

**Índex**

[1. Introducció 2](#_heading=)

[1.1. Objectiu 3](#_heading=)

[2. Abans de començar 3](#_heading=)

[3. Partició per emmagatzemar les còpies de seguretat. 3](#_heading=h.3znysh7)

[Quina modificació és necessària fer perquè aquesta nova partició es munti automàticament durant el boot amb mode de només lectura? 6](#_heading=)

[3. Realització de còpies amb TAR 6](#_heading=h.3dy6vkm)

[3.1. Realització de còpies completes 6](#_heading=)

[3.2. Realització de còpies incrementals 11](#_heading=)

[3.3. Restauració d'una còpia de seguretat 18](#_heading=h.4d34og8)

[3.4. Restauració d'un fragment 20](#_heading=h.2s8eyo1)

[4. Realització de còpies usant RSYNC 21](#_heading=h.17dp8vu)

[4.1. Realització de backups a través d'una xarxa 21](#_heading=)

[$ echo “nou arxiu” > /root/arxiu\_nou.txt 25](#_heading=)

[4.2. Realització de còpies incrementals inverses 27](#_heading=h.lnxbz9)

[4.3. Realització de còpies incrementals inverses tipus snapshot 31](#_heading=)

[Repàs d'enllaços durs 31](#_heading=)

[4.3.1 Backups tipus “snapshot” amb rsync i cp -al 32](#_heading=)

[4.4. Script per fer backups tipus snapshot 33](#_heading=h.2jxsxqh)

[5. Referències 41](#_heading=)

[6. Apèndix. Codi de l'script per còpies tipus snapshot 41](#_heading=)

# **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 depén de la variabilitat de les dades. Una base de dades pot necessitar múltiples copies de seguretat diàries, mentre que un servidor web pot requerir només una copia diària i altres sistemes de fitxers poden requerir només una copia 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 copies 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 copia 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 diferent estratègies de còpies de seguretat.

## ***Objectiu***

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

# **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 utilitzem tar amb l'opció -cvf per exemple: *tar -cvf nom\_empaquetat.tar fitxer1 fitxer2 directori/* on:

c: Crear un fitxer empaquetat., v: Mostra el procés en pantalla, f: Especifica el nom del fitxer de sortida

Per desempaquetar utilitzem -xvf (extreure, veure i especificar fitxer): tar -xvf nom\_empaquetat.tar

1. **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 (hard link) és una referència directa al mateix fitxer. Això significa que els dos noms apunten al mateix contingut en el disc. I en cas de modificar un dels fitxers, l'altre també reflectirà els canvis, ja que són un.

Còpia (cp): Es crea un nou fitxer independent i, ocupa espai addicional al disc, ja que és una còpia completa del contingut.

Enllaç dur (ln): No es duplica el contingut; només es crea un nou nom de referència que apunta al mateix. S’han de eliminar tots els enllaços per que es perdi el contingut.

# **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 he d’activar root en cas que no el tinguem activat amb la comanda

**sudo usermod -s /bin/bash root**



Hem utilitzat les mateixes comandes que al T1. Primer hem entrar a la configuració del dis amb

**fdisk /dev/sda**



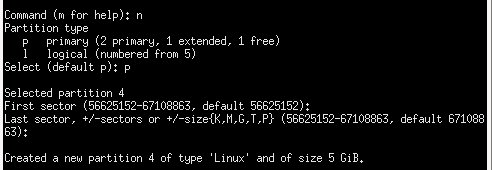
Després executem

**n** //Per crear la nova partició

Ens demana que escollim entre p (primary) o e (extended).

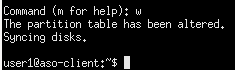
**p**

Ara ens demana que fiquem l’espai que volem donar-li a la partició, com volem l’espai lliure que queda del disc, li donem enter.



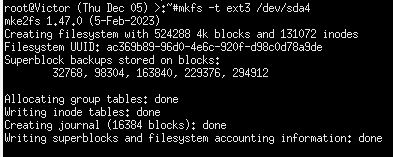
Per surtir escrivim

**w** //Per guardar els canvis



Una vegada hem sortit, li donem el format de ext3 a la partició

**mkfs -t ext3 /dev/sdax**



Ara hem de muntar la partició a /backup

Primer creem el directori

**mkdir /backup**

I muntem la partició al directori

**mount /dev/sdaX /backup**



I ara hem de donar permisos

Com hem desactivat root, li donem permís a l’usuari root.

**chown root:root /backup**

**chmod 700 /backup**



**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**

****

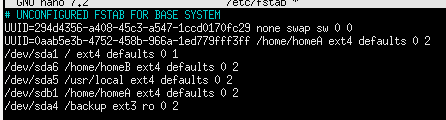
**Muntar per lectura i escriptura:**

**$ 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?

Hem de modificar el fitxer /etc/fstab i afegir /dev/sda4 /backup ext3 ro 0 2



# **Realització de còpies amb TAR**

## ***Realització de còpies completes***

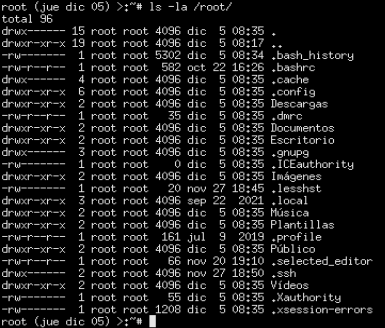
**Realitzeu una còpia completa del directori /root (comproveu que existeixen fitxers en aquest directori) amb la comanda tar. Useu noms significatius per als fitxers de *backup*: feu que el nom del fitxer tingui informació sobre el contingut del fitxer, la data i hora en que es va fer la còpia, i si la còpia és completa o incremental, i el nivell de la còpia (0, 1, ...).**

Primer, accedim amb usuari root, escrivint la comanda:

***su - root***

Després, comprovem que hi ha arxius en la carpeta /root fent servir la comanda:

***ls -la /root***

******

Una vegada sabent que tenim arxius, executem la comanda:

***tar -cvpzf /backup/backup-root-nivell0-complete-202412021900.tar.gz /root***



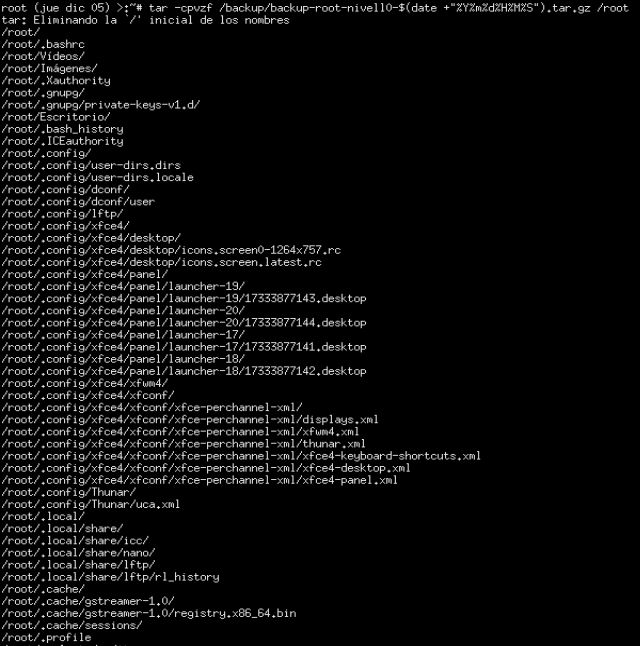
On:

* -c: Crea un nou fitxer tar.
* -v: Mostra els fitxers que s'estan processant (opcional).
* -p: Preserva els permisos dels fitxers.
* -z: Comprimeix amb gzip.
* -f: Especifica el nom del fitxer de sortida.

**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.**

Per aconseguir això, hauríem de fer servir la comanda:

**tar -cpvzf /backup/backup-root-nivell0-$(date +"%Y%m%d%H%M%S").tar.gz /root**



Si mirem la carpeta /backup:



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

Per aconseguir això, hauríem de fer servir la comanda:

***tar -cpvzf /backup/backup-etc-nivell0-$(date +"%Y%m%d%H%M%S").tar.gz /etc***



Si mirem la carpeta /backup:



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

Pot ser no aconsellable per varies raons:

* Comprimir un fitxer de backup pot ser costós en termes de temps, especialment si el backup conté molts fitxers grans o si el sistema té recursos limitats.
* En un fitxer de backup comprimit, cal descomprimir tot el backup per accedir a un sol fitxer o directori. Això és poc pràctic si es necessita fer una restauració parcial.
* Si un fitxer comprimit es corromp (per exemple, per errors de xarxa, problemes de disc, etc.), és més probable que tot el backup sigui irrecuperable. Amb fitxers no comprimits, la corrupció d'una part del backup pot afectar només fitxers individuals, deixant la resta accessibles.

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

Aleshores hauríem de fer servir la comanda:

***tar -cpvzf backup.tar.gz /backup/nomArxiu***

Com hem mencionat anteriorment, l'opció ***z*** fa que es comprimeixi fent servir gzip, ja que és una manera eficaç de reduir la mida dels fitxers i millorar la transferència de dades, amb l'avantatge de mantenir la qualitat original i ser fàcil d'usar.

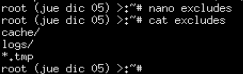
**A vegades quan es realitza la còpia completa es requereix excloure certs fitxers. Per això es pot construir un fitxer amb una llista de fitxers que no haurien de ser al *backup.***

**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*?**

Per fer això, creem l’arxiu fent servir la comanda:

***nano excludes***

I afegim els fitxers que creiem innecessaris.



I executem la següent comanda:

***tar -cpvzf /backup/backup-etc-nivell0-$(date +"%Y%m%d%H%M%S").tar.gz --exclude-from=excludes /etc***

******

**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.**

Fem servir la comanda:

***sha512sum /backup/nomBackup.tar.gz > /backup/nomBackup.tar.gz.asc***



En cas que es vulgui verificar la integritat del fitxer:

***sha512sum -c /backup/nomBackup.tar.gz.asc***



## ***Realització de còpies incrementals***

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



Fem les següents comandes per fer totes les modificacions anteriors:

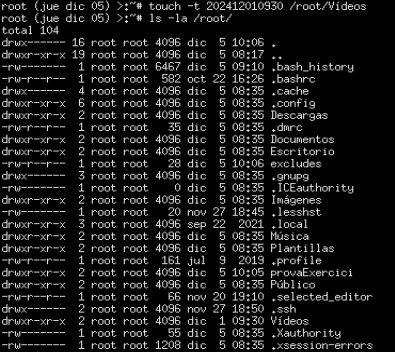
***mkdir /root/provaExercici***

***nano /root/provaExercici/exemple.c*** (arxiu buit)

***nano /root/excludes*** (afegim aquesta línia al final de l’arxiu:

#Exemple)

***touch -t 202412010930 /root/Vídeos***

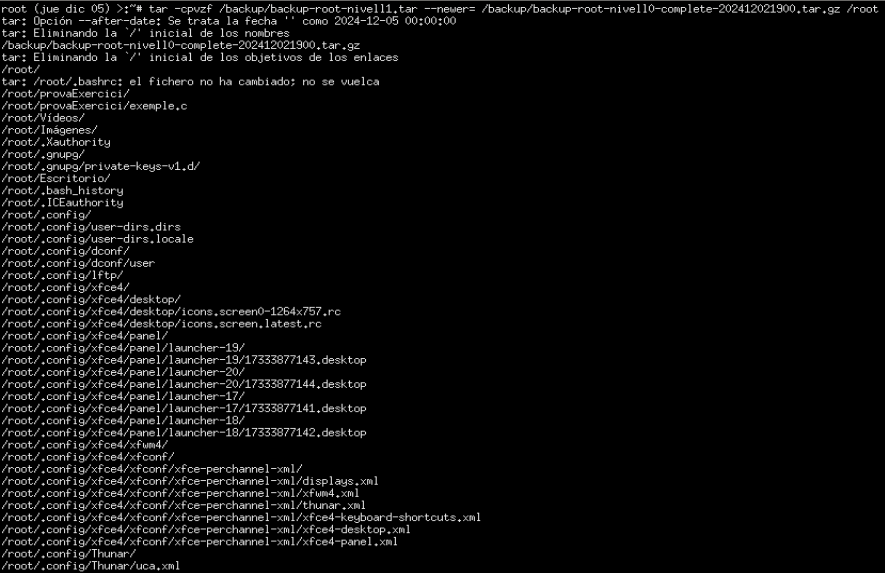


**Per fer còpies incrementals amb tar tenim l'opció --newer que només inclou els fitxers que hagin estat modificats des d'una data determinada. Aquesta data es pot especificar de dues formes: la primera, posant-la directament, per exemple: “--newer=”2021-11-28 12:10”. La segona manera consisteix en agafar 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.**

Fem servir la comanda:

***tar -cpvzf /backup/backup-root-nivell1.tar --newer=/backup/backup-root-nivell0-complete-202412021900.tar.gz /root***

******

Fem la comanda:

***sha512sum /backup/backup-root-nivell1.tar > /backup/backup-root-nivell1.tar.asc***



**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?**

Els potencials problemes són:

* Desviació en la data d'última modificació: La data associada al fitxer .tar no reflecteix necessàriament el moment exacte en què es van copiar els fitxers originals, sinó quan el fitxer de backup es va crear. Si aquest fitxer .tar es mou o es modifica posteriorment (per exemple, canviant-ne la ubicació o comprimint-lo de nou), la data d’última modificació canviarà, i això pot afectar la selecció de fitxers per a la còpia incremental.
* Confusió en la integritat: Si s’utilitza un .tar modificat o corromput, la còpia incremental pot no ser precisa, ja que la data de modificació pot no ser fiable.

Es pot resoldre d’aquestes maneres:

* Crear un fitxer amb la data de l'última còpia completa, utilitzar-lo com a referència per a còpies incrementals i actualitzar el fitxer de referència després d'una còpia completa.
* Una altra solució més robusta és utilitzar un sistema de còpies de seguretat que ja estigui dissenyat per gestionar còpies incrementals i diferencials de manera eficient, com ara rsync o eines especialitzades de còpies de seguretat.

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

Fem les següents comandes per fer totes les modificacions anteriors:

***rm /root/provaExercici/exemple.c***

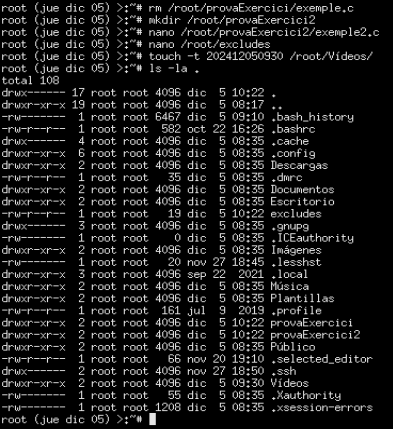
***mkdir /root/provaExercici2***

***nano /root/provaExercici/exemple2.c*** (arxiu buit)

***nano /root/excludes*** (esborrem aquesta línia al final de l’arxiu:

#Exemple)

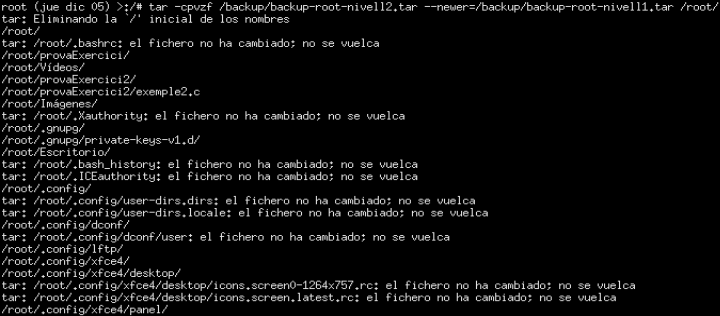
***touch -t 202412050930 /root/Vídeos***

****

**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.**

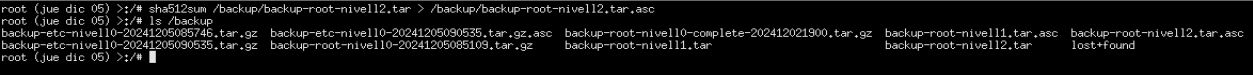
Fem la comanda:

***tar -cpvzf /backup/backup-root-nivell2.tar --newer=/backup/backup-root-nivell1.tar /root***

******

Fem la comanda:

***sha512sum /backup/backup-root-nivell2.tar > /backup/backup-root-nivell2.tar.asc***

**

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

Per aconseguir això, es descomprimeix el tar de la còpia incremental en un directori nou (/backup-extra) i es compara amb el directori /root.

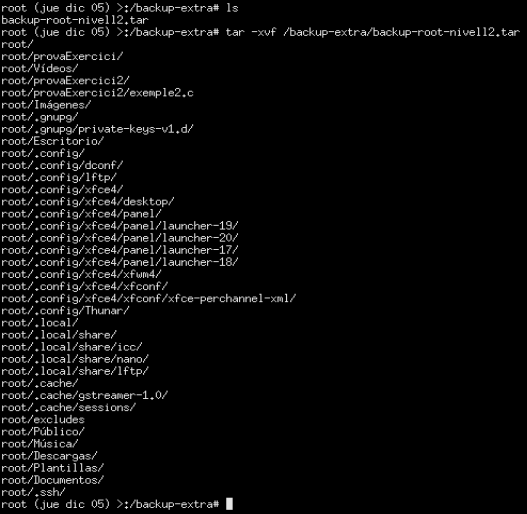
Fem les comandes:

***mkdir /backup-extra***

***cp /backup/backup-root-nivell2.tar /backup-extra***

***cd /backup-extra***

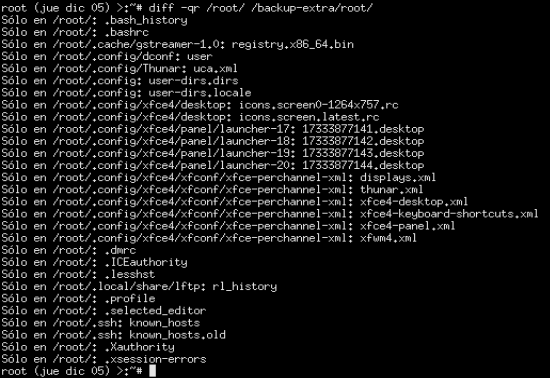
***tar -xvf /backup-extra/backup-root-nivell2.tar***



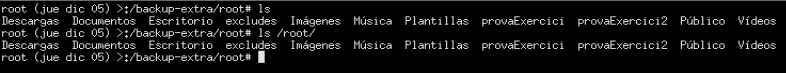


Per comparar fem:

***diff -qr /root/ /backup-extra/root/***

******

Com podem observar, hi ha diferències, però són diferències d’arxius que sempre s’estan modificant (.bash\_history, .bashrc…), però no hi ha diferències dels arxius que hem modificat.



**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 d’un fitxer de còpia de seguretat amb sha512sum, necessitem disposar del fitxer de hash (el .asc) que es va generar quan es va crear la còpia de seguretat. Aquest fitxer conté el hash calculat del fitxer en el moment de la còpia, i serveix com a referència per comprovar si el fitxer ha estat modificat posteriorment.

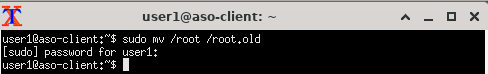
Per comprovar-ho:

***sha512sum -c /backup/nomBackup.tar.gz.asc***

******

## **Restauració d'una còpia de seguretat**

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





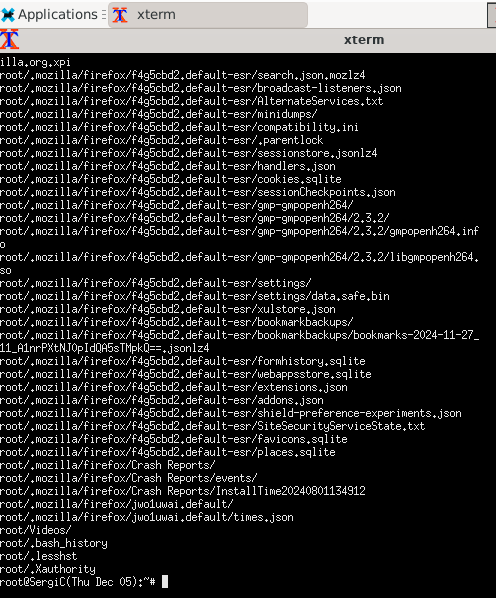
**En quin ordre cal restaurar els fitxers per tal que el resultat final sigui el desitjat?**

Primer hem de restaurar la còpia completa, i després els incrementals en l’ordre en què van ser creats, per tant, restaurarem el complet, l'incremental nivell 1 i per últim l'incremental nivell 2

**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.**

tar -xvf /backup/backup-root-nivell0-complete-202412021900.tar.gz -C /





Incremental1 

Incremental2



**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 hem eliminat abans de fer la segona còpia incremental no seran restaurats, ja que no existeixen a la còpia incremental més recent. Per detectar aquests fitxers eliminats, compararem els directoris abans i després de la restauració amb diff -qr /root.old /root.



i veiem com entre d’altres fitxers un es el fitxer exemple.c que vam eliminar abans de fer la còpia incremental 2 i no s’ha restaurat, nomes esta en el arxiu root i no en el root.old.

Els fitxers eliminats seran eliminats de les còpies de seguretat en cas d’executar una nova còpia completa que ja no inclogui aquests fitxers. A les còpies incrementals, els fitxers eliminats es mantindran fins que es realitzi una nova còpia completa sense ells.

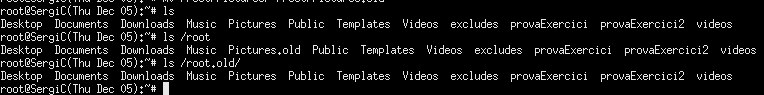
## **Restauració d'un fragment**

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

Per renomanar un subdirectori fem mv, en el nostre cas, renaomanem el subdirectori de pictures dins de root que pasara a ser anomenat pictures.old



ho comprovem fent ls, ls /root i ls /root.old on com podem veure el nou subdirectori només el veiem en fer ls /root



**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.**

Seguint l’ordre que hem parlat abans utilitzem les comandes:

tar -xvpzf /backup/backup-root-nivell0-2024….-.tar.gz -C / root/Pictures



tar -xvpzf /backup/backup-root-nivell1.tar -C / root/Pictures



tar -xvpzf /backup/backup-root-nivell2..tar -C / /root/Pictures



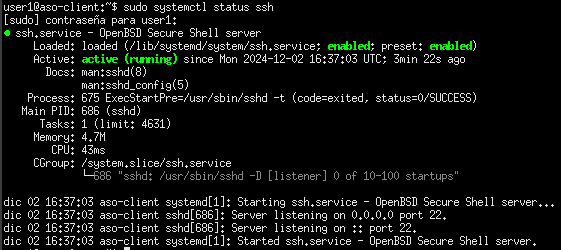
# **Realització de còpies usant RSYNC**

**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.**

## ***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.**

Comprovem l’estat de ssh amb:  
 sudo systemctl status 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.**

Abans de començar, activem temporalment l’accés a root per ssh, primer assignem una contrasenya a root amb:

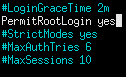
sudo passwd root



Després activem root i l’accés a root per ssh modificant l’arxiu sshd\_config. Per activar root utilitzem sudo usermod -s /bin/bash root. Per activar l’accés per ssh obrim l’arxiu amb nano i canviem la linia PermitRootLogin prohibit-password per PermitRootLogin yes:







I per últim fem un restart del servei ssh amb:

sudo systemctl restart ssh



Un cop activat l’accés a root per ssh, creem el directori /backup/backup-rsync i fem que només root tingui accés:

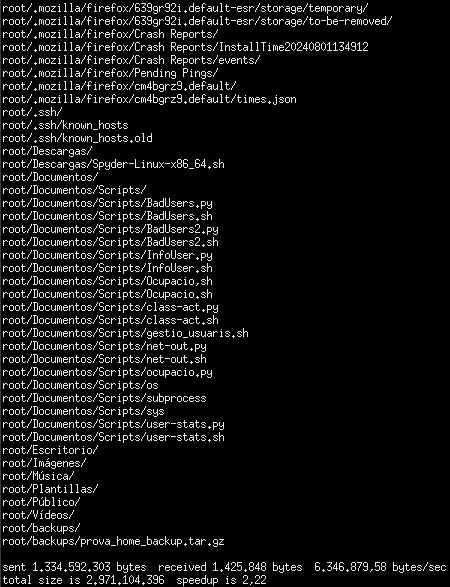


Quan ja tenim creat el directori instal·lem rsync i executem la comanda:

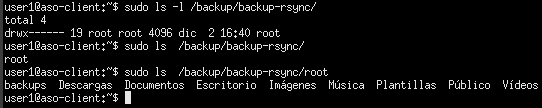
sudo apt install rsync

sudo rsync -avz /root -e ssh root@localhost:/backup/backup-rsync/





I verifiquem que els fitxers s’han copiat correctament amb ls:



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

-a: Mode arxiu. Preserva atributs dels fitxers com permisos, propietaris, grups, dates, enllaços simbòlics i subdirectoris.

-v: Mode detallat. Mostra informació sobre els fitxers copiats i el progrés.

-z: Compressió. Redueix la mida de les dades durant la transferència per accelerar el procés en xarxes lentes.

**Creeu un arxiu en el directori root amb:**

### $ echo “nou arxiu” > /root/arxiu\_nou.txt

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

Per crear l’arxiu “arxiu\_nou.txt” fem:



I comprovem que s’ha creat correctament amb:

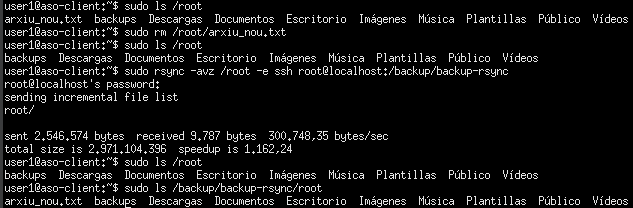


I tornem a fer rsync:

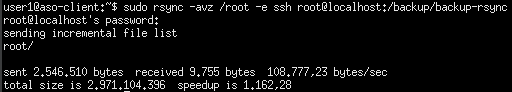


**Ara esborreu l'arxiu que heu creat abans i torneu a sincronitzar.**

Esborrem l’arxiu i comprovem que s’ha fet correctament:

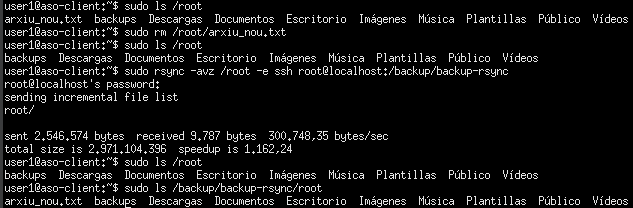


Un cop esborrat, sincronitzem de nou:



**Què ha passat amb el fitxer esborrat?**

El que ha passat és que rsync no ha eliminat l'arxiu arxiu\_nou.txt de /backup/backup-rsync després d’esborrar de /root l’arxiu perquè per defecte, rsync només sincronitza els fitxers i directoris que han canviat o s'han afegit, però no elimina fitxers a la destinació a menys que utilitzis l'opció específica per fer-ho (--delete).



**Amb quin paràmetre podríeu sincronitzar exactament els dos directoris?**

Per sincronitzar exactament els dos directoris hem d’emprar la comanda:

sudo rsync -avz --delete /root/ -e ssh root@localhost:/backup/backup-rsync/

Amb aquesta comanda, rsync sincronitzarà exactament els dos directoris, assegurant-se que els fitxers que falten a /root/ es sincronitzin a /backup/backup-rsync/ i que els fitxers esborrats de /root/ també esborrin de /backup/backup-rsync/.

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

Per copiar tots els fitxers del directori /home excepte els que tenen extensió .txt, podem utilitzar l'opció --exclude d'rsync, que permet especificar patrons per excloure certs fitxers. Amb la següent comanda copiariem tots els arxiu del directori /home excepte els que tenen extensió .txt.

sudo rsync -avz --exclude='\*.txt' /home -e ssh root@localhost:/backup/backup-rsync/

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

Quan executes rsync /source /destí, es copia tot el directori source com un subdirectori dins de /destí, de manera que el resultat serà /destí/source. En canvi, si utilitzes rsync /source/ /destí, només es copia el contingut del directori source directament a /destí, sense incloure el directori source mateix.

## **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

# Directori d'origen que vols sincronitzar

SOURCE\_DIR="/root"

# Directori de destí on es guardarà el backup complet

DEST\_DIR="/backup/backup-rsync"

# Fitxer que conté els patrons de fitxers/directoris a excloure

# Crea un fitxer (exemple: /backup/excludes.txt) amb les exclusions, una per línia.

EXCLUDES="/backup/excludes.txt"

# Nom o IP del servidor de backups

BSERVER="localhost"

# Nom del directori d'incrementals basat en la data i l'hora actual

BACKUP\_DATE=$(date +"%d-%m-%Y\_%H-%M-%S")

# Opcions per a rsync

OPTS="--ignore-errors --delete-excluded --exclude-from=$EXCLUDES \

--delete --backup --backup-dir=$DEST\_DIR/$BACKUP\_DATE -av"

# Realització de la transferència

rsync $OPTS $SOURCE\_DIR root@$BSERVER:$DEST\_DIR/complet



Un cop creat l’script hem de crear l’arxiu excludes.txt, per fer-ho fem:  
 sudo nano /backup/excludes.txt

I escrivim les següents linies:



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

Per crear l’arxiu arxiu.txt fem:

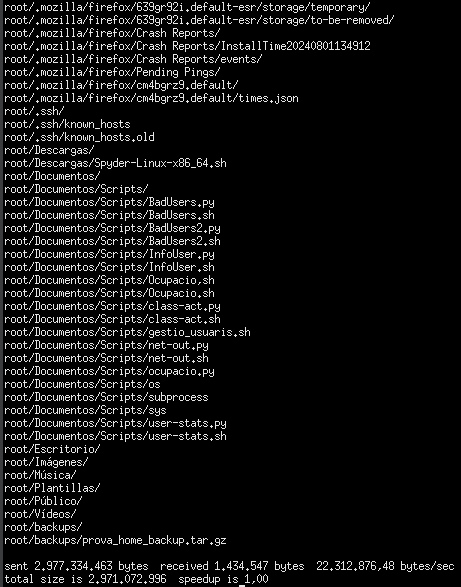


I per comprovar que s’ha creat correctament fem:

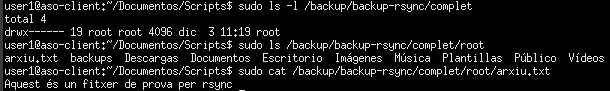


Un cop creat l’arxiu, executem l’script:





I verifiquem que ha funcionat tot correctament:



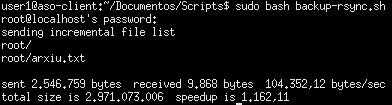
**Després modifiqueu aquest arxiu.txt i torneu a sincronitzar.**

Comencem modificant arxiu.txt fent:

sudo nano /root/arxiu.txt



Després tornem a executar l’script:



I comprovem el contigut d’arxiu.txt:



També podem observar com s’ha creat una copia de l’antic backup:

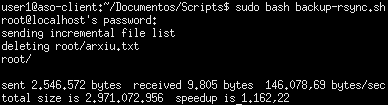


**Finalment esborreu el fitxer arxiu.txt i feu una sincronització més.**

Comencem esborrant arxiu.txt:



I executem de nou l’script:



I comprovem si arxiu.txt està o no en el backup:



Podem tornar a comprovar que s’ha creat una altra copia del backup anterior:



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

Quan modifiquem un fitxer, rsync actualitza la còpia al directori de destinació amb la versió més nova i desplaça la versió anterior al directori de còpies de seguretat.

Quan esborrem un fitxer, rsync elimina el fitxer de la destinació (gràcies a l'opció --delete), però manté una còpia de seguretat de la versió anterior en un directori separat (gràcies a l'opció --backup-dir).

## **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?**

Per comprovar que file\_a i file\_b es troben al mateix inode, hem de fer ús de la comanda **stat nom\_fitxer**, en el nostre cas:

**stat file\_a**

**stat file\_b**

**Què passa amb el fitxer file\_b si es fa un canvi a file\_a?**

Com que els dos son enllaços durs al mateix inode, qualsevol canvi al contingut de de l’arxiu es reflexara als dos fitxers (file\_a i file\_b).

**I si es fa un canvi de permisos a file\_a?**

Els arxius estàn associats a l’inode, per tant, si els canviem a file\_a, els estem canviant tambè a file\_b.

**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?**

Per defecte, cp sobreescriu el fitxer de destinació, en aquest cas file\_a, però manté el seu inode si és possible. Això significa que el contingut de file\_a serà reemplaçat pel de file\_c, i el canvi també afectarà file\_b perquè comparteixen el mateix inode.

**I què passa amb file\_b si file\_a és esborrat?**

Al eliminar file\_a, l’enllaç dur file\_b segueix existint, ja que l’inode no s’elimina fins que tots els enllaços durs siguin eliminats.

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

### **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:**

**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:**

## **Script per fer backups tipus snapshot**

Per poder emmagatzemar els backups tipus snapshot crearem una carpeta amb el nom snapshots dins del directori backup. Per fer-ho, primer hem de posar permisos d’escriptura.



Ara ja podem crear els directoris



Tornem a posar el directori backup en mode Read-only:



**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.**

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

MOUNT\_DEVICE="/dev/sda4"

SNAPSHOT\_MOUNTPOINT="/backup"

SNAPSHOT\_DIR="snapshots"

EXCLUDES="/root/excludes"

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/

Codi complet:

#!/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="/dev/sda4"

SNAPSHOT\_MOUNTPOINT="/backup"

SNAPSHOT\_DIR="snapshots"

EXCLUDES="/root/excludes"

SOURCE\_DIR="/root"

# ------------- 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 -a --delete --exclude-from=$EXCLUDES --link-dest=$SNAPSHOT\_MOUNTPOINT/$SNAPSHOT\_DIR/daily.1 $SOURCE\_DIR/ $SNAPSHOT\_MOUNTPOINT/$SNAPSHOT\_DIR/daily.0/

# 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;

**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.**

Primer de tot creem un nou arxiu:

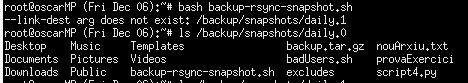
echo "Contingut inicial" > /root/nouArxiu.txt

Executem l’script per fer el backup:

bash backup-rsync-snapshot.sh

Mirem que s’hagi fer el backup correctament

ls /backup/snapshots/daily.0

****

Modifiquem el contingut de l’arxiu creat

echo "Contingut modificat" > /root/nouArxiu.txt

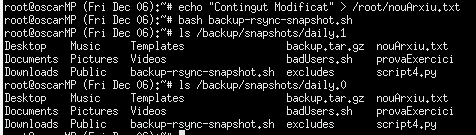
Executem l’script per fer el backup:

bash backup-rsync-snapshot.sh

Mirem que s’hagi fer el backup correctament

ls /backup/snapshots/daily.0

ls /backup/snapshots/daily.1

****

****

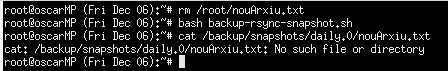
Ara eliminarem l’arxiu creat

rm /root/nouArxiu.txt

Executem l’script per fer el backup:

bash backup-rsync-snapshot.sh

I comprovem que es troba el nou backup sense l’arxiu:

****

****

Per últim, canviarem la data d'accés a qualsevol arxiu

touch /root/nom\_arxiu

Executem l’script per fer el backup:

bash backup-rsync-snapshot.sh

Comprovem que es crea un nou backup

ls /backup/snapshots

****

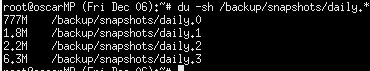
**Què observeu al modificar un fitxer i fer una sincronització?**

Quan es modifica un fitxer al directori d'origen i es torna a executar l’script podem veure que:

* L’ snapshot daily.0 reflecteix els canvis.
* Els snapshots anteriors mantenen la versió antiga del fitxer.

Això passa perquè l’ script utilitza l'opció --link-dest de rsync, que crea enllaços durs per als fitxers que no han canviat, però copia els fitxers modificats al snapshot més recent.

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



Podem veure que l’snapshot daily.0 ocupa més memòria que els altres perquè és el més recent i conté una còpia completa de l'estat actual del directori d'origen. Això inclou tots els fitxers nous o modificats des de l'última sincronització.

En canvi, els snapshots més antics (daily.1, daily.2, daily.3) ocupen menys espai perquè:

* Els fitxers que no han canviat entre snapshots no es copien de nou, sinó que es referencien mitjançant enllaços durs als snapshots més recents. Això redueix significativament l'espai ocupat.
* Els snapshots antics només emmagatzemen els canvis respecte al següent, i la seva mida disminueix.

**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.**

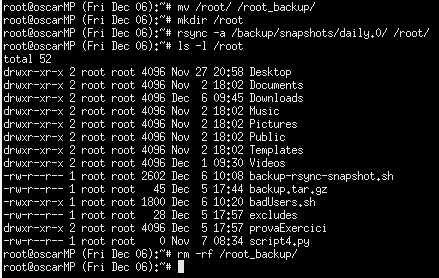
**mv /root /root\_backup**

**mkdir /root**

**rsync -a /backup/snapshots/daily.0/ /root/**

**ls -l /root**

**rm -rf /root\_backup**

****

# **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/*](http://www.rsnapshot.org/)

# **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;