /root/incremental_backup2_root.tar /backups/
root@francesco0 (Mon Dec 11):</># mv -f /root/incremental_backup2_root.asc /backups/
root@francesco0 (Mon Dec 11):</># ls /backups/
incremental_backup2_root.asc  incremental_backup2_root.tar  incremental_backup_root.asc  incremental_backup_root.tar
root@francesco0 (Mon Dec 11):</># ls /root/
Desktop Documents Downloads Music Pictures Public Templates Videos file_new
root@francesco0 (Mon Dec 11):</># █

```

Com es pot verificar que el contingut del fitxer de *backup* sigui el mateix que el directori que s'ha copiat?

Per assegurar-nos que el contingut del fitxer de còpia és idèntic al directori original, hem de descomprimir la còpia incremental en un directori nou i comparar-lo amb el directori /root. Ho podem fer amb la comanda diff.

Per fer el diff utilitzem 2 opcions (-q i -r, el -q farà que només ens digui les diferències i el -r farà que miri dins els subdirectoris):


```

root@francesco0 (Mon Dec 11):~/comparar_backups># ls
incremental_backup2_root.tar root
root@francesco0 (Mon Dec 11):~/comparar_backups># cd
root@francesco0 (Mon Dec 11):<~># cd ..
root@francesco0 (Mon Dec 11):<~># diff -qr /root/ /comparar_backups/root/
Only in /root/: .ICEauthority
Only in /root/: .bashrc
Only in /root/.cache/gstreamer-1.0: registry.x86_64.bin.tmpC1QJF2
Only in /root/.cache/mesa_shader_cache: index
Only in /root/.cache/mozilla/firefox/329v340s.default-esr/cache2: ce_T151c2VyQ29udGV4dElkPTUs
Only in /root/.cache/mozilla/firefox/329v340s.default-esr/cache2: ce_T151c2VyQ29udGV4dElkPTUsYSw=
Only in /root/.cache/mozilla/firefox/329v340s.default-esr/cache2/entries: 00099279F4E23512F2798630BF151B609CB93793
Only in /root/.cache/mozilla/firefox/329v340s.default-esr/cache2/entries: 0021ED275A50F7CC444C9E8CC9D4F713DD8740DF
Only in /root/.cache/mozilla/firefox/329v340s.default-esr/cache2/entries: 0061994E21E8C0FAF461CF574807A33EE83067BF
Only in /root/.cache/mozilla/firefox/329v340s.default-esr/cache2/entries: 0091866340353D0575851D16AEB618E2AFA429C6
Only in /root/.cache/mozilla/firefox/329v340s.default-esr/cache2/entries: 00B877BC7A20E68C735D09FE5E8D99560575A406
Only in /root/.cache/mozilla/firefox/329v340s.default-esr/cache2/entries: 00E4834B3CFBFCEDD2D78F80B61EE5955176910F
Only in /root/.cache/mozilla/firefox/329v340s.default-esr/cache2/entries: 00E796C2BFC63FBBC014992122775DC851A3D71D
Only in /root/.cache/mozilla/firefox/329v340s.default-esr/cache2/entries: 00EE82B78034761745E35DCA753784C4F831709E
Only in /root/.cache/mozilla/firefox/329v340s.default-esr/cache2/entries: 010B03F798DD3AB27726A3840E7A8CE2A7DABAB8
Only in /root/.cache/mozilla/firefox/329v340s.default-esr/cache2/entries: 013DE866275E0B8041BCF19A79393FE4E457492C
Only in /root/.cache/mozilla/firefox/329v340s.default-esr/cache2/entries: 014C98341EB1374763C7D4C2BC02A7FA5C93DF6A
Only in /root/.cache/mozilla/firefox/329v340s.default-esr/cache2/entries: 01738B5E4C7C9B6EDCABC2B31AE80ADC0A2E7BD2
Only in /root/.cache/mozilla/firefox/329v340s.default-esr/cache2/entries: 017BE3C98BFA6DF51F0991F9D11ADAA2672ADE
Only in /root/.cache/mozilla/firefox/329v340s.default-esr/cache2/entries: 01A870F86316CBF49A03F0CF3E3062B9F57FA808
Only in /root/.cache/mozilla/firefox/329v340s.default-esr/cache2/entries: 01AC7085C6FA9BE831895894125CEE11241A0688
Only in /root/.cache/mozilla/firefox/329v340s.default-esr/cache2/entries: 01B324FBE6C5C939857D76B1217BA5E8F0F395D6
Only in /root/.cache/mozilla/firefox/329v340s.default-esr/cache2/entries: 01B788380BD3A5C1B8721EEE3FAF826B08AD2560
Only in /root/.cache/mozilla/firefox/329v340s.default-esr/cache2/entries: 01C1BDD9D90EE6DA96637A5FDB641802B2712D8F
Only in /root/.cache/mozilla/firefox/329v340s.default-esr/cache2/entries: 01EB990468ADE71D71B961F9D06A17D7C1B4EF79

```

Quan fem el diff podem veure que hi ha moltíssimes diferències ja que a la còpia incremental només hem guardat els canvis nous que hem fet després de la primera còpia incremental.

Però podem comprovar que ha guardat els canvis fets després d'una data en concret (després de la primera còpia incremental):

```

root@francesco0 (Mon Dec 11):<~># cd /comparar_backups/root/
root@francesco0 (Mon Dec 11):~/comparar_backups/root># ls
Desktop Documents Downloads Music Pictures Public Templates Videos
root@francesco0 (Mon Dec 11):~/comparar_backups/root># cd Desktop/
root@francesco0 (Mon Dec 11):~/comparar_backups/root/Desktop># ls
root@francesco0 (Mon Dec 11):~/comparar_backups/root/Desktop># cd ..
root@francesco0 (Mon Dec 11):~/comparar_backups/root># cd /Doc
bash: cd: /Doc: No such file or directory
root@francesco0 (Mon Dec 11):~/comparar_backups/root># ls Documents/
empty_file
root@francesco0 (Mon Dec 11):~/comparar_backups/root># ls Downloads/
root@francesco0 (Mon Dec 11):~/comparar_backups/root># ls Pictures/
firefox.jpeg text training5_photos
root@francesco0 (Mon Dec 11):~/comparar_backups/root># ls Templates/
new_file
root@francesco0 (Mon Dec 11):~/comparar_backups/root># █

```

Com es pot verificar, fent ús de la comanda **sha512sum**, la integritat d'una còpia de seguretat, és a dir, que el fitxer no ha estat modificat des que es va realitzar la còpia?

Per verificar la integritat hem de calcular el hash SHA-512 del fitxer de còpia, hem de comparar el nou hash amb el hash original. Si els dos coincideixen, la còpia no ha estat modificada. Si són diferents, el fitxer pot haver estat modificat o danyat.

Ho podem fer amb la comanda diff, creem un arxiu (en .asc) que contingui el resultat d'aquest checksum just després d'haver fet la còpia incremental, i quan vulguem comprovar si s'ha modificat el backup, tornem a fer el checksum i el comparem amb el primer checksum que vem fer. Si hi ha qualsevol diferència vol dir que ha estat modificat o

danyat.

3.3. Restauració d'una còpia de seguretat

Reanomenem el directori /root per /root.old per simular l'efecte que es produiria si esborréssim el directori.

Per a renombrar el directori root utilitzarem la comanda mv: mv /root /root.old, aixó modificarà el nom de la nostre carpeta

En quin ordre cal restaurar els fitxers per tal que el resultat final sigui el desitjat?

L'ordre que s'ha de seguir es:

1. La copia completa
2. La copia incremental més antiga possible
3. La copia incremental més nova possible

D'aquesta manera no eliminarem ningú arxiu o el sobreescrivem de manera erronea

Ara restaureu la còpia de seguretat del directori /root, la qual cosa implica restaurar els tres fitxers que hem creat: la còpia completa i les dos incrementals.

Per a restaurar les nostres còpies tindrem que descomprimir en l'ordre que hem dit anteriorment a la carpeta on es trobava el directori /root.

A continuació descomprimem primer la copia completa, utilitzarem la comanda tar:
tar -xvf /backup/[nom_del_backup_complet]

```
| root@MarcR (Tue Dec 12):<~># tar -xvf /backup/backup-root-nivell0-20231212113937 |
```

```
root/.mozilla/firefox/lkcwiutx.default-esr/security_state/
root/.mozilla/firefox/lkcwiutx.default-esr/bookmarkbackups/
root/.mozilla/firefox/tx9rut44.default/
root/.mozilla/firefox/tx9rut44.default/times.json
root/Downloads/
root/Pictures/
root/.xsession-errors.old
root/Music/
root/.ssh/
root/.ssh/known_hosts
root/.ssh/id_rsa
root/.ssh/known_hosts.old
root/.ssh/id_rsa.pub
root/.viminfo
root/Public/
root/Videos/
root/.local/
root/.local/share/
root/.local/share/icc/
root/.local/share/nano/
root/.xsession-errors
root/.dmrc
root/.Xauthority
root/text.txt
```

Una vegada descomprimit farem el tar de les altres dues incrementals en ordre de més vell a més nou:

```
tar -xvf /backup/[nom_del_backup_incremental1]
```

```
tar -xvf /backup/[nom_del_backup_incremental2]
```

```
root@MarcR (Tue Dec 12):<~># tar -xvf /backup/backup-root-nivell1-20231212115552
```

```
77ntouromlalnody--naod.files/  
root/.mozilla/firefox/lkcwiutx.default-esr/storage/permanent/chrome/idb/35612888  
49sdhlie.files/  
root/.mozilla/firefox/lkcwiutx.default-esr/storage/permanent/chrome/idb/38701127  
24rsegmnoittet-es.files/  
root/.mozilla/firefox/lkcwiutx.default-esr/storage/permanent/chrome/idb/16571145  
95AmcateirvtiSty.files/  
root/.mozilla/firefox/lkcwiutx.default-esr/storage/permanent/chrome/idb/14513188  
68ntouromlalnody--epcr.files/  
root/.mozilla/firefox/lkcwiutx.default-esr/crashes/  
root/.mozilla/firefox/lkcwiutx.default-esr/crashes/events/  
root/.mozilla/firefox/lkcwiutx.default-esr/security_state/  
root/.mozilla/firefox/lkcwiutx.default-esr/bookmarkbackups/  
root/.mozilla/firefox/tx9rut44.default/  
root/Downloads/  
root/Pictures/  
root/Music/  
root/.ssh/  
root/Public/  
root/Videos/  
root/.local/  
root/.local/share/  
root/.local/share/icc/  
root/.local/share/nano/
```

Què ha passat amb els fitxers que havíeu esborrat abans de fer la segona còpia incremental? Com es pot detectar que aquest fitxers han estat esborrats? Quan seran esborrats de les còpies de seguretat?

Els fitxers que s'han borrrat en la segona copia incremental també s'han borrrat una vegada hem descomprimit l'última copia de seguretat. Podem detectar que s'han borrrat fent ús de la comanda [find] o comparant les dues còpies de seguretat amb un [diff]. Si la política de retenció no conserva les versions antigues, els fitxers eliminats es perdran en la propera rotació o neteja de les còpies de seguretat, ja que aquestes versions antigues s'eliminaran per alliberar espai per a les noves còpies.

3.4. Restauració d'un fragment

Reanomenau un dels subdirectoris dins del directori /root per simular l'efecte que es produiria si esborréssim el subdirectori.

```
root@MarcR (Tue Dec 12):<~># mv Music/ Musica  
root@MarcR (Tue Dec 12):<~># █
```

Restaureu només aquest directori a partir de la còpia de seguretat que hem fet amb **tar**. Això implica restaurar únicament aquest subdirectori a partir dels tres fitxers que hem

creat: la còpia completa i les dos incrementals.

Per a obtenir només la carpeta que hem modificat, en el meu cas la de Music, utilitzarem la següent comanda:

```
root@MarcR (Tue Dec 12):</tmp/root># tar -xvf /backup/backup-root-nivell0-20231212113937 --wildcards '*/Music/' -C /tmp/root
root/Music/
```

Una vegada haguem fet això amb la completa, si en les incrementals hem modificat la carpeta en algun moment tindrem que fer el mateix però amb els dos altres tars.

4. Realització de còpies usant RSYNC

Fins ara hem vist com fer *backups* i guardar-los en la mateixa màquina, però el més comú és tenir diferents màquines en què fem *backups* i una altra màquina en xarxa que emmagatzema aquests *backups*. Per fer això tenim la comanda **rsync** que permet copiar un directori (o un conjunt de fitxers) a un altre directori a través de una connexió de xarxa. Per això **rsync** usa un algorisme de *checksum* eficient per transmetre només les diferències entre els dos directoris i al mateix temps comprimeix els fitxers per fer més ràpida la transmissió de dades. Aquesta eina permet copiar fitxers des de o cap a un directori situat en una màquina remota, o entre directoris de la mateixa màquina. El que no permet és copiar directoris entre dos màquines remotes. A més a més **rsync** permet copiar enllaços, dispositius, i preservar permisos, grups i propietaris. També suporta llistats de exclusió i connexió remota amb *shell* segur (ssh) entre altres possibilitats. Per a més informació veure **man rsync**.

```
root@marionaF (Tue Dec 12):<~># apt-get install rsync
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  bsdmainutils libcupsfilters1 libcupsimage2 libflac8 libgs9-common libicu63
  libilmbase23 libldap-2.4-2 libmpdec2 libnetpbm10 libopenexr23 libperl5.28
  libpython3.7-minimal libpython3.7-stdlib libreadline7 libruby2.5 libtiff5
```

4.1. Realització de backups a través d'una xarxa

Com ja hem dit abans, **rsync** permet fer còpies en una màquina remota. Això es podria fer utilitzant **rsh** o posant **rsync** en mode servidor, però això pot ser perillós perquè en una xarxa local alguna altra màquina podria estar "escoltant" la connexió i podria agafar informació confidencial. Per aquesta raó, **rsync** permet fer connexions segures amb **ssh**. Per realitzar les nostres proves utilitzarem la nostra pròpia màquina com servidor ssh. En

primer lloc inicialitzeu el *daemon* de **ssh**.

Creeu un directori per fer les còpies en la partició de *backups* i després feu la següent comanda:

```
$ rsync -avz /root -e ssh root@localhost:/backup/backup-rsync/
```

Nota: Perquè l'anterior comanda funcioni bé és necessari activar el compte del root i posar-li una contrasenya vàlida. Recordeu instal·lar també el paquet ssh si us cal.

Quin és el significat de les opcions -avz de l'**rsync**?

-a (o --archive): Aquesta opció és una combinació de diverses opcions. Activa el mode arxiu, el qual assegura que la transferència mantingui les propietats dels fitxers (com permisos, propietats, enllaços simbòlics, etc.) intactes. A més, també manté la informació de les carpetes (com per exemple, l'estructura de directoris).

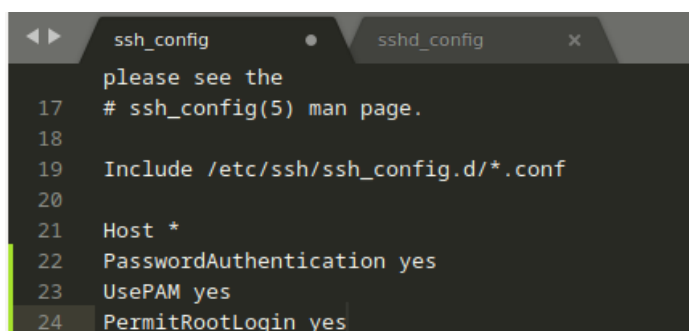
-v (o --verbose): Aquesta opció fa que rsync sigui més xerraire. Proporciona informació detallada sobre la transferència de fitxers, mostrant quins fitxers estan sent transferits i un resum de les estadístiques de la transferència.

-z (o --compress): Aquesta opció habilita la compressió de dades durant la transferència. És útil quan s'estan transferint fitxers a través d'una xarxa amb una banda ampla limitada, ja que pot reduir la quantitat de dades que necessiten ser enviades a través de la xarxa.

Primer el que farem es configurar el ssh a la màquina a la que ens volguem connectar per poder connectar-nos com a root, de normal esta desactivat i només amb una comanda del rsync no podem copiar arxius de superusuaris.

Hem d'executar la comanda **nano /etc/ssh/ssh_config** i afegim les següents instruccions (com a la foto).

PasswordAuthentication yes, UsePAM yes, PermitRootLogin yes



```
ssh_config
sshhd_config
please see the
17 # ssh_config(5) man page.
18
19 Include /etc/ssh/ssh_config.d/*.conf
20
21 Host *
22 PasswordAuthentication yes
23 UsePAM yes
24 PermitRootLogin yes
```

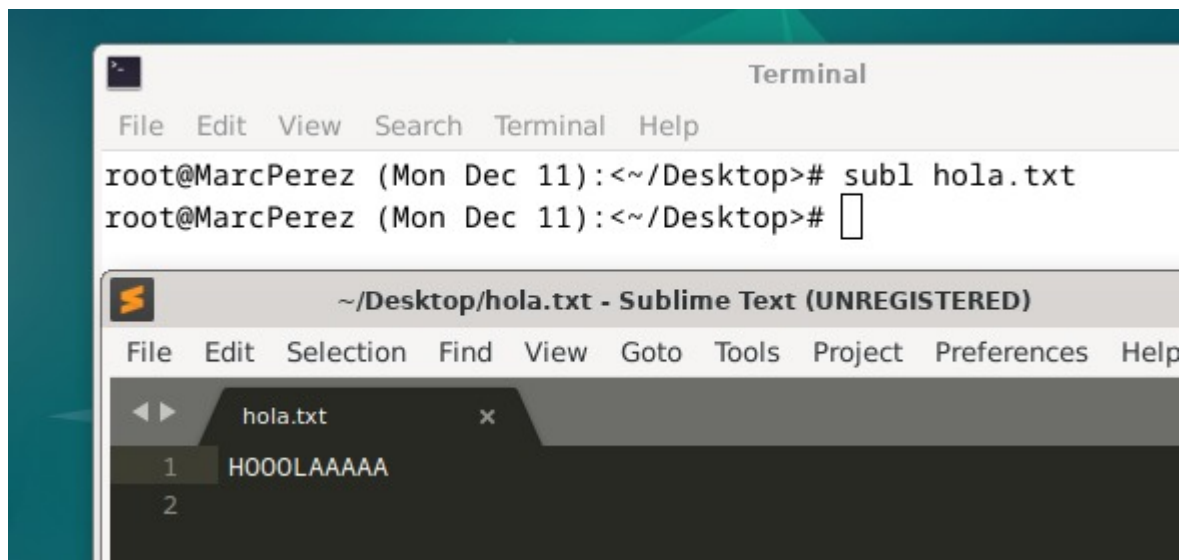
Farem el mateix amb l'altre arxiu anomenat sshd_config. Mirem si estan les instruccions anteriors, i ens asegurem que tinguin el mateix valor que he descrit.

Per acabar executem **service ssh restart** i **service sshd restart**

Creeu un arxiu en el directori root amb:

```
$ echo "nou arxiu" > /root/arxiu_nou.txt
```

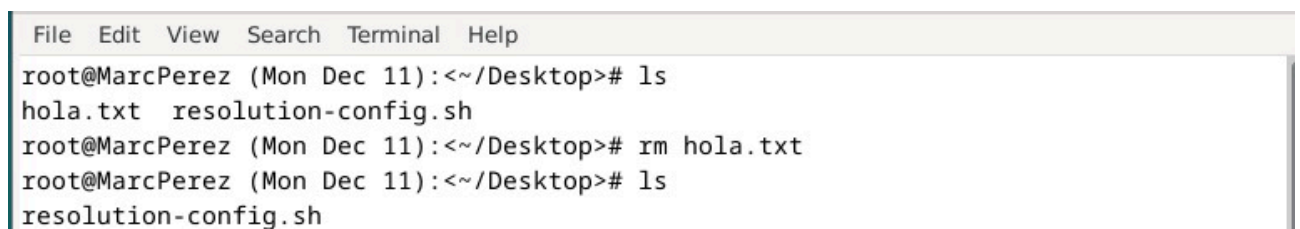
Crearem l'arxiu que anomenarem hola.txt



i torneu a fer el mateix **rsync** d'abans.

```
root@MarcPerez (Mon Dec 11):<~/Desktop># rsync -avz -e "ssh -p 3022" /root/Desktop/hola.txt
root@192.168.0.19:/backups/backup-rsync
```

Ara esborreu l'arxiu que heu creat abans i torneu a sincronitzar.



Què ha passat amb el fitxer esborrat?

S'ha eliminat de la nostra màquina, però seguim mantenint la còpia a la màquina on hem fet el backup amb la comanda anterior (en el nostre cas esta dins un directori anomenat

backup-rsync dins la partició ext3, em montat el directori principal backups).

```
root@MarcPerezBackups (Mon Dec 11):</backups/backup-rsync># ls
hola.txt
```

Amb quin paràmetre podríeu sincronitzar exactament els dos directoris?

Si utilitzem la mateixa comanda que em fet servir abans pero invertida. D'aquesta manera també recuperariem el arxiu.

```
root@MarcPerez (Mon Dec 11):<~/Desktop># rsync -avz -e "ssh -p 3022" root@192.168.0.19:/backups/backup-rsync/hola.txt /root/Desktop/hola.txt
root@192.168.0.19's password:
receiving incremental file list
hola.txt

sent 43 bytes  received 120 bytes  46.57 bytes/sec
total size is 11  speedup is 0.07
root@MarcPerez (Mon Dec 11):<~/Desktop># ls
hola.txt  resolution-config.sh
root@MarcPerez (Mon Dec 11):<~/Desktop># cat hola.txt
H000LAAAAA
root@MarcPerez (Mon Dec 11):<~/Desktop>#
```

En el cas que vulguem copiar tot el directori, i no només un arxiu, hauriem de fer servir la mateixa comanda però sense indicar el arxiu, deixant el directori nomes.

Com faríeu per copiar tots el arxius del directori /home excepte els que tenen extensió .txt?

Utilitzariem la mateixa comanda del rsync, però afegint aquesta opció.

```
--exclude='*.txt'
```

Quina diferencia hi ha entre fer rsync /source /destí i rsync /source/ /destí/?

L'ús del caràcter / al final dels directoris en rsync té un significat especial que diu quins arxius són copiats i on són posats en el directori de destinació:

- **rsync /source /destí:**
 - Sense la barra inclinada al final de /source, rsync copiarà el directori source sencer (inclòs) dins de /destí.
 - Això significa que acabaràs amb /destí/source i tots els arxius i

subdirectoris dins de source estaran dins d'aquest nou directori source.

- **rsync /source/ /destí/:**
 - Amb la barra inclinada al final de /source/, rsync copiarà tots els arxius i subdirectoris *dins* de source al directori destí, però no el directori source en si.
 - Això significa que el contingut de source serà copiat directament dins de destí, sense crear un subdirectori addicional. Així, si source conté file1 i file2, aquests arxius acabaran com /destí/file1 i /destí/file2.

4.2. Realització de còpies incrementals inverses

Com hem vist a l'apartat anterior, cada vegada que realitzem una còpia i sincronitzem, el directori en què tenim el *mirror* queda exactament igual que el directori d'origen. Això és un problema perquè no tenim control dels canvis realitzats. Per solucionar aquest problema podem utilitzar l'opció **--backup i --backup-dir**. Els *backups* generats amb les opcions **--backup i --backup-dir** es diuen inversos perquè la còpia completa es la més recent i no la més antiga com amb **tar**. Amb aquesta opció la còpia completa correspon a la última data en que s'ha fet el *backup* i les incrementals a les dels dies anteriors.

A continuació teniu un script senzill per fer *backups* incrementals amb **rsync**.

Completeu-lo amb les dades que facin falta.

```
#!/bin/bash
SOURCE_DIR=
DEST_DIR=

# excludes file: list of files to exclude
EXCLUDES=

# the name of the backup machine
BSERVER=

# the name of the incremental backups directory
# put a date command for: year month day hour minute second
BACKUP_DATE=
```



```
# options for rsync
OPTS="--ignore-errors --delete-excluded --exclude-from=$EXCLUDES \
--delete --backup --backup-dir=$DEST_DIR/$BACKUP_DATE -av"

# now the actual transfer
rsync $OPTS $SOURCE_DIR root@$BSERVER:$DEST_DIR/complet
```

script complet:

```
#!/bin/bash

# Directori d'origen per fer el backup
SOURCE_DIR="/home/username/documents"

# Directori de destinació per guardar el backup
DEST_DIR="/backup/documents"

# Fitxer que conté la llista d'exclusions
EXCLUDES="/home/username/backup_excludes.txt"

# Nom o adreça IP del servidor de backup
BSERVER="192.168.1.15"

# Directori on es guardaran els backups incrementals
INCREMENTAL_BACKUP_DIR="incrementals"

# Comanda per generar la data actual
BACKUP_DATE=$(date +"%Y-%m-%d_%H-%M-%S")

# Opcions per a rsync
OPTS="--ignore-errors --delete-excluded --exclude-from=$EXCLUDES \
--delete --backup --backup-dir=$DEST_DIR/$INCREMENTAL_BACKUP_DIR/$BACKUP_DATE -av"

# Transferència actual
rsync $OPTS $SOURCE_DIR root@$BSERVER:$DEST_DIR/complet
```

Crear el fitxer per fer el script, en aquest cas es dirà: copia_incremental.sh

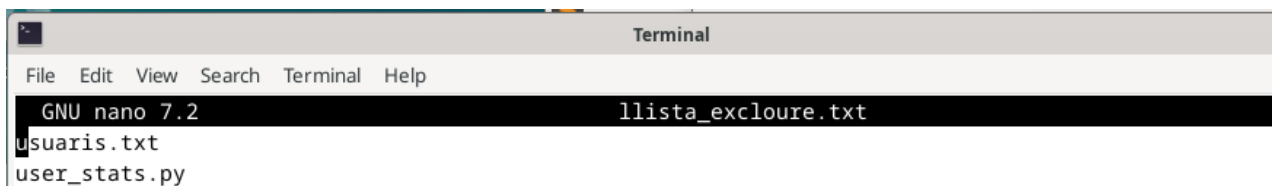
Donar-li les propietats perquè pugui ser executat: `chmod -x copia_incremental.sh`

```
root@marionaF (Sun Dec 17):<~/Documents># subl copia_incremental.sh
root@marionaF (Sun Dec 17):<~/Documents># chmod -x copia_incremental.sh
root@marionaF (Sun Dec 17):<~/Documents># ls
BadUsers.py      asosh.sh          mkht3.py          subprocess        user_stats.sha256sum
BadUsers.sh      copia_incremental.sh mkht3.sh          user_stats.py     usuarios.txt
BadUsers_t6.py   delete_user.sh    ocupacio_t6.sh    user_stats.sh
```

Fer un fitxer amb la llista de fitxers que es volen excloure: llista_excloure.txt

```
root@marionaF (Sun Dec 17):<~/Documents># nano llista_excloure.txt
root@marionaF (Sun Dec 17):<~/Documents># ls
BadUsers.py      asosh.sh          llista_excloure.txt ocupacio_t6.sh    user_stats.sh
BadUsers.sh      copia_incremental.sh mkht3.py          subprocess        user_stats.sha256sum
BadUsers_t6.py   delete_user.sh    mkht3.sh          user_stats.py     usuarios.txt
```

En aquest fitxer posar el nom dels fitxers que no es volen en la còpia incremental, per exemple no guardarem `user_stats.py` i `usuarios.txt`



Dins del fitxer posar el codi, ara complet:

```
#!/bin/bash

# Directori d'origen per fer el backup
SOURCE_DIR="/root/"

# Directori de destinació per guardar el backup
DEST_DIR="/backup/backup-rsync"

# Fitxer que conté la llista d'exclusions
EXCLUDES="/root/Documents/llista_excloure.txt"

# Nom o adreça IP del servidor de backup
BSERVER="127.0.0.1"

# Nom del directori on es guardaran els backups incrementals
INCREMENTAL_BACKUP_DIR="incrementals"

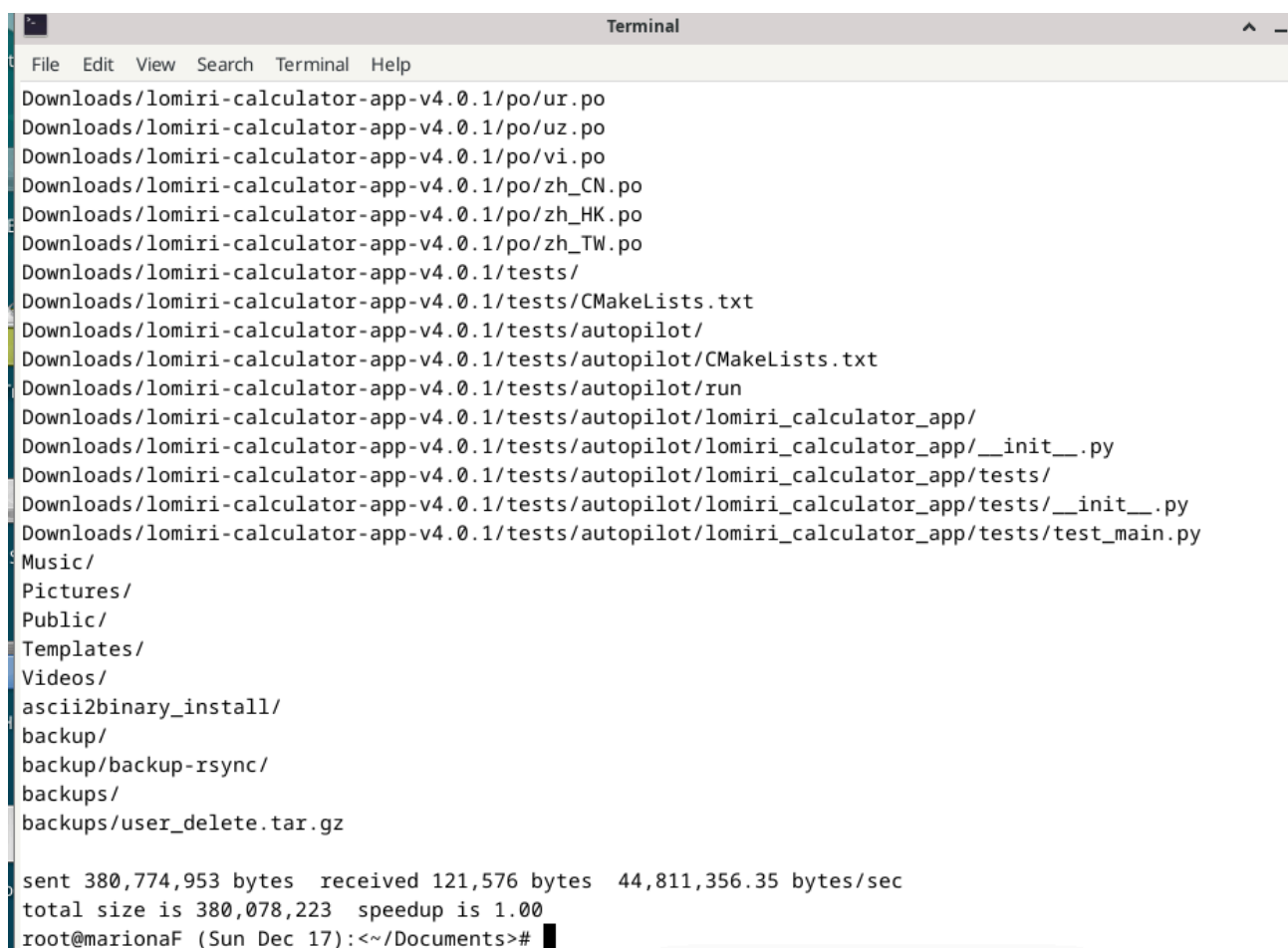
# Comanda per generar la data actual amb format d'any, mes, dia, hora, minut i segon
BACKUP_DATE=$(date +"%Y-%m-%d_%H-%M-%S")

# Opcions per a rsync
```

```
OPTS="--ignore-errors --delete-excluded --exclude-from=$EXCLUDES \  
--delete --backup  
--backup-dir=$DEST_DIR/$INCREMENTAL_BACKUP_DIR/$BACKUP_DATE -av"  
  
# Assegura que el directori de backup incremental existeix  
mkdir -p $DEST_DIR/$INCREMENTAL_BACKUP_DIR/$BACKUP_DATE  
  
# Transferència actual amb rsync  
rsync $OPTS $SOURCE_DIR root@$BSERVER:$DEST_DIR/complet
```

Executar el fitxer amb `./copia_incremental.sh`

Preguntarà per la contrasenya del root del localhost, serà la mateixa que el root



```
Terminal  
File Edit View Search Terminal Help  
Downloads/lomiri-calculator-app-v4.0.1/po/ur.po  
Downloads/lomiri-calculator-app-v4.0.1/po/uz.po  
Downloads/lomiri-calculator-app-v4.0.1/po/vi.po  
Downloads/lomiri-calculator-app-v4.0.1/po/zh_CN.po  
Downloads/lomiri-calculator-app-v4.0.1/po/zh_HK.po  
Downloads/lomiri-calculator-app-v4.0.1/po/zh_TW.po  
Downloads/lomiri-calculator-app-v4.0.1/tests/  
Downloads/lomiri-calculator-app-v4.0.1/tests/CMakeLists.txt  
Downloads/lomiri-calculator-app-v4.0.1/tests/autopilot/  
Downloads/lomiri-calculator-app-v4.0.1/tests/autopilot/CMakeLists.txt  
Downloads/lomiri-calculator-app-v4.0.1/tests/autopilot/run  
Downloads/lomiri-calculator-app-v4.0.1/tests/autopilot/lomiri_calculator_app/  
Downloads/lomiri-calculator-app-v4.0.1/tests/autopilot/lomiri_calculator_app/__init__.py  
Downloads/lomiri-calculator-app-v4.0.1/tests/autopilot/lomiri_calculator_app/tests/  
Downloads/lomiri-calculator-app-v4.0.1/tests/autopilot/lomiri_calculator_app/tests/__init__.py  
Downloads/lomiri-calculator-app-v4.0.1/tests/autopilot/lomiri_calculator_app/tests/test_main.py  
Music/  
Pictures/  
Public/  
Templates/  
Videos/  
ascii2binary_install/  
backup/  
backup/backup-rsync/  
backups/  
backups/user_delete.tar.gz  
  
sent 380,774,953 bytes  received 121,576 bytes  44,811,356.35 bytes/sec  
total size is 380,078,223  speedup is 1.00  
root@marionaF (Sun Dec 17):<~/Documents>#
```

Si entrem a la carpeta de backups ens sortirà la carpeta de les còpies incremental i la completa amb tots els documents:

Incremental guarda la data quan s'ha fet la còpia:

```

root@marionaF (Sun Dec 17):<~/Documents># cd /backup/backup-rsync/
root@marionaF (Sun Dec 17):</backup/backup-rsync># ls
arxiu_nou.txt  complet  incrementals  root
root@marionaF (Sun Dec 17):</backup/backup-rsync># cd incrementals/
root@marionaF (Sun Dec 17):</backup/backup-rsync/incrementals># ls
2023-12-17_20-04-01
root@marionaF (Sun Dec 17):</backup/backup-rsync/incrementals># █

```

Complet és on es guarda tots els documents :

```

root@marionaF (Sun Dec 17):</backup/backup-rsync># cd complet/
root@marionaF (Sun Dec 17):</backup/backup-rsync/complet># ls
-hola.txt  Music      Videos      backup      sudousers
Desktop    Pictures   arxiu_nou.txt  backups     usuarios-20231212171129.txt
Documents  Public     ascii2binary_install  private.key  usuarios-20231212172100.txt
Downloads  Templates  autoInstall.sh  sudoers

```

Podem observar que les fitxers que hem posat a la llista d'exclusió no sha fet la còpia:

```

root@marionaF (Sun Dec 17):</backup/backup-rsync/complet/Documents># ls
BadUsers.py      copia_incremental.sh  mkht3.sh      user_stats.sha256sum
BadUsers.sh      delete_user.sh        ocupacio_t6.sh
BadUsers_t6.py   llista_excloure.txt   subprocess
asosh.sh         mkht3.py              user_stats.sh

```

(No hi ha ni `user_stats.py` ni `usuarios.txt`)

Ara creeu un fitxer **arxiu.txt** i feu-lo servir per comprovar el funcionament de l'script anterior

```

root@marionaF (Sun Dec 17):<~/Documents># ls
BadUsers.py      asosh.sh          mkht3.py        user_stats.py
BadUsers.sh      copia_incremental.sh  mkht3.sh        user_stats.sh
BadUsers_t6.py   delete_user.sh     ocupacio_t6.sh   user_stats.sha256sum
arxiu.txt       llista_excloure.txt  subprocess       usuarios.txt

```

```

GNU nano 7.2                                arxiu.txt
Nou arxiu a Documents

```

Abans de fer una altra copia la carpeta de Documents no hi ha el fitxer `arxiu.txt`:

```

root@marionaF (Sun Dec 17):</backup/backup-rsync/complet># cd Documents/
root@marionaF (Sun Dec 17):</backup/backup-rsync/complet/Documents># ls
BadUsers.py      copia_incremental.sh  mkht3.sh      user_stats.sha256sum
BadUsers.sh      delete_user.sh        ocupacio_t6.sh
BadUsers_t6.py   llista_excloure.txt   subprocess
asosh.sh         mkht3.py              user_stats.sh

```

Si es fa una còpia després de la seva creació, es pot veure que es un dels documents que es copia en el backup:

```
.mozilla/firefox/57zbz911.default-esr/places.sqlite-wal
.mozilla/firefox/57zbz911.default-esr/datareporting/
.mozilla/firefox/57zbz911.default-esr/datareporting/aborted-session-ping
.mozilla/firefox/57zbz911.default-esr/datareporting/glean/db/data.safe.bin
.mozilla/firefox/57zbz911.default-esr/sessionstore-backups/
.mozilla/firefox/57zbz911.default-esr/sessionstore-backups/recovery.baklz4
.mozilla/firefox/57zbz911.default-esr/sessionstore-backups/recovery.jsonlz4
.mozilla/firefox/57zbz911.default-esr/storage/default/https+++accounts.google.com/ls/
.mozilla/firefox/57zbz911.default-esr/storage/default/https+++accounts.google.com/ls/data.s
lite
.mozilla/firefox/57zbz911.default-esr/storage/default/https+++accounts.google.com/ls/usage
.mozilla/firefox/57zbz911.default-esr/storage/permanent/chrome/idb/
Documents/
Documents/arxiu.txt
sent 3,489,407 bytes  received 45,886 bytes  1,414,117.20 bytes/sec
total size is 339,214,095  speedup is 95.95
root@marionaF (Sun Dec 17):<~/Documents>#
```

El arxiu s'ha pogut copiar correctament:

```
root@marionaF (Sun Dec 17):</backup/backup-rsync/complet/Documents># ls
BadUsers.py      asosh.sh          mkht3.py          user_stats.sh
BadUsers.sh      copia_incremental.sh mkht3.sh          user_stats.sha256sum
BadUsers_t6.py   delete_user.sh    ocupacio_t6.sh
arxiu.txt        llista_excloure.txt subprocess
root@marionaF (Sun Dec 17):</backup/backup-rsync/complet/Documents># cat arxiu.txt
Nou arxiu a Documents
```

Després modifiqueu aquest **arxiu.txt** i torneu a sincronitzar.

Modifiar el arxiu.txt des de la carpeta de Documents:

```
GNU nano 7.2 arxiu.txt
MODIFIACIÓ al arxiu a Documents
```

Fer una còpia:

```

root@marionaF (Sun Dec 17):<~/Documents># ./copia_incremental.sh
root@127.0.0.1's password:
sending incremental file list
.cache/mozilla/firefox/57zbz911.default-esr/cache2/entries/
.cache/mozilla/firefox/57zbz911.default-esr/cache2/entries/668DB06239121C0F982445DDA56B4BAF6
995D4DF
.cache/mozilla/firefox/57zbz911.default-esr/cache2/entries/8211576DCEAD9BF3A38A9D0366E78A4C1
C6A449D
.cache/mozilla/firefox/57zbz911.default-esr/cache2/entries/F96B2EB5EB440388D27175D0AE29F5AE6
D830BC5
.cache/mozilla/firefox/57zbz911.default-esr/cache2/entries/FC549FA793095A6193AFF337281F79710
7E20234
.mozilla/firefox/57zbz911.default-esr/
.mozilla/firefox/57zbz911.default-esr/AlternateServices.txt
.mozilla/firefox/57zbz911.default-esr/cookies.sqlite
.mozilla/firefox/57zbz911.default-esr/cookies.sqlite-wal
.mozilla/firefox/57zbz911.default-esr/datareporting/glean/db/data.safe.bin
.mozilla/firefox/57zbz911.default-esr/storage/permanent/chrome/idb/
Documents/
Documents/arxiu.txt

sent 554,441 bytes  received 9,557 bytes  161,142.29 bytes/sec
total size is 339,422,859  speedup is 601.82

```

L'arxiu dins del backup sha modificat correctament:

```

root@marionaF (Sun Dec 17):</backup/backup-rsync/complet># cd Documents/
root@marionaF (Sun Dec 17):</backup/backup-rsync/complet/Documents># ls
BadUsers.py      asosh.sh          mkht3.py          user_stats.sh
BadUsers.sh      copia_incremental.sh mkht3.sh          user_stats.sha256sum
BadUsers_t6.py   delete_user.sh    ocupacio_t6.sh
arxiu.txt        llista_excloure.txt subprocess
root@marionaF (Sun Dec 17):</backup/backup-rsync/complet/Documents># cat arxiu.txt
MODIFICACIÓ al arxiu a Documents

```

Finalment esborreu el fitxer **arxiu.txt** i feu una sincronització més.

Borrar el fitxer:

```

root@marionaF (Sun Dec 17):<~/Documents># ls
BadUsers.py      asosh.sh          mkht3.py          user_stats.py
BadUsers.sh      copia_incremental.sh mkht3.sh          user_stats.sh
BadUsers_t6.py   delete_user.sh    ocupacio_t6.sh    user_stats.sha256sum
arxiu.txt        llista_excloure.txt subprocess         usuaris.txt
root@marionaF (Sun Dec 17):<~/Documents># rm arxiu.txt
root@marionaF (Sun Dec 17):<~/Documents># ls
BadUsers.py      copia_incremental.sh mkht3.sh          user_stats.sh
BadUsers.sh      delete_user.sh       ocupacio_t6.sh    user_stats.sha256sum
BadUsers_t6.py   llista_excloure.txt  subprocess         usuaris.txt
asosh.sh         mkht3.py             user_stats.py

```

Executar la copia incremental:

```
.cache/mozilla/firefox/57zbx911.default-esr/cache2/entries/6D89348819C8881868053197CA0754F36
0DD51B0
.cache/mozilla/firefox/57zbx911.default-esr/cache2/entries/9C0164270849AFB372E44DA44A9FCA49
784BF5F
.cache/mozilla/firefox/57zbx911.default-esr/cache2/entries/F9825E7F2B03B1E8575E08767D2F602AD
AA630D3
E5F87B7
deleting Documents/arxiu.txt
.mozilla/firefox/57zbx911.default-esr/
.mozilla/firefox/57zbx911.default-esr/AlternateServices.txt
.mozilla/firefox/57zbx911.default-esr/cookies.sqlite
.mozilla/firefox/57zbx911.default-esr/cookies.sqlite-wal
.mozilla/firefox/57zbx911.default-esr/permissions.sqlite
.mozilla/firefox/57zbx911.default-esr/datareporting/
.mozilla/firefox/57zbx911.default-esr/datareporting/aborted-session-ping
.mozilla/firefox/57zbx911.default-esr/datareporting/glean/db/data.safe.bin
.mozilla/firefox/57zbx911.default-esr/sessionstore-backups/
.mozilla/firefox/57zbx911.default-esr/sessionstore-backups/recovery.baklz4
.mozilla/firefox/57zbx911.default-esr/sessionstore-backups/recovery.jsonlz4
.mozilla/firefox/57zbx911.default-esr/storage/default/https+++accounts.google.com/ls/
.mozilla/firefox/57zbx911.default-esr/storage/default/https+++accounts.google.com/ls/data.sq
lite
.mozilla/firefox/57zbx911.default-esr/storage/default/https+++accounts.google.com/ls/usage
.mozilla/firefox/57zbx911.default-esr/storage/permanent/chrome/idb/
Documents/

sent 1,384,840 bytes  received 12,884 bytes  559,089.60 bytes/sec
total size is 339,421,154  speedup is 242.84
```

Com a la sortida de l'execució surt, s'haurà borrat el fitxer en el backup complet:

```
root@marionaF (Sun Dec 17):</backup/backup-rsync/complet># cd Documents/
root@marionaF (Sun Dec 17):</backup/backup-rsync/complet/Documents># ls
BadUsers.py      copia_incremental.sh  mkht3.sh          user_stats.sha256sum
BadUsers.sh      delete_user.sh        ocupacio_t6.sh
BadUsers_t6.py   llista_excloure.txt  subprocess
asosh.sh         mkht3.py              user_stats.sh
```

Què observeu en modificar un fitxer i fer una sincronització? I en esborrar-lo?

Quan es modifica un fitxer, ja sigui el nom o el contingut, quan es faci la còpia completa es guardarà la modificació feta, però si no es fa una copia, el fitxer del backup continuarà tenint la versió antiga, la ultima feta en la còpia.

Si s'esborra algun fitxer, aquest també s'esborrarà del backup quan es faci una còpia de seguretat, mentre que no es faci aquell fitxer continuarà existint.

4.3. Realització de còpies incrementals inverses tipus snapshot

Una possibilitat que dóna `rsync` és fer *backups* incrementals on, utilitzant una propietat dels enllaços durs, és possible fer que les còpies incrementals semblin còpies completes. Per això analitzarem primer algunes propietats dels enllaços durs.

Repàs d'enllaços durs

El nom d'un fitxer no representa el fitxer mateix, per al sistema és només un enllaç dur al *inode*. Això permet que un fitxer (inode) pugui tenir més d'un enllaç dur. Per exemple si teniu un fitxer `file_a` es pot crear un enllaç cridat `file_b`:

```
$ ln file_a file_b
```

Amb la comanda `stat` es pot saber quants enllaços durs té un fitxer:

```
$ stat filename
```

Com es pot comprovar que `file_a` i `file_b` pertanyen al mateix *inode*?

```
root@PauA (Sun Dec 17):<~/Desktop># stat net-out.py
  File: net-out.py
  Size: 1338          Blocks: 8          IO Block: 4096   regular file
Device: 8,1    Inode: 159777      Links: 1
Access: (0755/-rwxr-xr-x)  Uid: (  0/   root)   Gid: (  0/   root)
Access: 2023-11-08 22:13:00.901379356 +0000
Modify: 2023-11-08 10:52:56.000000000 +0000
Change: 2023-11-08 22:12:51.553553171 +0000
 Birth: 2023-11-08 11:22:23.275169298 +0000
root@PauA (Sun Dec 17):<~/Desktop># stat net-out.sh
  File: net-out.sh
  Size: 827          Blocks: 8          IO Block: 4096   regular file
Device: 8,1    Inode: 135852      Links: 1
Access: (0644/-rw-r--r--)  Uid: (  0/   root)   Gid: (  0/   root)
Access: 2023-11-08 22:26:08.384931072 +0000
Modify: 2023-11-08 22:24:48.000000000 +0000
Change: 2023-11-08 22:25:54.869096478 +0000
 Birth: 2023-11-08 22:25:37.125315102 +0000
root@PauA (Sun Dec 17):<~/Desktop>#
```

Executar `stat file_a` i anotar el número d'inode que apareix.

Executar `stat file_b` i comparar el seu número d'inode amb el de `file_a`.

Si els números d'inode són idèntics, llavors `file_a` i `file_b` són enllaços durs del mateix fitxer.

Què passa amb el fitxer `file_b` si es fa un canvi a `file_a`?

Quan es fa un canvi en `file_a`, que és un dels noms associats a un inode a través d'un enllaç dur (`file_b` és l'altre), el canvi s'aplicarà al contingut de l'arxiu que es troba emmagatzemat en l'espai de disc que és referenciat per l'inode. Ja que els enllaços durs `file_a` i `file_b` apunten al mateix inode, qualsevol canvi en el contingut del fitxer a través d'un d'aquests noms serà visible quan s'accedeix al fitxer a través de l'altre nom.

I si es fa un canvi de permisos a `file_a`?

Si es canvia els permisos de `file_a` utilitzant una comanda com `chmod`, aquests canvis afectaran també a `file_b`, ja que els permisos del fitxer es guarden a l'inode. Com que tant `file_a` com `file_b` són enllaços durs al mateix inode, qualsevol modificació dels permisos s'aplicarà a l'inode i, per tant, serà reflectida a tots els noms que hi apunten.

I si copiem un altre fitxer sobreescrivint el fitxer `file_a` (`cp file_c file_a`)? I si es sobreescriu amb l'opció `--remove-destination`?

Quan copies un fitxer sobre un altre utilitzant la comanda `cp`, el comportament dependrà de la manera específica en què s'executa la comanda:

Si executas `cp file_c file_a` sense cap opció adicional:

- El contingut de `file_c` substituirà el contingut de `file_a`.
- L'inode al qual apuntava `file_a` mantindrà els mateixos enllaços durs, incloent `file_b`, però ara amb el nou contingut copiat de `file_c`.
- Els permisos i propietat del fitxer original `file_a` es mantindran, ja que l'operació de copia no modifica l'inode; només es canvia el contingut del fitxer.

Si utilitzas `cp --remove-destination file_c file_a`:

- L'opció `--remove-destination` esborra el fitxer destí (`file_a`) abans de realitzar la copia.
- Això significa que `file_a` serà eliminat (juntament amb la seva entrada a l'inode), i després `file_c` serà copiat en un nou fitxer anomenat `file_a`.
- En aquest cas, `file_a` tindrà un nou inode diferent de `file_b`, ja que has eliminat l'enllaç original i creat un de nou amb el contingut de `file_c`.
- El fitxer `file_b` continuarà apuntant a l'antic inode (i per tant al contingut original) fins que aquest inode no tingui més referències (és a dir, quan tots els enllaços durs i referències de processos s'eliminin).

Per tant, en el primer cas, `file_a` i `file_b` seguiran sent enllaços durs al mateix inode `i`, per tant, continuaran tenint el mateix contingut després de la comanda `cp`. En el segon cas, `file_a` serà ara un enllaç a un nou inode amb un contingut diferent, i `file_b` no es veurà afectat i seguirà apuntant al contingut original.

I què passa amb `file_b` si `file_a` és esborrat?

Si `file_a` és esborrat, això no afecta l'existència de `file_b` ni el contingut del fitxer a què tots dos noms apuntaven originalment.

La comanda `cp` té una opció per fer còpies en què realment no es fa una còpia sinó un enllaç dur (`cp -l`). Una altra opció interessant es `-a` que fa una còpia recursiva i preserva els permisos de accés, els temps i els propietaris dels fitxers

4.3.1 Backups tipus “snapshot” amb `rsync` i `cp -al`

Es poden combinar `rsync` i `cp -al` per crear *backups* que semblin múltiples còpies completes d'un sistema de fitxers sense que sigui necessari gastar tot l'espai en disc requerit per totes les còpies, en resum es podria fer:

```
rm -rf backup.3  
mv backup.2 backup.3  
mv backup.1 backup.2  
cp -al backup.0 backup.1  
rsync -a --delete source_directory/ backup.0/
```

Si les comandes anteriors s'executen cada dia, `backup0`, `backup1`, `backup2` i `backup3` apareixeran com si fossin còpies completes del directori `source_directory` com estava avui, ahir, abans d'ahir i tres dies abans respectivament. Però en realitat l'espai extra serà igual a la grandària del directori `source_directory` més la grandària total dels canvis dels últims tres dies. Exactament el mateix que un *backup* complet més els *backups* incrementals con heu fet abans amb `tar` i el mateix `rsync`. L'únic problema és que els permisos i propietaris de les còpies anteriors serien els mateixos que els de la còpia actual.

Hi ha una opció d'`rsync` que fa directament la còpia amb enllaços durs (`--link-dest`) i d'aquesta manera no seria necessari utilitzar la comanda `cp`, a més a més que preserva els permisos i propietaris de les còpies anteriors. Amb aquesta opció l'esquema platejat anteriorment quedaria així:

```
rm -rf backup.3

mv backup.2 backup.3

mv backup.1 backup.2

mv backup.0 backup.1

rsync -a --delete --link-dest=../backup.1 \
    source_directory/ backup.0/
```

4.4. Script per fer backups tipus snapshot

Per fer *backups* del directori `/root` utilitzarem l'script **backup-rsync-snapshot.sh**, que està disponible a l'apèndix.

En primer lloc modifiqueu les variables de la secció “file locations” per posar els valors adequats del vostre sistema.

En la secció de “file locations” ficarem els següents valors:

`MOUNT_DEVICE = "/dev/sda4"`

`SNAPSHOT_MOUNTPOINT = "/backup"`

`SNAPSHOT_DIR = "root-backup"`

`EXCLUDES = "/root/exclude_list.txt"`

`SOURCE_DIR = "/root"`

Després completeu la secció amb la comanda **rsync** amb els valors apropiats per fer la còpia tipus *snapshot*

```
$RSYNC -a --delete --exclude-from$EXCLUDES
--link-dest=$SNAPSHOT_MOUNTPOINT/$SNAPSHOT_DIR/daily.1/$SOURCE_DIR/
$SNAPSHOT_MOUNTPOINT/$SNAPSHOT_DIR/daily.0/
```

Ara feu modificacions als fitxers del directori origen (per exemple crear un nou fitxer, modificar el contingut i la data d'accés a un fitxer o esborrar un fitxer) i torneu a executar l'script. Feu això varies vegades fins que tingueu una còpia actual i tres còpies anteriors.

Què observeu al modificar un fitxer i fer una sincronització?

Creació de nous fitxers: Quan afegeixes nous fitxers al directori origen i executes l'script, aquests nous fitxers es veuran reflectits en la còpia de seguretat més recent (daily.0). Cada vegada que fas una nova còpia de seguretat, es crea una nova instantània del sistema en aquest directori.

Modificació de fitxers existents: Quan modifiques el contingut d'un fitxer existent i executes l'script, aquest canvi es reflectirà només en la còpia de seguretat més recent (daily.0). Les còpies anteriors (daily.1, daily.2, etc.) continuen mantenint les versions anteriors d'aquests fitxers, ja que es fan servir enllaços durs per compartir els fitxers inalterats entre els snapshots. El mateix passa quan eliminem un fitxer al directori origen del snapshot, que només es reflectirà en la còpia de seguretat més recent.

Quina és la grandària del directori /backup.0 i dels altres directoris backup1, backup2 i backup3?

Ho podem saber executant la comanda:

```
du -h /backup*    //du=disk usage, -h= human-readable, /backup= directori pare.
```

, la qual ens mostra la grandària de cada directori backup. Podem esperar que la grandària del directori backup.0 serà la major, ja que les còpies més antigues probablement compartiran molts fitxers amb les còpies més recents a través d'enllaços durs, de manera que ocuparan menys espai de disc.

Finalment, voldríem fer una restauració. Canvieu el nom del directori /root per simular una pèrdua de dades. Feu una restauració d'aquest directori amb la còpia de seguretat més recent.

En primer lloc, canviarem el nom del nostre directori root amb la comanda:

```
mv /root /root_old
```

A continuació, restaurem la còpia de seguretat més recent amb la comanda:

```
cp -al /backup.0/daily.0 /root
```

Aquesta comanda farà una còpia enllaçada al directori /root. L'ús de -al assegura que es facin enllaços durs als fitxers per optimitzar l'ús de l'espai de disc.

5. Referències

- (1) Lars Wirzenius, Joanna Oja, Stephen Stafford, Alex Weeks, **The Linux System Administrator's Guide** Version 0.9, <http://tldp.org/LDP/sag>
- (2) AEleen Frisch, **Essential System Administration**. O'Reilly. 2002.
- (3) Mike Rubel. **Easy Automated Snapshot-Style Backups with Linux and Rsync**.
http://www.mikerubel.org/computers/rsync_snapshots/
- (4) **Rsnapshot**. *a remote filesystem snapshot utility based on rsync for making backups of local and remote systems.* <http://www.rsnapshot.org/>

6. Apèndix. Codi de l'script per còpies tipus snapshot

```
#!/bin/bash

# -----

# mikes handy rotating-filesystem-snapshot utility
# http://www.mikerubel.org/computers/rsync_snapshots
# Modified by Mauricio Alvarez: http://people.ac.upc.edu/alvarez
# -----

# ----- system commands used by this script-----

ID=/usr/bin/id

ECHO=/bin/echo

MOUNT=/bin/mount

RM=/bin/rm

MV=/bin/mv

CP=/bin/cp

TOUCH=/usr/bin/touch

RSYNC=/usr/bin/rsync
```

```

# ----- file locations -----

MOUNT_DEVICE=

SNAPSHOT_MOUNTPOINT=

SNAPSHOT_DIR=

EXCLUDES=

SOURCE_DIR=


# ----- the script itself-----


# make sure we're running as root

if (( ` $ID -u ` != 0 )); then { $ECHO "Sorry, must be root.  Exiting...";
exit; } fi


# attempt to remount the RW mount point as RW; else abort

$MOUNT -o remount,rw $MOUNT_DEVICE $SNAPSHOT_MOUNTPOINT ;

if (( $? )); then

{

    $ECHO "snapshot: could not remount $SNAPSHOT_MOUNTPOINT readwrite";

    exit;

}

fi;


# rotating snapshots of /$SNAPSHOT_DIR


# step 1: delete the oldest snapshot, if it exists:

if [ -d $SNAPSHOT_MOUNTPOINT/$SNAPSHOT_DIR/daily.3 ] ; then

    $RM -rf $SNAPSHOT_MOUNTPOINT/$SNAPSHOT_DIR/daily.3

```

```

fi

# step 2: shift the middle snapshotss back by one, if they exist
if [ -d $SNAPSHOT_MOUNTPOINT/$SNAPSHOT_DIR/daily.2 ] ; then

    $MV      $SNAPSHOT_MOUNTPOINT/$SNAPSHOT_DIR/daily.2
$SNAPSHOT_MOUNTPOINT/$SNAPSHOT_DIR/daily.3

fi

if [ -d $SNAPSHOT_MOUNTPOINT/$SNAPSHOT_DIR/daily.1 ] ; then

    $MV      $SNAPSHOT_MOUNTPOINT/$SNAPSHOT_DIR/daily.1
$SNAPSHOT_MOUNTPOINT/$SNAPSHOT_DIR/daily.2

fi

if [ -d $SNAPSHOT_MOUNTPOINT/$SNAPSHOT_DIR/daily.0 ] ; then

    $MV      $SNAPSHOT_MOUNTPOINT/$SNAPSHOT_DIR/daily.0
$SNAPSHOT_MOUNTPOINT/$SNAPSHOT_DIR/daily.1

fi;

# step 3: rsync from the system into the latest snapshot
$RSYNC

# complete here what is missing for the rsync command:

# - basic options:

# - excludes:

# - --link-dest=

# - source and destination directories

# step 5: update the mtime of daily.0 to reflect the snapshot time
$TOUCH $SNAPSHOT_MOUNTPOINT/$SNAPSHOT_DIR/daily.0 ;

```

```
# now remount the RW snapshot mountpoint as readonly
$MOUNT -o remount,ro $MOUNT_DEVICE $SNAPSHOT_MOUNTPOINT ;

if (( $? )); then
{
    $ECHO "snapshot: could not remount $SNAPSHOT_MOUNTPOINT readonly";
    exit;
} fi;
```