NFS: Network File System

- palvelimella (tunnus2): apt-get install nfs-kernel-server

  /etc/exports:

  /home tunnus1(rw,sync,no_subtree_check,root_squash)

  exportfs -a

    jos virhe ”... not implemented”:

    service nfs-kernel-server restart
- asiakaskoneessa (tunnus1): apt-get install nfs-common

  mkdir /home2; mount tunnus2:/home /home2

    /etc/fstab:

    tunnus2:/home /home2 nfs defaults 0 0
- NFSv3: koneet luottavat toisiinsa, ei juuri tietoturvaa. NFSv4 lisää paljon tietoturvaominaisuuksia ja kaikenlaista muutakin.

Automounter (autofs)

- mounttaa nfs- tai muita verkkolevyjä tarpeen mukaan, ja umounttaa kun tarve poistuu
- asennus: apt-get install autofs
- mounttauksesta huolehtii demoni automount
- konfiguraatiotiedostot: /etc/auto.master ja siellä viitatut ns. kuvaustiedostot, map files (useimmiten /etc/auto.jotakin)
- suorat kuvaukset (direct maps): absoluuttinen polku kuvaustiedostossa, harvoin käytetty
- epäsuorat kuvaukset (indirect maps): kuvaustiedostossa suhteellinen polku, yhdistetään master-tiedoston polkumääritykseen
- säätömahdollisuuksia monimutkaisempiin ympäristöihin (LDAP jne)

autofs: esimerkki

- /etc/auto.master:

  /koti /etc/auto.koti --timeout=60

  /- /etc/auto.suora --timeout=180

- /etc/auto.koti:

  oma kotiserver:/home

  naapuri tulppu:/home

- /etc/auto.suora:

  /jako/peli peliserver:/pelijako

- Näkyvät hakemistot:

  /koti/oma /koti/naapuri /jako/peli

autofs: yleistä

- automountatut hakemistot tulevat näkyviin kun niitä käytetään; edellisessä esimerkissä komento “ls /koti” voi näyttää tyhjää, mutta “cd /koti/oma/tt” toimii kuitenkin (ja sen jälkeen ls toimii myös)
- automounter luo tarvittavat välihakemistot (edellä /koti jne) itse, niitä ei saa tehdä valmiiksi
- master-tiedoston muuttamisen jälkeen tarvitaan (esim)

  service autofs reload
- umount tapahtuu automaattisesti määräajan (oletus 5min) kuluttua siitä kun hakemistoja on viimeksi käytetty (tarvittaessa fuser tai lsof tai ps auttavat selvittämään käyttäjän)
- oletusasetukset tiedostossa /etc/default/autofs

autofs: wildcards, scripts, debug

- kuvaustiedostossa voi käyttää hakemistonimessä jokerimerkkejä * ja &:
  - * kotipalvelin:/home/&
- kuvaustiedosto voi olla skripti, joka tavalla tai toisella tuottaa halutun mounttikuvauksen (argumenttina levypalvelimen nimi); esimerkkinä /etc/auto.net, joka mounttaa mitä vain verkosta löytyy polkuun /net/kone/hakemisto
- debuggausta varten automount-demonia voi ajaa verbose-optiolla etualalla omassa ikkunassaan:

```
service autofs stop
/usr/bin/automount -f -v
```

shell-ohjelmointia: case

- case merkkijono in

  malli1) komentojono1;;

  malli2) komentojono2;;

  ...

  esac
- mallissa voi olla jokereita * ? [...] ja vaihtoehtoja erotettuna |:lla:

  case $(date +%a) in Mon) echo vihaan maanantaita;;

  T*|Wed) echo whatever;; Fri) echo kohta... ;; S*) echo “viikonloppu!”;;

  *) echo “kalenteriuudistus vai väärä kieli?” ;;

  esac
- käytetään usein yhdellä mallilla if-lauseen asemesta (lähinnä jokerien käytön helppouden ja yhdenmukaisuuden takia)

tar - “tape archive”

- paketoi kokoelman tiedostoja hakemistorakenteineen yhdeksi arkistotiedostoksi (tai nauhalle...)
- yleinen siirtoformaatti, käytetään myös varmuuskopiointiin
- huom. syntaksi: ei tarvitse viivaa (-) optioiden edessä
- arkiston luonti: tar c[v...]f *archive file...*

  tar cf /var/tmp/tthome.tar /home/tt
- arkiston purku: tar x[v..]f *archive [file...]*

  tar xjf /var/tmp/extra.tar.bz kala/hauki.txt
- arkiston sisällysluettelo: tar t[v..]f *archive*

  tar tvzf /var/tmp/odd.tgz
- paljon optioita erikoisempiin tilanteisiin

rsync

- remote sync, ideana kopioida koneesta toiseen vain muutokset (nopea)
- valittavissa vertaako vain tiedostojen kokoja ja aikaleimoja vai sisältöjäkin
- toimii ssh:n yli (yleisin tapa) tai omalla rsyncd-protokollallaan (asennettava demoniksi) tai minkä tahansa tiedostomountin kautta (myös lokaalisti, kätevä hakemiston toistuvaan kopiointiin)
- toimii sekä push- että pull-tyyliin (mutta ei suoraan kahden etäkoneen välillä)
- syntaksi muistuttaa scp:tä mutta tärkeitäkin eroja on

rsync syntaksi

- rsync src dest # lokaali (tai nfs-mountin yli tms)
- rsync user@host:src dest # ssh
- rsync src user@host:dest # ssh
- rsync user@host::src dest # rsyncd
- rsync rsync:user@host/src dest # rsyncd
- rsync src user@host::dest # rsyncd
- rsync src rsync:user@host/dest # rsyncd

- HUOM. lähteessä / lopussa merkitsevä: dir/ olennaisesti dir/*

rsync optiot

- Tässä vain tärkeimmät, man kertoo lisää:

  -r | --recursive

  -l | --links

  -p | --perms

  -t | --times

  -g | --group

  -o | --owner

  -D | --devices --specials

  -a | --archive | -rltpgoD

  -H | --hardlinks

  -n | --dry-run

Filesystem backup

- Kopioidaan tiedostoja käyttöjärjestelmätason komennoilla (cp/scp/tar/rsync yms) tai vastaavilla systeemikutsuilla
- Selkeä ja helppo ymmärtää ja käyttää, palauttaminen onnistuu käyttäjän oikeuksilla
- Yksittäisten tiedostojen palauttaminen helppoa
- Koherenssiongelma erityisesti tietokantojen kanssa (eri tiedostojen varmuuskopiot eri ajanhetkiltä); lvm snapshot (tai vastaava joissakin tiedostojärjestelmissä) auttaa
- Ei yleensä sovellu koko käyttöjärjestelmän varmistukseen (mikä ei usein ole tarpeenkaan)

Full vs. incremental backups

- Full backup: kopioidaan kaikki kerralla
- Incremental: kopioidaan vain aikaisemman varmuuskopioinnin jälkeen muuttuneet
  - Yleensä monta tasoa, esim. päivittäiset, viikottaiset, kuukausittaiset - tiedostosta löytyy vanhakin versio sitä varmemmin mitä kauemmin se on ollut olemassa
  - usein talletetaan edellisen saman tasoisen varmuuskopioinnin jälkeen muuttuneet
  - vaihtoehtoisesti talletetaan aina uusimman edellisen jälkeen muuttuneet ja tasot ovat vain säilytysajan merkkejä
  - palautus monivaiheinen (joka taso erikseen)

Backup-aikatauluesimerkki

Klassinen full-incremental voisi mennä näin:

- Kerran vuodessa full backup, säilytetään kolme vuotta
- Kerran kuussa incremental, siis edellisen kuukauden jälkeen muuttuneet, säilytetään 3kk
- Kerran viikossa incremental, säilytetään kuukausi (seuraavaan kuukausittaiseen saakka)
- Päivittäin incremental, säilytetään viikko
- Suunniteltu toimivaksi nauhojen tms vaihdettavan median kanssa; kovalevylle talletettaessa pitää jotenkin vaihtaa hakemistoa tai levyä

Backup-esimerkki: rsnapshot

- Kopiomediana vain kovalevy
- Käyttää (hard) linkkejä luomaan illuusion täydellisistä välikopioista vaikka kopioikin vain muuttuneita, palautus helppo (yksivaiheinen)
- Incremental-tasot vain säilytysaikamerkkejä, kopioi aina juuri edellisen tason jälkeen muuttuneet
- Pull-tyyppinen, ts. varmuuskopiopalvelin hakee tiedot varmistettavilta koneilta
- Ensisijainen etäkopiointimekanismi rsync + ssh (varmistettaville koneille ei tarvitse asentaa mitään muuta)
- Ei web-käyttöliittymää tms vaan käsin editoitava konfiguraatiotiedosto (vrt. BackupPC)

rsnapshot.conf

- Oletuskonfiguraatiotiedosto /etc/rsnapshot.conf, voi myös määritellä oman -c -optiolla (esim. jokaiselle varmistettavalle koneelle oma)
- Tärkeimmät asetukset:
  - snapshot_root päähakemisto, jonka alle (tasokohtaisiin alihakemistoihin) varmuuskopiot talletetaan
  - no_create_root luodaanko ko. hakemisto automaattisesti
  - retain taso n incremental-tason nimi ja säilytysmäärä
  - sync_first erottaa kopioinnin ja tasokierron
  - backup varmistettava data (hakemisto)
- Aikataulu erikseen, yleensä /etc/cron.d/rsnapshot

rsnapshot: hakemistorakenne

- rsnapshot_root'in alla tasohakemistot, niiden alla konehakemistot, esim.

  /rsnapshot/daily.0/kone1/home/...

  /rsnapshot/daily.0/kone1/etc/...

  /rsnapshot/daily.0/kone2/...

  ...

  /rsnapshot/daily.1/kone1/home/...

  ...

  /rsnapshot/weekly.0/kone1/home/...

  /rsnapshot/weekly.1/kone1/home/...

rsnapshot: hakemistorotaatio

- rsnapshot hoitaa tasot niin, että varmuuskopioinnin yhteydessä tasohakemistot nimetään uudelleen: ylimmän tason vanhin hävitetään (jos niitä on tarpeeksi), muiden tasojen viimeinen (suurinumeroisin) nimetään seuraavan tason nollahakemistoksi ja muiden numeroa kasvatetaan yhdellä. Esim.

  yearly.3 pois, yearly.1 → yearly.2, yearly.0 → yearly.1
  monthly.11 → yearly.0, monthly.10 → monthly.11.... weekly.3 → monthly.0,
  weekly.2 → weekly.3... weekly.0 → weekly.1
  daily.6 → weekly.0,daily.5 → daily.6...daily.0 → daily.1

- ensin siis tehdään ylimmän tason ajo (rsnapshot yearly tms), joka ei tee muuta kuin rotaation sillä tasolla, sitten toiseksi ylimmän jne

rsnapshot: kopiointivaihe

- ilman sync_first -optiota varsinainen kopiointi tehdään alimman tason rotaation yhteydessä (sen jälkeen) 0-hakemistoon (daily.0 tai hourly.0) ja virheen tapahtuessa rotaatio rullataan takaisin (vaihdetaan alimman tason hakemistonimet takaisin) - tämän pitäisi tapahtua automaattisesti (mutta jos kone kaatuu kesken...)
- sync_first -option kanssa kopiointi tehdään ennen alimman tason rotaatiota (rsnapshot sync) .sync-hakemistoon, ja rotaatiossa siitä tulee daily.0 (tai hourly 0 tms); tällöin virhetilanteessa ei tarvitse tehdä mitään erityistä, riittää kun käynnistää vain sync-ajon uudestaan

rsnapshot & cron

- esimerkki /etc/cron.d/rsnapshot (ilman sync_first -optiota), huomaa järjestys:

  30 3 * * * root /usr/bin/rsnapshot daily

  0 3 * * 1 root /usr/bin/rsnapshot weekly

  30 2 1 * * root /usr/bin/rsnapshot monthly

  - edellyttää riittävän pitkiä väliaikoja (rotaatiolle, ei kopioinnille)
- vaihtoehtoisesti tasovalinnan voi tehdä skriptin sisällä ja cron'illa vain käynnistää sen, esim.

  30 2 * * * root /root/backup.sh

rsnapshot script

```
#! /bin/bash
# /root/backup.sh
# rsnapshot w/ sync_first, daily...yearly
case $(date +%m%d) in 0101) rsnapshot yearly;; esac
case $(date +%d) in 01) rsnapshot monthly;; esac
case $(date +%u) in 7) rsnapshot weekly;; esac
rsnapshot sync
rsnapshot daily
exit 0
```

levyn salaus

- koko koneen tai levyn katoamisen/kierrättämisen varalta
- dm-crypt: levyn salakirjoitus, tuettu suoraan Linuxin kernelissä
- LUKS (Linux Unified Key Setup) avaimen hallintastandardi: varsinainen (pitkä, ei koskaan näkyvä) salausavain itse levyllä salattuna toissijaisella, ihmisen käyttämällä ja vaihdettavissa olevalla avaimella
- Voidaan salata oikeita tai virtuaalisia levyjä (voi olla tavallinen tiedostokin, jota sitten käsitellään levynä)
- komentorivityökalu cryptsetup, toimii vain roottina (sudolla)
- asennus: apt-get install cryptsetup-bin

cryptsetup esimerkki: tiedosto

- Tehdään tiedoston sisään salattu tiedostojärjestelmä:

  ```
  dd if=/dev/zero of=/home/tt/sala.img bs=1M count=10
  cryptsetup luksFormat /home/tt/sala.img
  cryptsetup luksOpen /home/tt/sala.img secret
  mkfs -t ext4 /dev/mapper/secret
  mkdir /jemma; mount /dev/mapper/secret /jemma
  # ...
  umount /jemma; cryptsetup luksClose secret
  ```

- aina tarvittaessa taas cryptsetup luksOpen...

cryptsetup esimerkki: LV

- Luodaan salattu looginen levy (LVM):

  lvcreate -L10M -nsala vg1

  cryptsetup luksFormat /dev/vg1/sala

  cryptsetup luksOpen /dev/vg1/sala sala_crypt

  mkfs -t ext4 /dev/mapper/sala_crypt

  mkdir /secret; mount /dev/mapper/sala_crypt /secret

  # ...

  umount /secret; cryptsetup luksClose sala_crypt

- Jos lisää /etc/fstab'iin mount ei onnistu ennen salauksen avaamista - bootti voi hyytyä (optio “noauto”, tai crypttab'iin)

cryptsetup esimerkki: PV

- Salataan fyysinen levy (LVM physical volume), tässä /dev/sdd:

  cryptsetup luksFormat /dev/sdd

  cryptsetup luksOpen /dev/sdd sdd_c

  pvcreate /dev/mapper/sdd_c

  vgextend vg1 /dev/mapper/sdd_c

- VG:tä ei saa nyt aktivoitua ennen salauksen avaamista (automatisoitavissa bootissa)
- Yleensä syytä salata VG:n kaikki PV:t tai ei yhtään
- systeemi-VG:n salaus edellyttää (toistaiseksi) salaamatonta /boot -osiota; usein koko / jätetään salaamatta

virtuaalikoneen salaus

- Koko virtuaalikoneen voi salata alustakoneen puolella
- Yksittäinen virtuaalikone voidaan asentaa salattuun levyosioon (tiedostoon, LV:hen, levypartitioon); tällöin sen (virtuaali)levyistä voidaan myös valikoiden salata vain joitakin
- Useita virtuaalikoneita voidaan salata asentamalla ne salatun LV:n alla oleviin tiedostoihin tai kokonaisen salatun VG:n (kaikki PV:t salattu) alle (omiin LV:hinsä tai tiedostoihin)
- Edellyttää root-oikeuksia alustakoneessa
- Salatunkin virtuaalikoneen sisällä voi tietysti vielä salata sen sisäisiä levyjä (siihen ei tarvita erioikeuksia alustakoneessa)

/etc/crypttab

- bootissa automaattisesti avattavaksi halutut salaukset (ehkä muutkin)

```
# <target> <device>   <key file>          <options>
sdd_c       /dev/sdd      none                luks
extra       /home/x.img none                luks,noauto
xswap       /dev/sdc2     /dev/urandom      swap
```

- mountataan automaattisesti bootissa ellei optioissa ole “noauto”
- mounttaus käsin: service cryptdisks start

  tai force_start (avaa myös noauto-laitteet)
- muutoksen jälkeen tarvitaan update-initramfs -u

cryptsetup: avainten hallinta

- avaimen voi antaa komentoriviltä tai tiedostossa
  - avain voi olla esim. USB-tikulla
- ei samaa salasanaa kahdelle laitteelle!
  - jos paljon levyjä (joilla on eri avaimet!) ne voi koota yhteen ylimääräiseen salattuun levyosioon niin, että bootissa kysytään vain sen avainta
- cryptsetup luksAddKey *laite [avaintiedosto]*
  - avaimia voi siis olla useita
- cryptsetup luksRemoveKey *laite [avaintiedosto]*
- cryptsetup luksKillSlot *laite avainnumero*
  - tällä voi poistaa tuntemattomankin avaimen
- jos kaikki avaimet katoavat, salausta ei saa auki millään!