# Tiedonsiirtosovellukset UNIXin kaltaisissa käyttöjärjestelmissä

Heikki Orsila <heikki.orsila@iki.fi>

# 1 Esitelmän aiheet

- Yleiset tiedonsiirtosovellukset UNIXin kaltaisissa käyttöjärjestelmissä
- Kopiointi verkon yli
  - Tavallinen kopioiminen
  - Kopioiminen skripteillä
  - Varmuuskopiointi
    - Täysi varmuuskopio
    - Olemassaolevan varmuuskopion päivitys
- Eheyden tarkistaminen
  - Tiedostojen suora vertaaminen
  - Tarkistussummiin vertaaminen

# 2 Aiheiden rajaus

- Allekirjoittaneen rajallinen kokemus- ja tietopohja.
- Esimerkkipainotteinen
- Komentorivinäkökulma
- *Tavallinen* käyttö (ei tuotantoympäristön automaatiota)

# 3 Protokollat ja tiedonsiirtotyökalut

- **BitTorrent** (BitTornado, BitTorrent, rTorrent, ...)
- FTP (ftp, **lftp**)
- HTTP (lftp, Links, **Wget**)
- **rsync** (käyttäen SSH:ta ja ilman)
- **SSH** (scp, sftp, ssh, ...)
- IRC (Irssi, XChat)
- TCP-työkalut (nc, socat)
- Verkkotiedostojärjestelmät (**sshfs** (FUSE))

## 3.1 BitTorrent

### 3.1.1 Ominaisuudet

- Perinteisissä protokollissa tiedostot ladataan keskitetyiltä palvelimilta

  $\rightarrow$ Suosittu tieto ruuhkauttaa palvelimet

- BitTorrent on vertaisverkkoprotokolla verkkokuorman hajauttamiseen. Se mahdollistaa suurien tietomäärien levittämisen pienillä tietoliikenneresursseilla (GNU/Linux distribuutiot, *Star Wreck*, ...).

- BitTorrentissa tieto siirretään muilta käyttäjiltä

  $\rightsquigarrow$ Julkaisija ei tarvitse suuria resursseja

- Käyttäjien joukossa on *lähteitä*, jotka toimivat samalla tavalla kuin perinteiset keskitetyt palvelimet. Kaikki tieto on alunperin hankittu *lähteistä*. Tiedon julkaisija pystyttää aluksi yhden tai useamman lähteen, jotta tieto olisi saatavilla verkossa.

- Käyttäjän ei tarvitse tietää BitTorrentin teknisiä yksityiskohtia, mutta on välttämätöntä tiedostaa, että hakiessaan tietoa itselleen, hän samalla myös kopioi sitä muille. Tiedonsiirto ei lopu siihen kun kaikki tiedostot on ladattu. Siinä vaiheessa käyttäjästä tulee uusi *lähde*.

- Tiedostoja imuroidaan *torrent*-tiedostojen avulla. Torrent -tiedostossa on tarvittavat ohjeet tiedoston noutamiseksi internetistä. Erityisesti siihen kuuluu ns. *träkkerikoneen* nimi. Träkkerikone huolehtii siitä että samoja tiedostoja hakevat käyttäjät tuntevat toisensa, jotta he voisivat vaihtaa dataa keskenään.

- *Torrent*-tiedostoja jaetaan tavallisesti seittisivuilla

- BitTorrentin siirtonopeus on tilastollisesti suoraan verrannollinen lähtevän liikenteen määrään.

- Sallimalla TCP-yhteydet sisäänpäin portteihin 6881 - 6999 (ks. btdownloadcurses.py -h) käyttäjä pystyy vaihtamaan tietoa kaikkien BT-verkon käyttäjien kanssa. Porttialueen pystyy säätämään --

minport ja --maxport vipujen avulla. Yhteyksien salliminen sisäänpäin saattaa nopeuttaa tiedonsiirtoa huomattavasti.

- Jos käyttäjä haluaa antaa omia tiedostojaan jakoon muille, luo hän torrent -tiedoston niistä. Torrent -tiedostossa täytyy olla olemassaolevan träkkerikoneen nimi. Tiedostojen levitys ei kuitenkaan onnistu ellei träkkerin ylläpitäjä ole erikseen sallinut käyttäjän tekemää torrent -tiedostoa. Käyttäjä voi joko pyytää tietyn träkkerin ylläpitäjää sallimaan hänen torrent -tiedostonsa tai vaihtoehtoisesti käyttäjä pystyttää oman träkkerin.

### 3.1.2 Tiedostojen lataaminen BitTorrentilla

Imuroidaan tiedosto lähettäen muille käyttäjille tietoa enintään 32 KiB/s nopeudella:

```
# btdownloadcurses.py --max_upload_rate 32 foo.torrent
```

Hankitaan käyttöjärjestelmä. Ei lähetysnopeuden rajoitusta muille käyttäjille.

```
# btdownloadcurses.py --url http://torrents.gentoo.org/torrents/\
stage3-amd64-2006.0.torrent
```

Imuroidaan kaikki tiedostot, joille löytyy vastaava torrent -tiedosto hakemistosta /minun/torrent/hakemisto. Ohjelma havaitsee jos hakemistoon ilmestyy uusia torrent -tiedostoja ja noutaa myös ne.

```
# btlaunchmanycurses.py /minun/torrent/hakemisto
```

### 3.1.3 Tiedostojen julkaiseminen BitTorrentilla

Oletetaan että käyttäjä haluaa luoda “foo” -hakemiston sisällöstä torrent -tiedoston, träkkerikoneen nimi on “bar.invalid”, ja träkkeri sijaitsee portissa 2000. Käyttäjä luo torrent -tiedoston seuraavasti:

```
# btmaketorrent.py http://bar.invalid:2000/announce foo
```

Tuloksena syntyy “foo.torrent” niminen tiedosto, jonka hän kopioi träkkerin ylläpitäjälle. Träkkerin voi pystyttää seuraavasti:

```
1. mkdir /tmp/tracker
2. cp foo.torrent /tmp/tracker
3. bttrack.py --port 2000 --allowed_dir /tmp/tracker \
   --dfile /tmp/downloaderinfo
```

Träkkeri sallii kaikki torrent -tiedostot levitykseen hakemiston “/tmp/tracker” alta. Jaossa olevat torrent -tiedostot voi tarkistaa selaimella osoitteesta *http://bar.invalid:2000*.

## 3.2 FTP (*File Transfer Protocol*)

- FTP (engl. *File Transfer Protocol*) -protokolla toimii keskitetyn mallin mukaisesti. Käyttäjät ottavat yhteyden FTP -palvelimeen ja lataavat tiedostot sieltä

- Julkisia FTP -palvelimia voi käyttää *anonyymisti* ilman käyttäjätunnusta tai salasanaa. Käyttäjätunnukseksi annetaan *anonymous* ja salasanaksi mikä hyvänsä merkkijono (joskus vaatimuksena oli että merkkijono näyttää sähköpostiosoitteelta, eli siinä täytyi olla @-merkki). Tavalliset asiakasohjelmat tekevät anonyymin sisäänkirjautumisen automaattisesti.

- FTP:tä käytetään myös yksityiseen tiedostojen kopioimiseen, jolloin käytetään oikeaa käyttäjätunnusta ja salasanaa. Turvallisuuden puolesta on suositeltavaa käyttää salattua jotain tiedonsiirtoprotokollaa SSH:n päällä.

  - Myös uudemmat FTP-palvelimet ja asiakasohjelmat tukevat salausta, mutta ohjelma ei välttämättä kerro käyttäjälle selvästi

onko salaus käytössä vai ei. Käyttäjän on syytä tarkistaa onko salaus käytössä ennen kuin syöttää salasanansa palvelimille.

- Erilaisia FTP-palvelimia ja asiakasohjelmia on paljon, mutta vain harvat niistä ovat yleistyneet.

- Myös selaimet tukevat FTP-protokollaa ja siksi erityistä FTP-ohjelmaa ei välttämättä tarvita olleenkaan. Mikäli käyttäjän tarvitsee lähettää palvelimelle tiedostoja, tarvitsee hän todennäköisesti FTP-ohjelman. Selaimissa FTP-protokollaa voi käyttää osoitteella, joka alkaa ftp://. Esimerkiksi ftp://ftp.funet.fi

- FTP-protokollassa on historiallisista syistä käsite teksti- ja binäärisiirroille. Binääritiedostojen lataaminen tekstitilassa aiheuttaa tiedostojen rikkoutumisen. Käyttäjän on syytä tarkistaa, että asiakasohjelma on binääritilassa ennen siirtoa. Tyypillisesti se tehdään *bin*-komennolla, mutta uudet asiakasohjelmat tekevät tämän automaattisesti.

### 3.2.1 ftp

*ftp* on eräs vanhimmista FTP-asiakasohjelmista. Ohjelma saatavilla osoitteesta ftp://ftp.uk.linux.org/pub/linux/Networking/netkit/. Ohjelmassa on paljon erilaisia komentoja, mutta tavallisesti käytettyjä komentoja on vähän. Taulukossa 1 on lueteltu tavallisia käskyjä.

Haetaan esimerkin vuoksi Linux-ytimen lähdekoodi ftp.funet.fi palvelimelta.

```
# ftp ftp.funet.fi
Connected to ftp.funet.fi (193.166.3.2).
...
Name (ftp.funet.fi:foo):
```

Tässä vaiheessa käyttäjä syöttää käyttäjätunnuksen ja salasanansa.

```
Name (ftp.funet.fi:foo): anonymous
331-Welcome to the FUNET anonymous ftp archive
...
Password: <käyttäjä syöttää tähän mitä haluaa>
```

Taulukko 1: *ftp*-ohjelman komentoja selityksineen

| Komento | Selostus |
|---|---|
| **!komento** | Ajetaan paikallinen komentorivikäsky. Esim. !pwd kertoo paikallisen oletuspolun. |
| **bin** | Siirtyy binääritilaan |
| **cd** | Vaihtaa hakemistoa |
| **help** | Tulostaa mahdolliset komennot |
| **help** komento | Tulostaa tietystä komennosta ohjeita |
| **lcd** /paikallinen/hakemisto | Vaihdetaan hakemistoa paikallisella koneella |
| **ls** | Listaa nykyisen hakemiston |
| **mget** tiedosto1 tiedosto2 ... | Lataa monta tiedostoa. Esim. mget *.txt |
| **mput** tiedosto1 tiedosto2 ... | Lähettää monta tiedostoa. |
| **prompt** no | Poistaa tarpeettomat kyllä/ei kyselyt. Hyödyllinen esim. mgetin kanssa. |
| **pwd** | Tulostaa polun FTP-palvelimella |

Tässä vaiheessa käyttäjä pääsee komentotilaan. Katsotaan **pwd** käskyllä missä tiedostopolussa ollaan palvelimella:

```
ftp> pwd
257 "/" is your current location
```

Ollaan siis juuressa. Yleinen tapa on laittaa jaettava julkinen tieto */pub* hakemiston alle. Listataan siis juurihakemisto, jotta nähdään onko *pub* hakemisto olemassa:

```
ftp> ls
...
drwxrwxr-x   45 108      200           1536 Mar 29 22:16 pub
...
```

*/pub* on olemassa. Säästäen turhia tulostuksia, mennään **cd** käskyllä suoraan lopulliseen hakemistoon.

```
ftp> cd pub/Linux/kernel/v2.6
250 OK. Current directory is /pub/linux/kernel/v2.6
```

Tässä vaiheessa käyttäjä tavallisesti tarkistaa **ls** käskyllä mitä tiedostoja hakemistosta löytyy, mutta tämänkin tiedämme entuudestaan, koska Linux-ytimien nimeäminen on säännöllistä. Käytämme **prompt** käskyä hiljentämään turhat kyllä/ei kyselyt, aloitamme siirron **mget** käskyllä ja lopuksi lopetamme **quit** käskyllä.

```
ftp> prompt off
Interactive mode off.
ftp> mget linux-2.6.16.1.tar.bz2
local: linux-2.6.16.1.tar.bz2 remote: linux-2.6.16.1.tar.bz2
200 PORT command successful
150-Connecting to port 38671
150 39883.8 kbytes to download
226-File successfully transferred
226 3.442 seconds (measured here), 11.32 Mbytes per second
40841031 bytes received in 3.46 secs (11539.7 kB/s)
ftp> quit
```

### 3.2.2 lftp

- Monipuolinen FTP-ohjelma (*http://lftp.yar.ru*)
- Samanaikaiset yhteydet moneen palvelimeen
- Samanaikaisten töiden joustava hallinta (ajastus, jonotus, odottaminen ja poistaminen)
- Tuki FTP, HTTP ja sftp protokollille
- Vikasietoinen (yrittää operaatiota uudestaan korjaantuvan vian ilmaantuessa)
- Tiiviisti integroitu komentorivitulkkiin
- Siirtonopeuden rajoittaminen

Teemme saman asian lftp:llä mitä teimme ftp:llä:

```
# lftp ftp.funet.fi
lftp ftp.funet.fi:~> cd /pub/Linux/kernel/v2.6
cd ok, cwd=/pub/Linux/kernel/v2.6
lftp ftp.funet.fi:/pub/Linux/kernel/v2.6> mget linux-2.6.16.1.tar.b
40841031 bytes transferred in 11 seconds (3.51M/s)
lftp ftp.funet.fi:/pub/Linux/kernel/v2.6>
```

Kuten esimerkistä huomataan, lftp käyttäjän ei tarvitse käsin syöttää käyttäjätunnusta eikä salasanaa silloin kun kyseessä on anonyymipalvelin.

**Vinkki:** Tiedostonimiä ei tarvitse syöttää kokonaan käsin. Riittää kirjoittaa nimen alkuosa ja painaa sarkainta. Tällä on sama vaikutus kuin komentorivitulkissakin, lftp päivittää tiedoston (tai hakemiston) nimestä loput.

Sisäänkirjautuminen käyttäjätunnuksen kanssa tapahtuu joko näin:

```
# lftp foo@host.invalid
```

```
Password: <syötetään salasana>
lftp foo@host.invalid:~>
```

tai epäsuorasti näin:

```
# lftp
lftp :~> open foo@host.invalid
Password: <syötetään salasana>
lftp foo@host.invalid:~>
```

Salasanan voi syöttää myös suoraa komentoriviltä tai antaa sen open -käskylle, mutta yleisesti ottaen ei ole suositeltavaa kirjoittaa salasanaa komentoriviparametriksi, sillä jokin toinen käyttäjä voi nähdä sen (joko olan takaa tai prosessilistauksesta).

```
# lftp
lftp :~> open foo:bar@host.invalid
lftp foo@host.invalid:~>
```

lftp ei ole rajoitettu ftp-protokollaan, vaan sitä voi käyttää myös http- ja sftp-ohjelmana (SSH:n päälle rakennettu ftp:n korvike):

```
# lftp http://ftp.fi.debian.org
cd ok, cwd=/
lftp ftp.fi.debian.org:/> ls
drwxr-xr-x  --  debian-cd
drwxr-xr-x  --  debian-ipv6
drwxr-xr-x  --  debian-minicd
...
-rw-r--r--  --  ubar.png
lftp ftp.fi.debian.org:/>

# lftp sftp://user@host1.invalid
Password: <syötetään salasana>
lftp user@host1.invalid:~> cat /etc/hostname
host1
lftp user@host1.invalid:~>
```

Jälkimmäisessä esimerkissä *cat* -komento toimii samalla tavalla kuin vastaava UNIX-komento. lftp pyrkii käyttäytymään samalla tavalla kuin peruskomentorivi. lftp:n omia toimintoja voi helposti yhdistellä paikal-

listen komentojen kanssa. Esimerkki: käytetään lftp:tä grepin kanssa:

```
lftp user@host1.invalid:~> cd /etc
lftp user@host1.invalid:/etc> ls |grep host
-rw-r--r--    1 root     root         1302 Nov 14 04:16 host.conf
-rw-r--r--    1 root     root            6 Mar 26  2005 hostname
-rw-r--r--    1 root     root          271 Sep 15  2005 hosts
-rw-r--r--    1 root     root           10 Apr 15  2004 hosts.allow
-rw-r--r--    1 root     root            0 Apr 15  2004 hosts.deny
lftp user@host1.invalid:/etc> ls |less
drwxr-xr-x   64 root     root         8192 Apr 12 21:12 .
drwxr-xr-x   26 root     root         4096 Feb 18 18:45 ..
-rw-------    1 root     root            0 Feb 23  2004 .pwd.lock
-rw-r--r--    1 root     root         3572 Apr 11  2005 DIR_COLORS
drwxr-xr-x   18 root     root         4096 Jan  6 02:49 X11
...
lines 1-19 <pysähtyy odottamaan käyttäjän syötettä>
```

Komentojen kannalta ei ole merkitystä onko käytössä ftp-, http- vai sftp-protokolla.

Seuraavassa esimerkissä käytetään lftp:n jonotustoimintoa. Käyttäjä aloittaa hakemalla uusinta 2.4.x sarjan Linux-kerneliä, mutta kesken imuroinnin muistaa tarvitsevansa myös uusimman 2.6.x kernelin toista konetta varten. Toista siirtoa ei suinkaan tarvitse pistää rinnalle (se haittaisi verkon vasteellisuutta) eikä ensimmäistä tarvitse myöskään keskeyttää, käyttäjä voi pistää käynnissäolevan siirron taustalle ja jonottaa uuden siirtotyön tapahtumaan ensimmäisen siirron jälkeen:

```
lftp ftp.funet.fi:/pub/Linux/kernel/v2.4> mget linux-2.4.32.tar.bz2
<CTRL-z siirtää käyttäjän takaisin komentotilaan. Ohjelma tulostaa
taustalle menevän työn. Työn numero on nolla ([0]).>
[0] mget linux-2.4.32.tar.bz2 &
        ‘linux-2.4.32.tar.bz2’ at 280800 (0%) 91.1K/s eta:6m ...
lftp ftp.funet.fi:/pub/Linux/kernel/v2.4> cd ..
lftp ftp.funet.fi:/pub/Linux/kernel> cd v2.6
```

Käyttäjä katsoo taustatyön edistymistä:

```
lftp ftp.funet.fi:/pub/Linux/kernel/v2.6> j
[0] mget linux-2.4.32.tar.bz2
        ‘linux-2.4.32.tar.bz2’ at 855100 (2%) 97.8K/s eta:5m ...
```

Käyttäjä luo uuden jonon, joka siirtää 2.6.x kernelin työn 0 valmistumisen jälkeen. Käyttäjä tietää taustatyön olevan numero 0, joten uuden jonon ensimmäiseksi komennoksi tulee odotuskäsky joka päättyy kun työ numero 0 valmistuu.

```
lftp ftp.funet.fi:/pub/Linux/kernel/v2.6> queue wait 0
```

Tämän jälkeen lisätään jonoon varsinainen siirto-operaatio.

```
lftp ftp.funet.fi:...> queue mget linux-2.6.16.2.tar.bz2
```

Käyttäjä voi katsoa jonossa olevat työt. 'wait 0' on suorituksessa, ja seuraava komento jonossa on 'mget linux-2.6.16.2.tar.bz2'.

```
lftp ftp.funet.fi:/pub/Linux/kernel/v2.6> j -v
[1] queue (ftp://ftp.funet.fi)
        ftp://ftp.funet.fi/pub/Linux/kernel/v2.6
        Now executing: [0] mget linux-2.4.32.tar.bz2
        Commands queued:
         1. mget linux-2.6.16.2.tar.bz2
 [0] mget linux-2.4.32.tar.bz2
```

```
ftp://ftp.funet.fi/pub/Linux/kernel/v2.4
‘linux-2.4.32.tar.bz2’ at 1990500 (6%) 97.7K/s eta:5m ...
```

Taulukko 2: Tavallisia *lftp*-komentoja

| Komento | Selitys |
|---|---|
| **!komento** | Suorittaa paikallisen komentorivikomennon |
| **cat** | Tulostaa tiedoston terminaaliin |
| **cd** | Vaihtaa hakemistoa etäkoneella |
| **du** | Laskee tiedoston tai hakemiston koon. |
| **help** | Tulostaa ohjeita |
| **jobs** | Näyttää käynnissä olevat työt |
| **lcd** | Vaihtaa paikallista hakemistoa (johon tiedostot latautuvat) |
| **ls** | Listaa hakemiston |
| **mget** | Lataa tiedoston |
| **mirror** | Lataa kokonaisen hakemistorakenteen etäkoneelta |
| **pwd** | Tulostaa nykyisen hakemiston |
| **queue** | Lisää ja poistaa töitä jonosta |
| **rm** | Poistaa tiedoston tai kokonaisen hakemistorakenteen (-r) |
| **set** | Asettaa ohjelman sisäisiä valintoja. set -a tulostaa kaikki mahdolliset valinnat. |
| **slot** | Luo uuden rinnakkaisen komentorivikontekstin |

## 3.3 HTTP (*HyperText Transfer protocol*)

HTTP (engl. *HyperText Transfer Protocol*) on yleisimmin tunnettu protokolla yhteyksien muodostamiseen ja tiedostojen siirtämiseen koneiden välillä. Tämä kenties siksi, että seittisivut toimivat HTTP-protokollan avulla.

Tavallinen komentorivikäyttäjä tuskin hyötyy paljon seitistä, mutta komentoriviltä pystyy tekemään paljon sellaisia asioita, mitä selaimen käyttöliittymä ei mahdollista kätevästi. Näitä asioita ovat:

- Sivujen ja tiedostojen hakeminen skripteillä
- Kokonaisten sivustojen peilaaminen
- Tarkempi kontrolli isojen tiedostojen lataamiseen (tavallinen selain saattaa yrittää tallentaa gigatavun kokoisen tiedoston tilapäispaikkaan ennen lopullista sijaintia, mutta tilapäispaikassa ei riitäkään tila)
- Etäkoneiden hallinta (tiedoston voi hakea suoraa etäkoneelle)

### 3.3.1 Links

*Links* on komentorivipohjainen selain, joka toimii parhaiten isolla terminaalilla ja vahvasti tekstipitoisilla sivuilla. Käyttöesimerkki:

```
# links slashdot.org
```

Taulukossa 3 on listattuna tavallisimpia komentoja *Linksiin*.

Taulukko 3: Tavallisia näppäinkomentoja *Linksiin*

| Painallus | Toiminto |
|---|---|
| **Page Up / b** | Sivu ylöspäin |
| **Page Down / Space** | Sivu alaspäin |
| **Cursor down** | Siirrä valitsin seuraavan linkin kohdalle |
| **Cursor up** | Siirrä valitsin edellisen linkin kohdalle |
| **F9** | Näytä valikko eri toiminnoille |
| **/** | Etsi sivulla (suunta eteenpäin) |
| **?** | Etsi sivulla (suunta taaksepäin) |
| **d** | Lataa valitsimen osoittama linkki |
| **g** | Antaa valikon uuden URLin kirjoittamiseen |
| **G** | Muuta nykyistä osoitetta |
| **q** | Lopeta |

### 3.3.2 Wget

*Wget* on työkalu linkkien lataamiseen komentoriviltä tai skripteistä. Ohjelma hallitsee HTTP-, HTTPS- ja FTP-protokollat.

Esimerkkejä:

- Tehdään kopio kokonaisesta sivustosta (sivut kuvineen):

```
# wget -r --no-parent http://host/path
```

- Ladataan yksi tiedosto:

```
# wget http://host/file.tar.gz
```

- Ladataan kaikki *.png nimiset tiedostot sivustolta:

```
# wget -r -A '*.png' http://host/path
```

## 3.4 rsync

*rsync* on työkalu tiedostojen siirtämiseen koneelta toiselle. Tavallisesti rsynciä käytetään SSH:n päällä. rsync toimii myös ilman SSH:ta käyttäen erillistä rsync-palvelinta. rsyncin ominaisuuksia:

- Siirtää vain uudet ja muuttuneet tiedostot
- Muuttuneista tiedostoista siirretään vain muutokset
- Toimii kaikkien koneiden välillä jonne käyttäjällä on SSH-tunnus
- Ymmärtää tavalliset tiedostot, laitetiedostot ja symboliset ja kovat linkit
- Äärimmäisen nopea tapa tehdä päivitys isolle määrälle tiedostoja (silloin kun on tapahtunut vain vähän muutoksia). Perinteinen tiedonsiirtoprotokolla voi tehdä kymmenien tai jopa satojen millisekuntien viipeen jokaisen siirretyn tiedoston jälkeen. Tuhansien tiedostojen kanssa tästä kertyy pitkä odotus. Rsync-protokolla on suunniteltu siten, että kahden osapuolen välillä ei tarvita turhia kuittauksia, jolloin vältytään verkkoviipeeltä.

- Esimerkki: Gentoo Linuxin Portagessa on noin $140k$ tiedostoa ja käyttäjät päivittävät sitä rsyncin avulla. Portagen koko on noin 120 MiB. Käyttäjä siirtää siitä vain pienen murto-osan päivityksen yhteydessä.

Lyhykäisyydessään rsyncin syntaksi:

```
rsync [-vivut] lähde/lähteet kohde
```

Lähteet ja kohde ovat tiedostoja tai hakemistoja paikallisella tai etäkoneella. Jos lähde tai kohde ei ole paikallinen tiedosto tai hakemisto, annetaan se muodossa [user@]host:[polku]. Mikäli käyttäjätunnus on sama kummallakin koneella ei "user@" osaa tarvitse antaa. Mikäli hakemisto etäkoneella on kotihakemisto, ei "polku" osaa tarvitse antaa.

### 3.4.1 Varmuuskopiointi kahden koneen välillä

SSH-tunnuksen avulla käyttäjä voi helposti tehdä varmuuskopion etäkoneesta omalle koneelleen tai toisinpäin. Oletetaan että käyttäjällä on jo ennentään olemasta vanha versio etäkoneen kotihakemistosta polussa /varmuuskopiot/home/. *rsync* siirtää vain uudet ja muuttuneet tiedostot:

```
# rsync -avP root@host:/home /varmuuskopiot/
user@host's password: <käyttäjä syöttää salasanansa>
receiving file list ...
17571 files to consider
home/user1/
home/user2/
...
wrote 29544 bytes  read 9702497 bytes  452653.07 bytes/sec
total size is 815979458  speedup is 83.84
```

Komento kertoo paljonko tavuja olisi siirretty, jos kaikki tieto olisi siirretty kerralla. Tässä tapauksessa siirrettiin vain 1.2% siitä määrästä.

Toisessa esimerkissä kopioidaan hakemisto toisen koneen kotihakemistoon. Tässä esimerkissä käyttäjätunnusta ei tarvitse antaa, koska ole-

tetaan että se on sama lähdekoneella. Myöskään kohdehakemistoa ei tarvitse antaa, koska kotihakemisto on käyttäjän oletushakemisto:

```
# rsync -avP foobar host:
Password:
...
```

### 3.4.2 Varmuuskopiointi kahden kovalevyn välillä

*rsync* toimii paikallisesti aivan samalla tavalla kuin verkonkin ylitse:

```
# rsync -avP /home /varmuuskopiolevy/
building file list ...
150513 files...
...
wrote 123287237 bytes  read 21100 bytes  1239279.77 bytes/sec
total size is 8053036449  speedup is 65.31
```

Jos käyttäjä ajaa saman komennon uudestaan heti perään, siirtyy toisella kerralla huomattavasti vähemmän tietoa (noin 2% ensimmäisestä kerrasta):

```
# rsync -avP /home /varmuuskopiolevy/
building file list ...
150513 files...
...
wrote 2958071 bytes  read 20 bytes  51445.06 bytes/sec
total size is 8053036489  speedup is 2722.38
```

Taulukko 4: Hyödyllisiä rsync -vipuja

| Vipu | Kuvaus |
|---|---|
| --bwlimit=KBPS | Rajoittaa siirtonopeuden tiettyyn KiB/s nopeuteen |
| -a | tarkoittaa *archive* tilaa. Siinä kahden osapuolen tiedostot pyritään kopioimaan mahdollisimman tarkasti (säilyttäen tiedosto -oikeudet ja käyttäjät). |
| -c | Pakottaa tarkistussummien uudelleenlaskemisen kummassakin päässä. Tämä on hyödyllinen tilanteessa, jossa kohdetiedoston koko on sama kuin lähteessäkin, mutta muutospäiväys on uudempi. |
| --delete | Poistaa kohteesta ne tiedostot, joita ei löydy lähteestä |
| -e “cmd” | Ajetaan rsynciä “cmd” käskyn päällä. Oletuksena tämä on “ssh”. Esimerkki: “ssh -c blowfish” voisi vähentää suoritinkuormaa. |
| -n | Ei tehdä muutoksia kohteeseen, mutta katsotaan mitä tapahtuisi |

Taulukko 5: Lisää hyödyllisiä rsync -vipuja

| Vipu | Kuvaus |
|---|---|
| -P | käskee rsyncin tulostamaan tietoa siirtonopeudesta sekä säilyttämään osittaiset tiedostot mikäli verkkoyhteyskatkeaa kesken siirron |
| -v | käskee rsynciä kertomaan tarkasti mitä on tekemässä (tulostaa uusien ja muuttuneiden tiedostojen nimet ja lopuksi yhteenvedon) |
| -z | Laittaa tiivistyksen päälle. Nopeuttaa siirtoa hitaiden yhteyksien, mikäli tieto on tiivistettävissä |

## 3.5 SSH (engl. *Secure Shell*)

SSH (engl. *Secure Shell*) -protokolla on yleisin käytetty tapa hallita etäkoneita komentorivin avulla. OpenSSH (*http://openssh.org*) on yleisin toteutus protokollasta. OpenSSH sisältää sekä palvelimen että asiakasohjelman etäyhteyksiä varten. SSH:n ominaisuuksia:

- Tarjoaa komentoriviyhteyden toiselle koneelle (man ssh)
  - Mahdollisuus skriptata asioita etäkoneille niin ettei tarvitse syöttää salasanaa käsin: käytetään avaintiedostoja (man ssh-keygen) tai ssh-agenttia (man ssh-agent)
  - Voidaan käyttää komentoriviputken osana, jolloin jokin komentoriviputken osa suoritetaankin etäkoneella
- Vahvasti salattu yhteys
- Tarjoaa tiedostonsiirtomahdollisuuden (man scp, sftp)
- Tarjoaa mahdollisuuden portinkopiointiin palomuurien kiertämiseksi

### 3.5.1 ssh

*ssh* käsky tarjoaa kaksisuuntaisen salatun yhteyden etäkoneelle. Syntaksi on yksinkertainen:

```
ssh [-vivut] [user@]host [komento]
```

Jos käyttäjätunnus on sama etäkoneella, voidaan käyttäjätunnus jättää antamatta:

```
[user@foo ~]# ssh bar
Password:
Last login: Tue Apr 18 19:17:15 2006 from baz.host.invalid
[user@bar ~]#
```

Jos käyttäjätunnus on eri kohteessa:

```
[user@foo ~]# ssh anoncoward@bar
...
[anoncoward@bar ~]#
```

*ssh*:n avulla voi laukaista komentoja ajoon etäkoneelle ilman että ne tarvitsee käynnistää kohteessa komentoriviltä:

```
# ssh bar uname -a
Password:
Linux bar 2.6.15-rc7 #1 Fri Dec 30 04:19:20 EET 2005 x86_64 AMD \
Athlon(tm) 64 Processor 3000+ AuthenticAMD GNU/Linux
#
```

Käyttämällä *tar* -komentoa on helppo tehdä kopio etänä olevista tiedostosta (rsyncin käyttö on kuitenkin suositeltavaa tähän verrattuna):

```
# ls
# ssh host tar cv /dir |tar x
Password:
tar: Removing leading ‘/’ from member names
/dir/
/dir/file1
/dir/file2
# ls
dir
#
```

Esimerkki cd:n polttamisesta etäkoneelta:

```
# ssh host mkisofs -f -J -l -R /dir |cdrecord -
```

Tiedostoja ei tarvitse siirtää paikalliselle koneelle ISO9660-kuvan tekemiseksi. ISO9660-kuva voidaan luoda etäkoneella ja putkittaa suoraan ssh:n yli paikallisen koneen poltto-ohjelmalle.

Hyödyllisiä ssh-vipuja:

-6 IPv6-yhteys

-c Asettaa salausalgorithmin (-c blowfish voi vähentää suoritinkuormaa)

-X Ottaa käyttöön X11-välityksen jolloin on mahdollista ajaa etäkoneella myös X-ohjelmia.

### 3.5.2 scp

*scp* on työkalu tiedostojen kopioimiseen SSH:n päällä. Työkalu ei ole tarkoitettu vuorovaikutteiseen siirtämiseen (toisin kuin *sftp*). Toiminnoiltaan scp on hyvin yksinkertainen. Valitettavasti scp kärsii suorituskykyongelmista, joista *rsync* ei kärsi. Siksi scp:n asemesta kannattaa yleensä käyttää *rsynciä*, mutta scp löytyy oletuksena useammalta koneelta kuin rsync.

Sivuutetaan tarkempi ohjeistus scp:n ominaisuuksista, ja esitetään pari esimerkkiä. Kopioidaan yksi tiedosto ja sen jälkeen yksi hakemisto kohdekoneelle:

```
# scp file1 host:/polku/
Password:
...
# scp -r dir1 host:/polku/
Password:
...
```

### 3.5.3 sftp

*sftp* on vuorovaikutteinen tiedostonsiirto-ohjelma SSH:n päällä. OpenSSH-projektin luoma sftp-työkalu on toiminnoiltaan varsin alkukantainen, jonka vuoksi lftp:n sftp-tuki on varteenotettava vaihtoehto sftp:lle. Komennon syntaksi:

```
# sftp [user@]host
```

sftp on käytöltään hyvin samanlainen kuin UNIXien ftp-käsky. sftp:n ajoaikainen "help" käsky antaa kattavan ohjeen eri toiminnoista.

Esimerkki: kopioidaan yksi tiedosto etäkoneelle:

```
[user@foo ~]# sftp bar
Connecting to bar...
Password:
sftp> pwd
Remote working directory: /home/user
sftp> ls
dir1      dir2
sftp> cd dir1
```

```
sftp> ls
file1
sftp> mput file2
Uploading file2 to /home/user/dir1/file2
file2                                 100% 2180     2.1KB/s   00:00
sftp> ls
file1     file2
sftp>
```

## 3.6 Suora viestintä: IRC

Suurta suosiota UNIX -piireissä saavuttanut IRC -keskustelujärjestelmä antaa käyttäjille mahdollisuuden reaaliaikaiseen keskusteluun. Tämän rinnalle on kehitetty DCC -protokolla, joka mahdollistaa myös tiedostojensiirron IRC -asiakasohjelmien välillä.

IRC -protokollaa varten on tehty monia erilaisia asiakasohjelmia, joiden eroja ei kuitenkaan käsitellä tässä yhteydessä. Esimerkin vuoksi mainitaan kaksi asiakasohjelmaa:

- Irssi (*http:/irssi.org*) (komentorivipohjainen)
- XChat (*http://xchat.org*) (graafinen)

Näissä ohjelmissa DCC -lähettäminen onnistuu "/dcc send" komennolla. Seuraava esimerkki lähettää Irssi -ohjelman avulla "anon" käyttäjälle kaikki "/tmp/*.txt" tiedostot:

```
/dcc send anon /tmp/*.txt
```

Lähetyksen onnistuminen vaatii, että lähettäjään päin voidaan avata

TCP-yhteyksiä, ja siksi lähettäjän on syytä tarkistaa mahdolliset palomuuriasetuksensa.

## 3.7 Suora viestintä: netcat, socat

Toisin kuin muissa esitetyissä työkaluissa, *netcat* ja *socat* eivät vaadi erityisiä palvelimia tiedonsiirtoon. netcat ei tarjoa minkäänlaista sovellustason protokollaa, vaan avaa pelkän TCP-yhteyden (2-suuntainen) kahden koneen välille. Yksinkertaisuudessaan netcat tekee kahta asiaa samanaikaisesti:

- lukee tietoa stdinistä ja kirjoittaa sen TCP-yhteyteen
- lukee tietoa TCP-yhteydestä ja kirjoittaa sen stdouttiin

Tämän vuoksi netcatia voi soveltaa moneen eri tarkoitukseen:

- TCP-porttien kolkutteluun
- TCP-porttien välittämiseen
- Tiedostojen lähettämiseen

socat on netcatia monipuolisempi työkalu, joka tekee tekee samat asiat, mutta socatin käyttöön ei keskitytä tässä yhteydessä.

### 3.7.1 Esimerkki: keskustelu kahden koneen välillä

Luodaan keskusteluyhteys kahden koneen ("foo" ja "bar") välille. Kone "bar" kuuntelee portissa 1234 TCP-yhteyttä ja kone "foo" ottaa siihen yhteyden.

```
                                        [user@bar]# nc -l -p 1234
[user@foo]# nc bar 1234
Suora yhteys.
                     -viesti->
                                        Suora yhteys.
                                        Kivaa.
                     <-viesti-
Kivaa.
<CTRL-C>
[user@foo]#                             [user@bar]#
```

### 3.7.2 Esimerkki: tiedonsiirto kahden koneen välillä

Siirretään tiedostoja koneelta “foo” koneelle “bar” käyttäen apuna *tar* -komentoa. Kone “bar” kuuntelee porttia 1234 ja purkaa sinne saapuvan tar-virran.

```
                                [user@bar]# nc -l -p 1234 |tar xv
[user@foo]# tar cv dir1 |nc -q0 bar 1234
dir1/
dir1/tiedosto
                -tar-virta->
                                dir1/
                                dir1/tiedosto
[user@foo]#                     [user@bar]#
```

“-q0” vipu tarkoittaa, että yhteys suljetaan sen jälkeen kun stdinistä loppuu syöte ja viimeinenkin syöte on kirjoitettu TCP-yhteyteen.

Käyttäjä voi tarvitessaan käyttää lisätyökalua, joka mittaa siirron nopeutta komentoriviputkessa (*http://freshmeat.net* → etsi työkaluja *pmr*, *pipemeter*, *pv*, ...):

```
# tar cv /hakemisto |pmr |nc kone portti
/hakemisto/
/hakemisto/tiedosto
bandwidth: 17.55 MiB/s  total: 35.12 MiB (36831232 bytes)
bandwidth: 20.20 MiB/s  total: 75.57 MiB (79245312 bytes)
bandwidth: 20.18 MiB/s  total: 116.05 MiB (121690112 bytes)
...
```

### 3.7.3 Esimerkki: tiedonsiirto kolmannen osapuolen välityksellä

Tässä esimerkissä siirretään tietoa kahden osapuolen välillä käyttäen kolmatta osapuolta välittäjänä. Kolmas osapuoli on tarpeellinen mikäli kaksi muuta ovat kummatkin palomuurien takana. Oletetaan että halutaan siirtää tietoa koneelta A koneelle B hyödyntäen kolmatta konetta C. Koneelle C laitetaan kaksi netcatia, joista ensimmäinen kuuntelee porttia 1234 johon kone A ottaa yhteyden, ja toinen kuuntelee porttia 2345 johon kone B ottaa yhteyden. Porttiin 1234 tuleva yhteys putkitetaan porttiin 2345 tulevaan yhteyteen:

```
1. kone C: nc -l -p 1234 |nc -l -p 2345
2. kone A: tar cv tiedostot* |nc koneC 1234
3. kone B: nc koneC 2345 |tar xv
```

## 3.8 sshfs (FUSE)

*sshfs* on *FUSEn* (*http://fuse.sourceforge.net*) päälle tehty tiedostojärjestelmä, joka mahdollistaa etäkoneilla olevien tiedostojen suoran käyttämisen paikallisella koneella.

FUSE tarjoaa mahdollisuuden käyttää mielivaltaisia tiedostojärjestelmiä ilman pääkäyttäjän oikeuksia. FUSE sisällytettiin virallisesti Linux ytimeen 2.6.14. FUSEn saa asennettua myös aikaisempiin ytimiin, mukaan lukien 2.4.x ytimet.

sshfs:ää käyttäen, etäkoneella sijaitsevat tiedostot näkyvät paikallisen koneen sovellukselle aivan kuten ne sijaitsisivat samalla koneella. sshfs luo käyttäjän tiedostoavaruuteen yksityisen liitospisteen (engl. *mount point*). sshfs vaatii toimiakseen (nimestä arvatenkin) SSH -tunnuksen etäkoneelle. Etäkoneella tarvitsee olla sftp -tuki.

Lisäominaisuutena sshfs:n tarjoama näkymä etäkoneelle on käyttäjälle yksityinen, jonka seurauksena esimerkiksi pääkäyttäjän ajama varmuuskopiointiohjelmisto ei pääse käyttäjän tiedostoihin käsiksi.

Suurin etu sshfs:n käytössä on sen yksinkertaisuus. sshfs ei tarvitse minkäänlaista säätöjen asetusta toimiakseen tavallisesti (muuta kuin

suoraviivaisen asennuksen).

sshfs:n syntaksi:

```
sshfs [user@]host:[hakemisto] paikallinenhakemisto
```

### 3.8.1 Kotihakemiston käyttäminen toiselta koneelta

Tässä esimerkissä käyttäjä haluaa tuoda etäkoneelta näkyviin oman kotihakemistonsa. Syntaksi on sama kuin ssh:n tapauksessa.

```
# mkdir paikallinen
# sshfs user@host: paikallinen/
# ls paikallinen/
<etäkoneen kotihakemiston tiedostot>
```

Etäyhteyden saa suljettua “fusermount” komennolla:

```
# fusermount -u paikallinen
```

# 4 Muut työkalut

## 4.1 Eheydentarkistaminen

### 4.1.1 Tarkistussummat (md5sum, sha1sum)

Yleinen ongelma varmuuskopioinnissa on tiedostojen eheyden tarkistaminen lähde tai kohdepäässä. Eräs ratkaisu tähän on käyttää ohjelmia kuten *md5sum* tai *sha1sum*. md5sum laskee mielivaltaisesta tiedostosta 128-bittisen tarkistussumman, jota voi verrata oikeaan tiedostoon. Mikäli tarkistussumma on eri, on tiedosto varmasti rikki. Mikäli tarkistussumma on sama, on tiedosto hyvin todennäköisesti (mutta ei matemaattisen varmasti) ehjä.

Käyttäjä voi luoda tiedostoistaan tarkistussummat seuraavasti:

```
# find /polku/ -type f -print0 |xargs -0 md5sum >> md5.txt
# cat md5.txt
60b725f10c9c85c70d97880dfe8191b3  ./dir1/file1
3b5d5c3712955042212316173ccf37be  ./dir1/file2
```

### 4.1.2 Tiedostojen suora vertaaminen (cmp)

Kahden tiedoston sisältöä voi verrata *cmp* käskyllä.

```
# cmp foo bar && echo samat
samat
```

Tarkistussummat voi tarkistaa seuraavasti:

```
# md5sum -c md5.txt
./dir1/file1: OK
./dir1/file2: OK
```

## 4.2 GNU Privacy Guard - gpg

GNU Privacy Guard (komentorivillä gpg) on työkalu tiedostojen vahvaan salaamiseen. gpg osaa sekä julkisen että ei-julkisen salaamisen. Esimerkiksi varmuuskopioinnin yhteydessä voi olla hyödyllistä käyttää salausta.

Tiedostojen salaaminen ei-julkisesti tapahtuu esimerkiksi seuraavasti:

```
# gpg -c tiedosto
Enter passphrase: <käyttäjä syöttää salasanan>
# ls
tiedosto   tiedosto.gpg
```

Tiedosto "tiedosto.gpg" on salattu versio alkuperäisestä. Salatun tiedoston voi purkaa seuraavasti:

```
# gpg tiedosto.gpg > tiedosto
gpg: CAST5 encrypted data
Enter passphrase: <käyttäjä syöttää salasanan>
```

## 4.3 screen

Komentorivityöskentely voi vaatia useita samanaikaisia komentoriviyhteyksiä ollakseen mahdollista tai käytännöllistä. Tätä varten on kehitetty erityinen työkalu nimeltä *screen*, jolla pystyy ajamaan rinnakkain useita komentorivitulkkeja etäkoneella niin että komentorivitulkkien olemassaolo ei riipu siitä onko yhteys etäkoneelle auki vai ei.

Jos käyttäjä esimerkiksi haluaa ladata etäkoneella tiedostoa, jonka lataaminen kestää pitkään, voi hän ajaa latausohjelmaa screenin sisällä. Käyttäjä voi sulkea komentoriviyhteyden etäkoneelle, ja latausohjelma jää ajamaan screenin sisälle. Myöhemmin käyttäjä voi kirjautua takaisin etäkoneelle katsomaan mitenkä asiat etenevät.

Ohjeita screenin käyttöön saa esimerkiksi osoitteesta:
*http://zakalwe.fi/~shd/ohjeet/screen-ohje.txt*

## 4.4 split

Joskus käyttäjällä on tarve pilkkoa iso tiedosto pienempiin palasiin. Tähän soveltuu mainiosta *split* työkalu, joka löytyy useimmista UNIXeista vakiona.

Tässä esimerkissä pilkotaan 115 MB tiedosto 50 MB paloihin:

```
# ls -la foo
-rw-------   1 user users 115000000 Apr 19 01:39 foo
# split -b 50000000 foo foo.
# ls -la foo.*
-rw-------  1 user users 50000000 Apr 19 01:39 foo.aa
-rw-------  1 user users 50000000 Apr 19 01:39 foo.ab
-rw-------  1 user users 15000000 Apr 19 01:39 foo.ac
```

Tämän jälkeen tiedostojen yhdistäminen takaisin alkuperäiseksi voidaan tehdä esimerkiksi *cat* -komennolla:

```
# cat foo.* > bar
```