Unix kertausta

Shell = komentotulkki

Shell on
- merkkipohjainen komentotulkki
  = komentojen syöttö ja ohjelmien käynnistäminen
- komento(ohjelmointi)kieli
- 'sovellus'ohjelma, jonka nimi on
  sh (Bourne shell), **bash** (Bourne again shell), ksh (Korn shell),
  csh (C-shell), tcsh (Tenex C-shell),
  zsh, ...
- graafisen liittymän ”komentoikkuna”

Tulkkien keskeiset komennot ovat samat, mutta erojakin löytyy runsaasti.

UNIX-ympäristössä isoilla ja pienillä kirjaimilla on merkitysero. Mm. kaikki komennot on kirjoitettava **pikkukirjaimin**.

Shell ilmaisee valmiutensa tulostamalla näytölle kehotteen $ (bourne, korn, bash), % (csh) tai > (tcsh). Kehotteen voi muuttaa mieleisekseen asettamalla muuttujan **PS1**

Oletusshell on kirjattu käyttäjätietojen yhteyteen tiedostoon /etc/passwd. Sitä voi vaihtaa komennolla

```
$ chsh kjätunnus polkunimi
```

Komento chsh kelpuuttaa uudeksi polkunimeksi vain tiedostoon /etc/shells merkityt komentotulkkien polut.

Huom: komentotulkit käyttävät erinimisiä alustustiedostoja, joten ainakin

aluksi saattaa olla viisasta pitäytyä ylläpitäjien asetuksissa!
Uuden komentotulkin voi käynnistää myös komentoikkunassa antamalla kehotteeseen sen nimen: tsch, bash, jne.

Mikä komentotulkki on käytössä?

```
$ echo $SHELL
$ ps
```

Ensimmäisen istunnon yhteydessä on syytä vaihtaa järjestelmän ylläpitäjien antama salasana

```
$ passwd
Changing password for hakkinen
Old password:                    (kirjoita vanha salasana)
New password:                    (tähän uusi salasana)
Re-enter new password:
$
```

HUOM: Stadiassa salasana vaihdetaan web-selaajalla sivuston *vo.stadia.fi* kautta.

Istunto päätetään komennolla

```
$ logout
```

Jos tulee ilmoitus *'not a login shell'*, on annettava ensin muutama **exit**-komento.

Komentorivin syntaksi

Komentoja voi antaa interaktiivisesti tai niitä voi kirjoittaa isomman joukon etukäteen komentotiedostoksi. Komentorivi muodostuu komentosanasta, sen mahdollisista valitsimista ja argumenteista.

```
komento [-valitsimet] [arg2] ... [argn]
```

Argumentti on tdstonimi, avainsana tms., johon komento kohdistuu

```
$ whoami                    (näytä käyttäjätunnus)
$ hostname                  (työkoneen nimi)
$ man man                   (opastusta komennosta man)
$ cp file1 file1.bak        (tiedoston kopiointi)
```

Valitsin alkaa tavuviivalla **-** . Niillä modifioidaan tai tarkennetaan komentoa

```
$ man -h ls                 (vain lyhyt kuvaus)
$ ls -C                     (hakemistolistaus sarakemuodossa)
$ ls -la                    (pitkä esitysmuoto, myös .-alkuiset)
```

Useampia yksikirjaimisia valitsimia voi kirjoittaa yhteen. Monet komentotulkit kelpuuttavat valitsimet myös ilman tavuviivaa. Tarvittaessa kaksi peräkkäsitä tavuviivaa -- ilmoittaa, että kyseessä on viimeinen valitsin ja rivin loppuosa tulee tulkita normaaleiksi argumenteiksi.

Tavallisesti UNIX-ohjelmat lopetetaan komennolla **q**, **Q**, **bye** tai **quit**. Suorituksen voi usein keskeyttää myös väkisin painamalla `Ctrl-c` (merkitään usein myös ^C). Komentorivin näppäimistöltä kaipaamien syötteiden loppumerkkinä on `Ctrl-d`

Bash osaa täydentää komentoriville komennon, tiedostonimen ja ~käyttäjätunnuksen, kun sen yksikäsitteinen alku annetaan ja painetaan `tab`-näppäintä. Nuolinäppäimillä voi selailla aiemmin annettuja komentorivejä. ^A vie kohdistimen komentorivin alkuun ja ^E loppuun.

Komentorivillä voi antaa useita komentoja käyttämällä erottimena puolipistettä `;`

```
$ echo hakemistossa;ls -r   (kaiutus, käänteinen aakkostus)
```

Komentojen suorituksen voi ehdollistaa

```
$ cat tdsto || echo ei onnistu
$ cat tdsto && echo onnistui
```

(tiedoston sisällön listaus)
(|| = oik.puoleinen komento suoritetaan, jos listaus epäonnistuu, && päinvastoin)

Komentoja voi ryhmitellä


```
$ (cd ..;ls);ls             (isähakemiston ja työhakemiston hakemistolistaus)
$ (cd ..;pwd);pwd
```

Komennon voi suorittaa myös taustaprosessina

```
$ cc ohjelma.c &            (käännös)
$ (sleep 300; echo kaffepaussi) &
```

Shell palauttaa työn / prosessin numeron, ja ne saa myös komennoilla

```
$ ps                        (process status)
$ jobs
```

Taustaprosessi loppuu istunnon päättyessä, ellei sitä ole erikseen kielletty (nohup).

Prosessin / työn suorituksen voi lopettaa komennolla kill

```
$ man kill &                (käynnistä taustaprosessiksi)
[1] 466
$ kill 466                  (jos ei tehoa, kokeile kill -KILL 466)
$ kill %1
```

Opastus

UNIXiin voi tutustua ohjelmilla (eivät löydy kaikista koneista)

learn (sysV)
help *[komento]* (bashin sisäiset komennot)
info *[komento]*
apropos *[avainsana]* (BSD, Linux: sama kuin man -k avainsana)
whatis *[avainsana]* (BSD, Linux, sma kuin man –f avainsana)
man *[-s luku | -h] komento*

Tiivistetyn selityksen peruskomennoista saa komentamalla

```
$ man intro
```

Yksittäisestä komennosta saa parhaiten tietoa **man**-komennolla

```
$ man man          (opastusta man-komennosta)
$ man -h man       (vain lyhyt kuvaus)
```

Monet komennot antavat virheellisesti syötettynä ilmoituksen, josta käy ilmi syntaksi ja valitsimet.

Manuaalisivut on ryhmitelty aihepiirin mukaan

```
1. user commands
2. system calls
3. subroutines
4. devices
5. file formats
6. games
7. miscellaneous
8. system administration
l. local
n. new
```

Komentotulkin komentoja on selvitetty luvuissa 1, 8, l ja n.

Lukuihin liittyy lyhyt esittelysivu, jonka saa avainsanalla intro.

```
$ whatis intro      (lyhyt yhteenveto kaikista intro sivuista)
$ man 5 intro       (luvun 5 intro-sivu)
```


Eräillä hakusanoilla löytyy tietoja useammastakin kohdasta. Viittausten yhteyteen on usein tapana merkitä myös manuaaliluvun numero, esimerkiksi passwd(1) tai passwd(5).

```
$ man 1 passwd
$ man 5 passwd
```

Ikkunointiympäristössä löytyy mukavampiakin liittymiä manuaalisivuihin, ks. esim.

www.die.net/doc/linux/man/

Syötön ja tulostuksen uudelleenohjaus

Kuhunkin prosessiin liittyy automaattisesti kolme tietovirtaa: **stdin**, **stdout** ja **stderr** (i.a. tiedostot numero 0, 1 ja 2). Syöttö oletusarvoisesti näppäimistöltä (stdin) ja tulostus oletusarvoisesti näytölle (stdout ja stderr).

Tulostuksen voi ohjata tiedostoon > -merkillä ja tiedostosta syöttö onnistuu < **-** merkillä. Komennoissa, joiden syntaksi vaatii tdstonimen viitataan stdin:iin merkillä **-**

```
$ cat > zappy                          (syötteet näppäimistöltä !)
Tämä menee tiedostoon zappy...
^D                                     (Ctrl-d:n painallus lopettaa syötön)
```

Tdston, johon tulostus ohjataan, vanha sisältö katoaa. Jos halutaan lisätä tdston loppuun, käytetään merkkiparia >>

Virheilmoitukset ohjautuvat oletusarvoisesti näytölle:

```
$ cat dippu zappy > dappu
dippu: No such file or directory
```

Ne voi ohjata omaan tdstoon merkinnällä 2>

```
$ cat dippu zappy > dappu 2> puppu
$ cat dappu
Tämä menee tiedostoon zappy...
$ cat puppu
dippu: No such file or directory
```

Merkinnällä **&>** ja **&>>** ohjataan sekä stdout että stderr samaan tdstoon.

```
$ cat dippu zappy &> dappu
```

Jos tulostukset eivät kiinnosta, ne voi ohjata tdstoon **/dev/null**.

```
$ time wc /etc/dict/words > /dev/null
                    (paljonko kuluu aikaa wc-komennossa)
```

Ohjelman tulosteet voi ohjata toisen ohjelman syötteeksi **putkella** (pipe)

```
$ cat tdsto | wc -w                    (sanojen lkm)
$ who | wc -l ; date
```


Putket ja UNIX-filosofia ”*small is beautiful*” ja ”*no news is good news*” on oivallinen keino koota yksinkertaisista komennoista tehokkaampia kokonaisuuksia.

Myös putken keskeltä (”T-kappaleesta”) voi saada tulosteen tiedostoon

```
$ who | sort | tee sorted.who| less
                    (näytä koneen tämänhetkiset käyttäjät)
                    (ota datasta kopio myös tiedostoon)
```

Asetus **set -o noclobber** estää uudelleenohjauksen vanhojen tdstojen päälle. Pakotettavissa `>|`, `2>|`, `&>|`, `>>|`, `2>>|`, `&>>|`

Tiedostojärjestelmä

Tiedosto luodaan joko editorilla, uudelleenohjauksella tai systeemikutsulla ohjelmasta käsin.

Tiedostot

Tdsto on vain jono tavuja. Tietueita tai jaksoja ei ole olemassa ja tiedon organisointi (rakenne) tehdään aina ohjelmallisesti, esim. erillisessä tiedonhallintajärjestelmässä.

Minkäänlaista tdston tyypin käsitettä ei ole olemassa. Esim. komento `file` päättelee tdston "tyypin" selaamalla tiedostoa.

Tdstonimen maksimipituus on 255 merkkiä. Siinä voi käyttää mitä tahansa merkkejä (myös välilyöntejä lainausmerkeissä). Vain kauttaviiva `/` on varattu hakemistopolun osien erottelemiseen.

Isoilla ja pienillä kirjaimilla on merkitysero.

Myös piste on vain merkki merkkien joukossa. Jos nimi alkaa pisteellä, se ei näy tavallisessa hakemistolistauksessa. Nekin saa näkyviin anatamalla komennon `ls –a`

Tekstitdstossa on näkyvien tavujen lisäksi erikoismerkkejä

`\b`	backspace	(\ poistaa seuraavan merkin norm. merkityksen)
`\t`	tab	
`\n`	newline	
`\r`	carriage return	
`\0`	null character	
`\f`	form feed	

Tdston loppumista ei ilmaista erikoismerkeillä, vaan UNIX ylläpitää tietoa tiedoston koosta ja, siitä kuinka paljon tiedostoa on luettu / lukematta.

Erikoistiedostot

Myös jokaiseen laitteeseen, linjaan ja jopa keskusmuistiin liittyy tdsto, jota lukemalla ja kirjoittamalla on toteutettu tdstojärjestelmän muut osat.

```
$ ls -l /dev
```

Esim. magneettinauhatulostus tapahtuu kirjoittamalla tdstoon `/dev/mt`. Kirjoitettu tieto menee edelleen laiteajurille, joka toimittaa sen laitteelle.

- tiedosto- ja laite-I/O ovat samanlaisia
- tiedosto- ja laitenimillä ei muoto eikä merkityseroa
- sama käyttöoikeusmekanismi

Omaan komentoikkunaan voi viitata merkinnällä `/dev/tty`

```
$ cp /dev/tty puppu
Tämä rivi menee tiedostoon puppu
^D                                   (ilmoita syötteiden loppumisesta)
```

Hakemistot

Toteuttavat kuvauksen NIMI --> FYYSINEN TIEDOSTO

Hakemisto on KJ:n ylläpitämä tiedosto, jossa on (tiedostonimi, indeksisolmu-numero) -pareja. Indeksisolmu (i-solmu) on erikseen taltiolla ja siinä on tiedoston hallinnolliset määreet. Hakemistotiedostoa ei voi käsitellä tavallisilla tdstokomennoilla, vaan hakemistojen käsittelyä varten on omat komentonsa.

Jokaisella käyttäjätunnuksella on oma kotihakemisto ja käyttäjä voi luoda alihakemistoja komennolla

mkdir *hakemisto [...hakemisto]*

==> hierarkkinen puurakenne, jossa solmut joko hakemistoja tai tiedostoja.

Hakemistoja voi poistaa komennolla

rmdir *hakemisto [...hakemisto]*

Tiedostoihin liittyvät toiminnot kohdistuvat oletusarvoisesti aina työhakemistoon. Aluksi työhakemisto = kotihakemisto, sitä voi vaihtaa komennolla

cd *[hakemisto]*

Pelkkä `cd` asettaa työhakemistoksi käyttäjän kotihakemiston.

Työhakemiston polkunimen saa näytölle komennolla

pwd

Tiedoston (hakemiston) yksilöivä polkunimi muodostuu hakemistonimistä ja tiedostonimestä. Absoluuttinen polkunimi alkaa hakemistopuun juuresta ja on muotoa

/hakemisto / *hakemisto* / ... / *tiedosto*

Suhteellinen polkunimi alkaa työhakemistosta ja on muotoa

hakemisto / *hakemisto* / *tiedosto*

Jokainen hakemisto sisältää aina hakemistot **.** ja **..** (miksi?)

- **.** on lyhennysmerkintä ko. hakemiston omalle nimelle ja
- **..** on lyhenne ko. hakemiston isähakemistolle

Omaan kotihakemistoon voi viitata notaatiolla ~ ja toisen käyttäjän hakemistoon merkinnällä `~kjätunnus`

esim. jos työhakemisto on `/home/lassara/temp`,
voi tiedostoon `/home/lassara/bin/x`
viitata notaatiolla `../bin/x` tai `~/bin/x`

/	juurihakemisto (root), suoritettava ydin
/boot	staattiset boottaamiseen tarvittavat tdstot
/bin	binääriset ohjelmaversiot (binaries)
/usr/bin	yleisemmät ylempänä
/usr/local/bin	'paikalliset' asennukset (vaihtelee)
/dev	laitetiedostot (devices)
/etc	hallinnolliset, ylläpitoon liittyvät ohjelmat
/usr/etc	ja tiedostot (asetukset) (et cetera, "system closet")
/home	käyttäjien kotihakemistojen hakemisto
/usr/home	
/lib	kääntäjän osia, arkistokirjasto (libraries)
/usr/lib	
/mnt	tiedostojärjestelmien liittäminen yhdeksi hakemistopuuksi (mount)
/opt	paikallisesti asennetut sovellukset (vaihtelee) (optional)
/opt/<pakkaus>	
/tmp	tilapäisille / tilatarpeeltaan voimakkaasti vaihteleville tiedostoille (kaikilla oikeudet) (temporaries)
/usr	toissijainen hierarkia
/usr/include	C-kielen kirjastorutiineja
/var	sovellusten muuttuville aputdstoille (variable data)

Indeksisolmu ja tiedostolinkit

Tiedoston hallinnollinen tieto on talletettu **indeksisolmuun** (i-solmu), joka sisältää mm. (ks. man 2 stat)

- omistajan tunnus, ryhmätunnus
- käyttöoikeusbitit (ns. rwx-bitit)
- koko tavuina
- 'tyyppi', esim. -: tavallinen tdsto, d: hakemisto
- missä päin levyä tiedosto sijaitsee, lohkohakemisto
    - 12 suoraa lohkonumeroa,
    - 1 epäsuora
    - 1 tuplasti epäsuora
    - 1 triplasti epäsuora
- linkkien lukumäärä
- milloin tiedostoa viimeeksi päivitettiin `ls -lt`
- milloin tiedostoon viimeeksi viitattiin `ls -lu`
- milloin indeksisolmua viimeeksi muutettiin `ls -lc`

Kullakin indeksisolmulla on yksikäsitteinen numero.

Kun tiedosto luodaan, järjestelmä luo indeksisolmun ja lisää hakemistotiedostoon parin: (tiedostonimi, i-solmunumero).

Kun tiedosto hävitetään, KJ nollaa i-solmunumeron hakemistosta ja vapauttaa solmulle varatun tilan.

```
$ ls -i                (listaa myös indeksisolmunumerot)
15768 puppu
$ date > x
$ ls -i
15768 puppu
15852 x
```

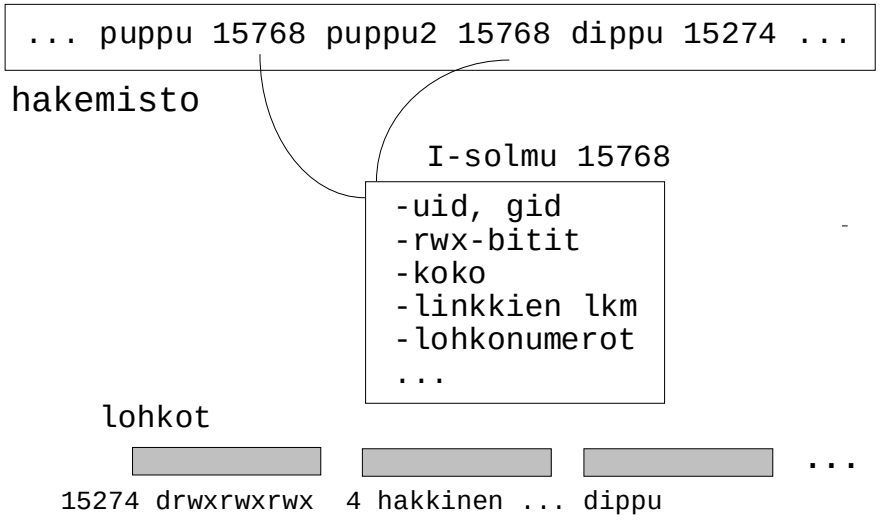

Tiedostojen linkittäminen komennolla

ln *[-fs] lähdetdsto kohdetdsto* (-f force, -s symbolic)

sallii tiedostojen uudelleennimeämisen kopioimatta niitä, ts. useampi nimi johtaa samaan fyysiseen tiedostoon. Jos ei käytetä valitsinta -s sama indeksisolmunumero voi liittyä useisiin tiedostonimiin yhtä aikaa (ns. kova linkki).

Linkin voi purkaa poistamalla tiedoston.

```
$ ln puppu puppu2                    (luo 'kova' linkki)
$ ls -li
15768 -rw-rw-rw-  2 hakkinen ... puppu
15768 -rw-rw-rw-  2 hakkinen ... puppu2
15274 drwxrwxrwx  4 hakkinen ... dippu
$ rm puppu2 ; ls -li
15768 -rw-rw-rw-  1 hakkinen ... puppu
```



Kovan linkin sijasta kannattaa käyttää symbolisia linkkejä (valitsin `-s`). Tällöin järjestelmä luo 'välitiedoston', jolla on oma indeksisolmu ja jonka sisältönä on kohteena olevan tiedoston polkunimi. Kyseessä on siis epäsuora viittaus.

```
$ ln -s puppu puppu2              (luo symbolinen linkki)
$ ls -li
15768 -rw-rw-rw- 1 hakkinen ... puppu
15800 -rw-rw-rw- 1 hakkinen ... puppu2->puppu
15274 drwxrwxrwx 4 hakkinen ... dippu
```

Kun käytetään symbolista linkkiä, voi tiedoston omistaja paremmin hallita tiedostoaan. Kun omistaja poistaa tiedoston, voi KJ todella poistaa tiedoston ja vapauttaa myös sen indeksisolmun.

Jokerimerkit

Jokerimerkkien avulla voi viitata useisiin tdstonimiin kerralla.

1) Kysymysmerkki **?** korvaa minkä tahansa yhden merkin.

2) Tähtimerkki ***** korvaa minkä tahansa merkkijonon.

3) Hakasulut **[...]** ilmaisee vaihtoehtoiset yksittäiset merkit.

   ```
   $ lpr osa[0-9] osa[12][0-9] osa3[0-5]
   ```

   Voi antaa myös muodossa `[!abc]`, jolloin kelpuuttaa mitkä tahansa muut kuin luettelut merkit

4) Aaltosuluilla **{...}** merkitään pitempi vaihteleva osa.

   ```
   $ cat ../{memo,pr*}.c
   ```

5) Tilde **~** on viite käyttäjän omaan kotihakemistoon.
   ~kjätunnus on viite toisen käyttäjän kotihakemistoon.

Ennen käskyn tulkintaa komentotulkki korvaa jokerimerkit vastaavilla hakemistosta löytyvillä tiedostonimillä.

(Tämän vuoksi esimerkiksi `cp *.c *.bak` ei toimi kuten Windowsin Command Promptissa)

Jos tdstonimi alkaa pisteellä, on se aina merkittävä paikalleen.

Jokerimerkkien merkityksen voi poistaa lisäämällä kenoviivan `\` ko. merkin eteen tai käyttämällä yksinkertaisia lainausmerkkejä ´...´.

Asetus **set -o noglob** poistaa merkkien *. ?, [,] ja ~ jokerimerkityksen.

Hakemistokomentoja

(ks. tarkennukset man-sivuilta)

ls tulosta hakemiston sisältö (list contents of directory)

pwd tulosta työhakemiston polkunimi (print working directory)

mkdir luo hakemisto (make directory)

rmdir poista *tyhjä* hakemisto (remove directory)
ei-tyhjän hston poisto: `rm -rf hstonimi`

cd vaihda työhakemistoa (change directory)

pushd hakemistonimipinon käsittelyyn (push directory)
popd (pop directory)
dirs (print directory stack)

Tiedostokomentoja

(ks. tarkennukset man-sivuilta)

cat listaa tdston sisältö / tdstojen yhdistely (concatenate and print)

head listaa tiedoston alkua
tail listaa tiedoston loppua

more tulosta tiedoston sisältö näytöllinen kerrallaan
less

cp tee kopio (copy)
mv uudelleennimeä / siirrä toiseen hakemistoon (move)
ln muodosta vaihtoehtoinen tiedostonimi (link)
rm poista tiedosto (remove)

tee kopioi stdin-virta sekä stdoutiin että tiedostoon

wc tiedoston rivien, sanojen ja merkkien lkm (word count)

file luokittelu: minkätyyppistä tietoa tiedostossa

du tietoja käytetystä levytilasta (disk usage)
df tietoja vapaasta levytilasta (disk free)

grep etsintä tekstitiedostosta (global regular expression print)

find etsi ehdot täyttäviä tiedostoja, kohdista joku toiminto löydettyihin
(ei edes tulosta mitään, ellei pyydetä)

diff kahden tiedoston erot (differential file and directory comparator)
comm yhteiset rivit (select/reject lines common to sorted files)

join eri tiedostojen rivien yhdistely (relational database operator)

sort järjestäminen, järjestettyjen tietojen lomitus
uniq tuplarivien poisto

touch muuta tiedoston 'muutettu'-aikaleimaa

Katso yksityiskohdat ja käyttötapa manuaalisivuilta.

Käyttätunnukset ja ryhmät

Järjestelmä erottaa käyttäjät ja ryhmät toisistaan käyttäjä- ja ryhmänumeroilla (uid, user id ja gid, group id). Käyttäjä- ja ensisijainen ryhmä on kirjattu tdstoon `/etc/passwd` tai /var/yp/passwd tai nimipalvelijan NIS-tietokantaan

kjätunnus:salasana:uid:gid:info:hakemisto:ohjelma

Käyttäjä voi olla jäsenenä useissa ryhmissä, mutta aktiivisena voi olla vain yhdessä ryhmässä kerrallaan. Käyttäjä voi tarkistaa mihin ryhmiin kuuluu komentamalla

groups

Käytettävä gid (engl. effective gid, egid) on aluksi salasanatiedoston gid.

Aktiivista ryhmätunnusta voi vaihtaa komennolla

newgrp *[-] [ryhmätunnus]*

Pelkkä newgrp näyttää käytössä olevan ryhmän numeron, ja newgrp - palauttaa ryhmänumeroksi salasanatiedoston ryhmätunnuksen.

Vain etuoikeutettu käyttäjä (tunnus root) voi luoda uusia ryhmiä ja liittää niihin jäseniä. Toissijaiset ryhmätunnukset ja ryhmän jäsenet on kirjattu tdstoon /etc/group tai /var/yp/group (tai NIS-tietokantaan)

ryhmätunnus:salasana:gid:jäsenten kjätunnukset

Työskenneltäessä luotavan tiedoston gid määräytyy aktivoidun ryhmätunnuksen mukaan.

Käyttöoikeudet (eli tiedostojen suojaus)

Tiedostojen ja hakemistojen käyttöoikeudet perustuvat
- käyttäjien luokitteluun
  - omistaja (u),
  - omistajan ryhmä (g) ja
  - muut (o)
- sekä kullekin luokalle erikseen annettaviin oikeuksiin
  - luku- (r),
  - kirjoitus- ja poisto-oikeus (w) ja
  - suoritusoikeuksiin (x).

Oikeuden x sijasta voi olla erikoisoikeus
- suid (s) oikeus asettaa uid,
- sgid (S) oikeus asettaa gid
- sticky (t) ohjelmakoodia käytetään paljon, pidä keskusmuistissa Vain superuserilla on oikeus asettaa sticky-bitti.

Hakemistolle
- lukeminen = oikeus listata mitä tdstoja hstossa sisältää
- kirjoittaminen = oikeus luoda tdsto hstoon / oikeus poistaa tdsto hstosta
- suorittaminen = oikeus käyttää nimeä hakupolussa (”läpikulkuoikeus”)

Tiedoston sisällön voi katsoa, kun on lukuoikeus (r) itse tiedostoon ja läpikulkuoikeus (x) kaikkiin absoluuttisen polkunimen hakemistoihin. Esim:

```
drwxr-xr-x root    ...  /
drwxr-x--x root    ...  /home
drwx--x--x hakka   ...  /home/hakka
drwx--x--x hakka   ...  /home/hakka/public_html/
-rwxr--r-- hakka   ...  /home/hakka/public_html/index.html
```

Tiedoston käyttöoikeudet voi tarkistaa komentamalla esim.

```
$ ls -lg puppu
-rw-r--r-- 1 hakkinen grpa 512 Sep ...puppu
|  |  |  |  |    |      |    |
1  2  3  4  5    6      7    8
```

1. - = tiedosto, d = hakemisto, b = lohkolaite, c = merkkilaite, s = socket, l = symbolinen linkki, p = putki
2. omistajan rwx-bitit (yllä rw-)
3. ryhmän rwx-bitit (yllä r--)
4. muiden rwx-bitit (yllä r--)
5. linkkien lukumäärä
6. omistaja
7. omistajan ryhmä (optio -g)
8. koko tavuina

UNIX tarkistaa oikeudet seuraavasti

jos käyttäjän uid = tiedoston uid
niin tarkista oikeudet omistajan rwx-biteistä
muuten
  jos käyttäjän gid = tiedoston gid
  niin tarkista oikeudet ryhmän rwx-biteistä
  muuten
    tarkista oikeudet muiden rwx-biteistä

HUOM: jos käyttäjä kuuluu omistajan kanssa samaan ryhmään, ei tarkista kohdasta muiden rwx-bitit!

Välikysymys: Miten käyttäjä voi muuttaa salasanatiedostossa olevaa salasanaansa? Tiedoston /etc/passwd omistaja on root, eikä siihen voi antaa muille kirjoitusoikeutta. (miksi ei?)

Ratkaisu: oikeuksien väliaikainen laajentaminen

```
$ ls -l /etc/passwd                          (salasanatdsto)
-rw-r--r-- 1 root ...
$ ls -l /usr/bin/passwd                      (ohjelma, jolla salasana vaihdetaan)
-rws--x--x 1 root  ...
```

Omistajan suoritusoikeuden (x) tilalla on oikeus s (suid), jolla komennon passwd suorittaja saa komennon suoritusajaksi omistajan oikeudet.

Käyttöoikeuksien muuttaminen

Tiedoston omistaja (ja root) voi muuttaa käyttöoikeuksia. Käyttöoikeuksia muutetaan komennolla

chmod *[-R] [ugoa] [+-=] [rwxst] tiedostot* (change access mode)
chmod *[-R] bittimaski tiedostot* (-R recursive)

Käyttöoikeuksia voi siis muuttaa kahdella tavalla:

1) ilmaistaan muutokset tai uusi arvo kirjaimin

```
$ chmod g+w,o-r,u+x tdsto
$ chmod a+rx tdsto                    (a sama kuin ugo)
$ chmod +rwx tdsto                    (sama kuin a)
$ chmod go=rx tdsto
$ chmod go= tdsto
$ chmod u+rwx *                       (kaikki hakemiston tiedostot)
$ chmod o-r .                         (itse hakemisto)
```

2) ilmaistaan uudet käyttöoikeudet kolmen kokonaisluvun avulla. Ensimmäinen bittimaski kuvaa omistajan rwx-bitit, seuraava ryhmän rwx-bitit ja viimeinen muiden rwx-bitit.

```
$ chmod 775 .

0 = ---    1 = --x 2 = -w- 3 = -wx
4 = r--    5 = r-x    6 = rw- 7 = rwx
```

Välikysymys: Miten annat oikeudet vain yhdelle tietylle käyttäjälle? Entä vain tietyille käyttäjille?

Muita käyttöoikeuksiin liittyviä komentoja

chown muuta tiedoston omistajaa
(sallittu vain omistajalle tai rootille)

chgrp muuta tiedostoon liittyvää ryhmätunnusta
(sallittu vain omistajalle ja rootille, omistajan on kuuluttava myös uuteen ryhmään)

newgrp vaihtaa käytössä olevaa ryhmtunnusta

umask - vaikuttaa luotavien tiedostojen käyttöoikeuksiin, määrää poistettavat oikeudet, esim. umask 066 poistaa ryhmältä ja muilta oikeudet rw
- pelkkä umask *[-S]* näyttää nykyisen arvon

Katso yksityiskohdat ja käyttötapa manuaalisivuilta.

Etsintä

---

grep *[optiot] merkkijono [tiedostot]* (Global Regular Expression Print)

Etsii annettua merkkijonoa tdstosta. Tulostaa rivit, joilla annettu merkkijono esiintyy.

```
$ who | grep hakkinen | wc -l
3
$ who | grep -c hakkinen
3
$ grep -i oskari puh
Olematon Oskari 4242    C464
$ grep Oskari nrot osoitteet
nrot: Olematon Oskari   4242    C464
osoitteet: Olematon Oskari       Nollakatu 0
```

-c löydettyjen rivien lkm (count)
-e etsittävä voi alkaa -:lla (expression)
-h ei tulosta tdstonimiä (header)
-i tulkitsee isot ja pienet kirjaimet samoiksi (ignore case)
-l vain tdstojen nimet, joista merkkijono löytyi (list)
-n rivinumero, jolta merkkijono löytyi (number)
-s vain exitin palauttama koodi (status)
-v rivit, joilla ko. merkkijono ei esiinny (reverse sense of test)

egrep Etsii säännöllisenä lausekkeena annettua merkkijonoa tdstosta. Täydellisempi kuin edellinen. Sama kuin `grep –E` (extended grep)

fgrep Etsii tdstossa annettuja merkkijonoja tdstosta (fixed grep). Nopea. Ei osaa säännöllisiä lausekkeita. Sama kuin `grep -F`

Etsittävän merkkijonon voi määritellä yksinkertaisena säännöllisenä lausekkeena (vastaavasti ohjelmissa perl, awk ja sed).

Säännöllisessä lausekkeessa käytettäviä merkintöjä:

.	mikä tahansa yksittäinen merkki
?	edeltävä merkki valinnainen (voi siis puuttua)
*	0 tai useampia edellisen merkin/lausekkeen esiintymiä
+	1 tai useampia edellisen merkin/lausekkeen esiintymiä
[...]	vaihtoehtoluetteloon kuuluva merkki
[^...]	vaihtoehtoluetteloon kuulumaton merkki
x | y	joko merkki/lauseke x tai y
-	osaväli
\	poistaa erikoismerkityksen
(...)	ryhmittely
^	täsmää rivin alkuun
$	täsmää rivin loppuun

Vaihtoehtoluettelossa (hakasulkut) em. merkit menettävät erityismerkityksensä, paitsi ^ ja -.

Säännöllinen lauseke vastaa pisintä mahdollista vastinetta mahdollisimman lähellä rivin alkua. Tyhjä lauseke vastaa viimeeksi käytettyä lauseketta.

```
$ grep'st.*ing' teksti          (* --> shell ei saa evaluoida)
$ grep '[aeiouy]' teksti
$ grep '[^a-zA-Z]' teksti
$ grep (ab|cd)e teksti
$ grep -i ^k teksti
$ grep 'end\.' ohj.pas
$ grep 'end\.$' ohj.pas
$ grep -i taulu\[5\] < ohj.pas
$ grep \(+\) teksti
```


find *[hakemistot] [boolen lauseke]*

Tdston etsintä. Etsii annetusta hakemistosta (ja alihakemistoista) annetut boolen ehdot täyttäviä tdstoja, ja kohdistaa niihin jonkun toimenpiteen. Ei edes tulosta nimiä, ellei erikseen pyydetä.

Boolen lausekkeen merkintöjä (esim.): (ks. man find)

-a tai välilyönti	AND	
-o	OR	
!	NOT	
( )	ryhmittely	*Huom: shell ei saa evaluoida näitä \(ja \)*

+n	enemmän kuin, suurempi kuin, vanhempi kuin
n	samansuuruinen kuin, sama
-n	vähemmän kuin, pienempi kuin, nuorempi kuin

-name tdstonimi		
-type [bcdfls]		block, char, directory, file, link, socket
-exec komento	\;	suorita komento, jos ehto tosi
-ok komento	' \;	kysy saako komennon suorittaa

Yksinkertaisten lainausmerkkien ja \:n avulla estetään shell-evaluointi.

```
$ find . -print
$ find . -name ’*.c’ -print          (-a merkitsemättä)
$ find ~ -atime +30 –name ’*.o’ –print

$ find ~ -name ’*.bak’ -ok rm ’{}’ \;
$ find . \( -name core -o -name junk \) \
          -print -exec rm ’{}’ \;
```

Jos komento-osassa esiintyy `'{}'`, niin se korvataan komentoa suoritettaessa ehdon täyttäneiden tdstojen nimillä.

whereis *[valitsimet] tdstonimi*

Etsii järjestelmässä määritellyiltä hakupoluilta ohjelman binäärikoodia, lähdekoodia ja manuaalisivua, tulostaa polkunimet

```
$ whereis emacs
emacs: /usr/bin/emacs /usr/libexec/emacs /usr/share/emacs
/usr/share/man/man1/emacs.1.gz
```

which *[valitsimet] tdstonimi*

Näyttää suoritettavan komennon koko hakupolun. Tästä tiedosta on joskus apua, kun on useita samannimisiä ohjelmia (-a näyttää kaikki).

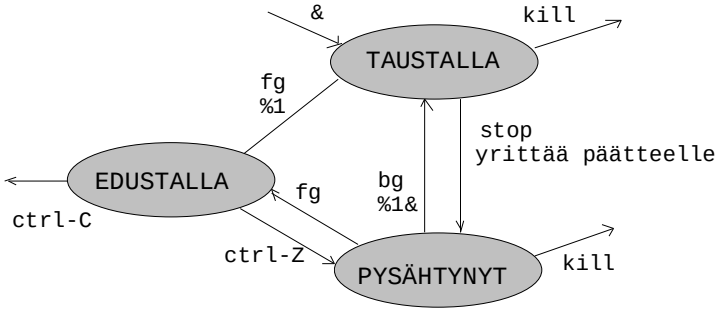
```
$ which emacs
/usr/bin/emacs
```

Jos komento on toteutettu shellin sisäisenä koodina, se suoritetaan aina ensisijaisesti, vasta sitten etsitään samannimisiä komento(tiedosto)ja hakupoluilta.

Prosessit

Komentotulkki luo (yleensä) uuden prosessin suorittamaan komentoa. Yleisimmät komennot on koodattu suoraan komentotulkkiin, niitä kutsutaan shellin sisäisiksi komennoiksi. Muut komennot (ohjelmat) sijaitsevat tavallisesti hakemistossa `/bin`, `/usr/bin` tai `/usr/local/bin`.

Työ eli suoritettava komento(sarja) voi olla
- suorituksessa edustalla (stdin ja stdout kuuluvat tälle)
- suorituksessa taustalla
- pysähtynyt
- päättynyt (zombie).



Esimerkki:

```
$ man man &
[1] 1277
$jobs
[1]+ Stopped  man man
$ fg
```

... painetaan Ctrl-z

```
$ kill –KILL %1
[1]+ Killed  man man
```

Jos käynnistettävänä on tavallinen ohjelma
- shell luo sille oman prosessin

Jos käynnistettävänä on komentotiedosto
- shell käynnistää uuden shell-prosessin suorittamaan
- vanhat shell-muuttujat eivät periydy uudelle
- komentotiedostossa shell-muuttujiin tehdyt muutokset eivät säily palattaessa

Jos käynnistettiin tausta-ajoksi (&)
- vanha shell heti valmis ottamaan uusia komentoja, eli ruudulle ilmestyy taas komentokehoite
- muuten vanha shell odottaa suorituksen päättymistä

Tapoja käynnistää prosessi

- järjestelmän komennot
- ohjelman nimellä (oltava x-oikeus)
- komentotiedoston nimellä (oltava x-oikeus, ja sijaittava hakupolulla)
- **sh** komento tai komentotiedosto
- **source** komentotiedosto tai **.** komentotiedosto
- **exec** komentotiedosto
- `komento`

bash luo taustaprosessin suorittamaan komentoa / komentotdstoa

source ei luo taustaprosessia, vaan argumenttina annettu komentotdsto suoritetaan nykyisessä ympäristössä.
Komentojen suorittamisen jälkeen jatkaa sama prosessi eteenpäin (myös jos source komentotdstossa).

. kuten source

exec kuten source, mutta komentotiedoston, jossa käsky oli, "jäljelle jääneitä" komentoja ei enää suoriteta.

`...` Huom. takakenohipsut, ei yksinkertainen lainausmerkki ´ eikä tavallinen lainausmerkki ". Shell evaluoi ``-merkkien välissä olevan komennon ja korvaa sen komennon tulosteilla. Esim.

```
$ echo "tunnuksesi on `who`"
käyttäjätunnuksesi on hakka
$ mail –s postia `cat kutka`< tdsto
$ echo Summa on `expr 2+5`
```


Myös komentojen ryhmittely suluin synnyttää aliprosessin.

```
$ (cd ..;ls);ls
```

Ohjelma/komentotdsto voi käynnistää lisää prosesseja. Yhteenkuuluvat prosessit muodostavat työn.

```
$ man man &
$ jobs
[1]+  Stopped           man man
$ ps
  PID TTY          TIME CMD
19435 pts/125  00:00:00 bash
20100 pts/125  00:00:00 man
20103 pts/125  00:00:00 sh
20104 pts/125  00:00:00 sh
20106 pts/125  00:00:00 gtbl
20108 pts/125  00:00:00 nroff
20109 pts/125  00:00:00 less
20110 pts/125  00:00:00 nroff
20112 pts/125  00:00:00 ps
```

Prosessien / töiden hallintakomentoja

ps näytä prosessien tilatietoja (mm. nrot) (process status)
jobs näytä työnumerot
top näytä tietoja järjestelmän prosesseista

& suorita taustaprosessina

kill lopeta taustaprosessi / työ
ctrl-c lopeta edustaprosessi ("kill")

ctrl-z pysäytä väliaikaisesti edustaprosessi ("stop")

fg, bg jatka pysäytettyä työtä edustalla / taustalla (foregroud, background)

%nro siirrä työ edustaprosessiksi
%nro & jatka työtä taustaprosessina

pgrep etsi prosesseja nimellä tai muilla kriteereillä
pkill tapa prosesseja nimellä tai muilla kriteereillä
killall tapa prosesseja niiden nimellä

sleep pistä prosessi odottamaan annettu aika (sek)

history tulosta viimeisimpien komentojen lista

at suorita komento/komentotdsto mainitulla hetkellä
atq näytä suoritusta odottavat työt (at queue)
atrm poista suoritusta odottava työ (at remove)

```
$ at 10:30am today
at> echo ”Syömään” | mail –s ”Asia on” avrll
at>ctrl-d
```

crontab toistuvien töiden hallintaan (ylläpito tykkää tästä...)
ks. /etc/crontab

nohup käynnistä komento, joka jää suoritukseen vielä istunnon päättyessä (no hangup)

nice aseta prosessin prioriteetti komennon suoritusajaksi

Katso yksityiskohdat ja käyttötapa manuaalisivuilta.

Editointi, tekstin muotoilu

Tekstieditoreita

ed Rivieditori (vrt sed). Ks. kadonnut kansanperinne.

vi, vim UNIXin alkuperäinen tekstieditori.

Ystävämme ed sekä vi (visual editor) kuuluivat jo alkuperäiseen ympäristöön ja ne löytyvät kaikista UNIXeista. Vim (vi improved) on ko. ohjelman uudistettu versio.

vi-editorin ohjeita esim. osoitteesta

http://cs.stadia.fi/~lehtonen/Unix/unix.htm

pico *[valitsimet][tdstonimet]* Ruutupohjainen editori.

Pine-postiohjelman mukanaan tuoma tekstieditori.

Ohjeita esim. osoitteesta

http://cs.stadia.fi/~lehtonen/Unix/unix.htm

emacs *[valitsimet][tdstonimet]*

GNU-projektin tuottama editori (sisältää paljon muutakin toiminnallisuutta). Löytyy myös Windows-ympäristöön.

Ikkunaympäristöön: Xemacs
Kevytversio ilman valikoita: em (mikro-emacs)

Komentoikkunassa ei voi käyttää hiirtä, ikkunaympäristössä kyllä! Näppäinkomennot räätälöitävissä.

emacs-komentoja

`^x^c`	lopeta käyttö - jos tallentamatta, kysyy varmistuksen: kirjoita sana *no* tai *yes* (Ctrl pohjassa, sitten x ja c)
`^x^s`	(save) talleta työtiedosto
`^x^w`	(write) talleta uudella nimellä
`^x^f`	(find) tiedoston nouto uudeksi työtiedostoksi
`^x i`	(insert) lisää tdsto kohdistimen osoittamaan kohtaan
`^g`	katkaise kesken komennon
`^x^u`	(undo) peru viimeisin editointikomento

Osaa täydentää vanhan tdstonimen sarkain-näppäimellä.

Kursoria voi liikuttaa nuolinäppäimin ja PageUp/PageDown-näppäimillä. Home/End-näppäimet eivät välttämättä toimi.

`^a`	rivin alkuun
`^e`	rivin loppuun
`Esc-<`	kohdistin tdston alkuun (paina ensin Esc sitten perään <)
`Esc->`	kohdistin tdston loppuun
`Esc-x goto-line`	kohdistin halutulle riville

Merkin voi poistaa Backspace tai Delete-näppäimellä (saattavat toimia 'väärinpäin'). `^h` toimii kuten Backspace.

`^k`	poistaa loppurivin leikepuskuriin
`^y`	palauttaa merkit leikepuskurista

Muistutus

Ctrl-S (stop) hyydyttää näytön ja ***Ctrl-Q*** aktivoi sen jälleen!

Alueen poisto, kopiointi ja siirto paikasta toiseen:

1) vie kohdistin siirrettävän kohdan alkuun ja paina
   `Ctrl-välilyönti`
2) vie kohdistin siirrettävän alueen loppuun ja paina
   `^w` (leikkaa pois) tai `Esc-w` (kopioi)
   - merkitty alue siirtyy leikepuskuriin
3) vie kohdistin kohtaan, jonne haluat liittää puskurista,
   paina `^y` (yank)

Etsi ja korvaa:

`^s` etsintä kohdistimesta eteenpäin (incremental search)
`^r` etsintä kohdistimesta taaksepäin (reverse)
`Esc-%` korvaa merkkijonon esiintymät toisella (quering replace)
- kysyy esiintymän kohdalla korvataanko:
  y=yes, n=no, q=quit, ! = korvaa kaikki kyselemättä

Esim: editoi / käännä / korjaa

Käynnistä editori `emacs test.c`
- kirjoita / muokkaa ohjelmakoodia
- talleta muutokset komennolla `^x^s` (save)

a) avaa toinen komentoikkuna
- käännä `gcc test.c`
- selaa virheilmoituksia tässä ikkunassa,
  korjaa koodia edellisessä ikkunassa
- virheen riville komennolla `Esc-x goto-line`

b) poistu komentotulkkiin komennolla `Esc-!` (command processor)
- käännä `gcc test.c`
- virheilmoitukset tulevat omaan ikkunaan
- valitse selailtava ikkuna komennolla `^x o` (other)
- sulje valittu ikkuna komennolla `^x 0` (zero)

Lisää emacs-ohjeita esim. osoitteesta
http://cs.stadia.fi/~lehtonen/Unix/unix.htm

Tekstin muotoiluohjelmia

fmt

(simple text formatter)

Yksinkertainen muotoiluohjelma, joka tuottaa esim. tasapitkiä rivejä. Ei tavutusta.

pr *[optiot] [tiedostot]*

(preparation for printing)

Tekstin esivalmistelu: mm. sivun ja rivin pituus määrättävissä, tekstin muotoilu palstoiksi, sivunumerot sekä ylä- ja alaotsikot.

nroff, troff

(typeset or format documets for display or printer)

UNIXin oma tekstimuotoilija: sivutus, tasaus, lihavointi, alleviivaus jne. Muotoiltava teksti ja siihen sisällytetyt muotoiluohjaukset kirjoitetaan tavallisella editorilla tiedostoksi. Mm. man-komento käyttää tätä.

tex, latex

(text formatting and typesetting)

Monipuolinen tekstin muotoilu/ladonta. Löytyy myös mm. Windows-ympäristöön. Teksti ja muotoiluohjaukset kirjoitetaan tavallisella editorilla tiedostoksi.

```
$ emacs tiedosto.txt kirjoita teksti ja muotoilut
\documentstyle [12pt,a4,Finnish]{article}
\begin{document}
\section{Luvun otsikko}
Tässä on ensimmäisen luvun ensimmäisen kappaleen teksti,
jossa {\bf lihavoitua tekstiä.}
\end{document}

$ tex tiedosto.txt      tee .dvi välitiedosto
$ dvips tiedosto.dvi    tee .ps tiedosto (PostScript)
```

Tietovirran muokkaus, suodinohjelmia

Merkkien poisto /vaihto

colrm *[sareke [sarake]]* (remove columns from file)

Leikkaa kultakin syöttövirran riviltä pois merkkejä. Jos argumenttina on vain yksi sarake, poistaa siitä alkaen rivin loppuun, muuten poistaa annettujen sarakkeiden välisen osan (ko. sarakkeet mukaan lukien).

```
$ ls -l | colrm 12 58
```

cut *[valitsimet] osa [erotin] [tiedostot]* (remove sections from line)

Leikkaa tdston kultakin riviltä mukaan määrätyt osat. Osa määräytyy sarakenumeroiden tai erotinmerkkien avulla (kentät).

Valitsimia:

-c (character position) käytä sarakenumeroja
-f (field) käytä kenttänumeroita
-d (delimiter) kenttien välinen erotinmerkin, oletus \t (tabulointi)

```
$ cut -d: -f1,5 /etc/passwd | tee tunnukset
```

tr *[valitsimet] [merkit1 [merkit2]]* (character transliteration)

Poistaa tai vaihtaa määrätyt syöttövirran (stdin) merkit toisiksi tulos-virrassa (stdout). *merkit1* vaihtuu *merkit2* vastinmerkkien mukaan.

Valitsimia:

-c (complement) muut kuin merkit1 korvataan
-d (delete) merkit1 poistetaan
-s (squeeze) peräkkäiset merkit2 tulosteessa vain kerran

```
$ tr -cs A-Za-z ´\012´ < sanat1 > sanat2
```

(kukin sana omalle rivilleen)

```
$ cut -d'\t' -f1-2 ilmo | tr '\t' ':'
```



```
$ cat muuta                                   (isot tdstonimet pieniksi)
for x in $* do
    mv $x `echo $x | tr A-Z a-z`
end
$ muuta ABC*.txt
$ muuta ABC_1.txt ABC_2.txt ABCDEFG.txt
```

Rivien valinta ja muokkaaminen

sed *[-n] [-e script] [-f tdsto] [tdsto] ...* (stream editor)

Eräajoeditori. Valitsee tdstosta (myös stdin) rivejä ja muokkaa niiden sisältöä editointi-komennoilla. Käyttöä komentotdstoissa (shell-skripteissä).
Tulostuvien rivien määrää voi rajoittaa käyttämällä valistinta -n (no print) ja komentoa p (print).

Komennot ovat muotoa

[rivi [,rivi] toiminto [argumentit]

ja ne voivat olla erillisessä tdstossa (-f). Komentoja

d poista rivi/rivejä (delete)
a lisää rivi mainitun rivin perään (append)
i lisää rivi mainitun rivin etupuolelle (insert)
c vaihda kohdealueen tilalle (change)
s korvaa merkkejä(substitute): *s/vanha/uusi/[g][p][w tdsto]*
p tulosta rivi stdoutiin (print)
r lisää tdsto (read)
w kirjoita tdstoon (write)
n ota seuraava rivi käsittelyyn (next)
q lopeta (quit)

Argumenttien paikalla voi käyttää säännöllisiä lausekkeita (-e), ja toimintoa voi rajoittaa argumenttien jälkeisillä lipukkeilla.
Jos komento muuttaa käsiteltävän rivin sisältöä, seuraava komento kohdistuu jo muutettuun riviin. Lopuksi rivi tulostuu stdoutiin.

Esimerkkejä:

```
$ sed '5 q' tdsto                          (käsittele vain rivit 1-5)

$ sed -n '3,6 p' tdsto                     (vain rivit 3-6 tulostuvat)

$ sed -e 's/[Uu]nix/UNIX/g' ohje > ohje.uusi

$ sed 's/ *$//' tdsto                      (poista tyhjät rivinlopuista)

$ cat script
   2 a\
   tämä kolmanneksi\
   ja tämä neljänneksi
   3 c\
   vanha rivi jää jalkoihin
   10,$ s/vanha/uusi/g                     ($=viimeinen rivi, g= global)
   /tämä/ s//tama/
$ sed -f script tdsto

$ cat script2
   1,10 !w temp                            (muut paitsi 1-10)
   1,10 s/vanha/uusi/w temp2
   1 r temp2
$ sed -f script2 tdsto
```

- skriptin rivi suoritetaan aina koko tdstolle ja seuraava rivi saadulle tulokselle

awk (Aho, Weinberger, Kernighan)

perl (Practical Extraction and Reporting Language)

ks. myös a2p (awk to perl translator)

Tarkoitettu erityisesti sellaisten tekstitiedostojen käsittelyyn, jossa riveillä on selkeä yhtenäinen tietuerakenne

- kenttien välissä erotinmerkki, esim. `\t` tai **`:`**

Perusidea: ohjelma (skripti) nopeasti suoritettavaksi

- 'itsestäänselvyyksiä' ei tarvitse merkitä
- ei käännöksiä vaan tulkinta
- ei muuttujien esittelyjä

Sopivat suodinohjelmiksi ja raporttien generointiin

- automaattinen lukeminen: kaikki tdston rivit
  - käsittelyyn vain valitut
  - kenttien erottelu helppoa
- etsintä (rivin valinta) grepin tapaan (säännölliset lausekkeet)
- tulosterivin muotoilu halutunlaiseksi
- laskee automaattisesti mm. rivin järjestysnumeron ja kenttien lukumäärän

Hyödynnetty C-kielestä tuttuja ideoita

- kontrollirakenteet kuten C:ssä: valinta, toisto, ..
- tulosteen muotoilu kuten C:ssä (printf ja sprintf)
- numeeriset ja merkkijonomuuttujat
- taulukkomuuttujat, assosiatiiviset taulukot

Tarkempi tutustuminen ja käyttö skipataan tällä kertaa (näistähän syntyisi jo oma kurssi...).

Järjestäminen

sort *[valitsimet] [+kenttä [-kenttä]] [tdstot]*

Tdston järjestäminen, stdin-tdstoon viitataan tavuviivalla -

Valitsimia:

-b ohita alussa olevat tyhjämerkit ja tabuloinnit (blank)
-c tarkista onko järjestetty (check)
-d sanakirjajärjestys: vain kirjaimet, numerot ja tyhjämerkit (dictionary)
-f käsittele pienet kirjaimet isoina (fold)
-i ei välitä ruudulla näkymättömistä merkeistä (ignore)
-m järjestettyjen tiedostojen lomitus (merge)
-u liittyy lomitukseen, vain yksi kutakin tulostdstoon (unique)
-n numerojono järjestetään arvon mukaan (numeric)
-o tulostdston nimi (output)
-r käänteinen järjestys (reverse)
-tc kenttien erottimena merkki c (char), oletus: tyhjä tai \t (tab)

+s -e järjestämiskentät (sort field) kentästä s (start) kenttään e (end), jos e puuttuu koskee loppuriviä.
Molempiin voi liittää merkkisiirtymän muodossa f.c (field.char).
HUOM. ens. kenttä on 0

```
$ ls -l | sort +5 +6n
```
(ajan mukaan järjestettynä)

```
$ sort +0 -1 +4n cars
```
(tulos stdoutiin)

```
$ sort +2.3 -3 -n tdsto
```
(numeerisen arvon mukaan)

```
$ sort -u +0f +0 lista
$ sort -d -t: +2 -o kayttajat /etc/passwd
$ sort -d -t: uudet |
  sort -mdub -t: -o kurssi kurssi –
```
(yhdistäminen)

Tiedostojen yhdistely

cat *[valitsimet] [tiedostot]* (concatenate and display)

Tulostaa tiedostot peräkkäin stdoutiin.

-n liitä rivinumerot rivin alkuun (number)
-b numeroi muut kuin tyhjät rivit (blank)
-s tulosta peräkkäisistä tyhjistä rivistä vain yksi (substitute)
-u älä käytä puskurointia (unbuffered)
-v esitä myös ei kirjoittuvat merkit (visible)

tee *[valitsimet] [tiedostot]*

T-liitin. Ohjaa tulostuksen stdoutin lisäksi tdstoon.
-a lisää tulostus tdston loppuun (älä kirjoita päälle) (append)
-i älä välitä keskeytyksestä (ignore)

join *[valitsimet] tdsto1 tdsto2* (join lines from two file)

Yhdistää järjestetyistä tdstoista samalla avaimella löytyvät tiedot. Tuloksena on yksi tulosrivi kustakin yhdistetystä parista (avain esiintyy vain kerran). stdin-tdstoon viitataan nimellä -.

-a1 tuota lisäksi tulosrivi tdsto1:n parittomista (append)
-a2 tuota lisäksi tulosrivi tdsto2:n parittomista
-a3 tuota lisäksi tulosrivi molempien parittomista
-e mjono korvaa tyhjät tuloskentät merkkijonolla (expression)
-j[1|2].m (join) yhdistelyavaimena käytettävän kentän numero
-o lista tulostukseen (output) valittavat kentät n.m, missä n on tdsto ja m kenttänumero
-tc kenttien erottimena merkki c (tab)

```
$ cat henkilot
020150123A:Virtanen Ville:lehtori
311260321X:Lahtinen Lassi:assistentti
$ cat osoitteet
Virtanen Ville:Vantaa 160 :221133
Lahtinen Lassi:Helsinki 74:112233
$ cat virat
assistentti:A18
lehtori:A21
$ cat keruu
echo htun:nimi:virka:palkka:osoite:puhelin
sort -t: +2 -d henkilot |
join -t: -o 1.1 1.2 1.3 2.2 -j1 3 - virat |
sort -t: +1 -d -r |
join -t: -o 1.1 1.2 1.3 1.4 2.2 2.3 -j1 2 - osoitteet
$ keruu
htun:nimi:virka:palkka:osoite:puhelin
311260321x:Lahtinen Lassi:assistentti:A18:Helsinki 74:112233
010150123A:Virtanen Ville:lehtori:A21:Vantaa 160:221133
```

paste *[-s] [-d list] tdsto tdsto ...* (merge lines of files)

Yhdistää useamman tdston rivejä tai yhden tdston peräkkäisiä rivejä yhdeksi riviksi. Rivin maksimipituus 511, korkeintaan 12 tiedostoa, osien erottimena \t, rivin lopussa \n.

-d list pistä eri tdstoista otettujen rivien erottimeksi (delimiter)
tässä mainittuja merkkejä. Listaa käytetään renkaana.
-s yhdistä tdston peräkkäisiä rivejä (subsequent)

```
$ unalias ls                        (poista mahd. alias-määrittely)
$ ls | paste - - - -                (neljässä sarakkeessa)
$ paste -s -d"\t\n" puppu           (yhdistä peräkkäisiä rivejä)
```


Tiedostojen vertailu

uniq *[-valitsimet [rivin osa]] [lähde] [kohde]*

Etsii ja poistaa järjestetystä tiedostosta duplikaattirivejä.

Valitsimia:
-u (unique lines) tulosta vain rivit, joista ei ole duplikaatteja
-d (duplicate lines) tulosta vain duplikaattirivit (vain yksi kopio)
-c (count) liitä rivin eteen esiintymien lukumäärä
-f (fields) siirtymä rivin alusta kenttinä
-w (width) vertaa vain rivin alkuosaa
+n siirtymä merkkeinä rivin tai annetun kentän alusta

comm *[-valitsimet] tdsto tdsto* (select/reject lines common to sorted files)

Kahden järjestetyn tdston vertailu: näyttää erilaiset (valitsimet 1 ja 2) ja samanlaiset rivit (3). stdin-tdstoon viitataan **-** :lla

Tulostus: rivit, jotka vain jommassakummassa ja rivit, jotka molemmissa. Valitsimet 123 poistaa tulostuksesta halutut osat.

diff *[valitsimet] tdsto1tdsto2* (find differences between two file)

Kahden tdston vertailu: etsii eroavat rivit ja esittää erot käskyinä (add, change, delete), joilla tdsto1:stä saa tdsto2:n. stdin-tdstoon viitataan **-**:lla.

Valitsimia:
-e esittää erot ed/sed-editorin käskyinä (HUOM: editorille ohjattaessa loppuun lisättävä käskyt write ja quit) (edit)
-b älä anna ylimääräisten tyhjämerkkien hämätä (trailing blanks)
-w poista vertailua varten kaikki tyhjämerkit (all blanks)
-i käsittele isot ja pienet kirjaimet samoina (ignore case)
-t säilytä sisennykset (tab)

```
$ cat kori-1
peruna
porkkana
lanttu
$ cat kori-2
peruna
porkkana
nauris
$ diff kori-1 kori-2
4d3                                   lisää riviksi 4, poista rivi 3
< lanttu                        lisää tämä
```

cmp *[valitsimet] tdsto tdsto [skip1] [skip2]* (byte-by-byte comparison)

Kahden tdston vertailu. Jos eroja, ilmoitetaan tavu ja rivi, jossa ero huomattiin.

Stdin-tdstoon viitataan **-** :lla.

skip1 siirtymä vertailun aloituskohtaan
skip2
-l ilmoittaa kaikista eroavaisuuksista (lines)
-s älä tulosta mitään, aseta vain exit-koodi (silent)
(0=samat, 1=eroavat, 2=vikaa argumenteissa)

Tiedostojen arkistointi ja tiivistäminen

---

tar *toiminto [tarkennin] [tdstot]* (tape archiver)

Tiedostoarkiston käsittely: luonti, tdstojen lisäys, listaus ja nouto. Muodostaa tdstoista yhden paketin, jonka voi purkaa takaisin alkuperäisen nimisiksi tdstoiksi. Alkujaan mg-nauhaa varten (varmuuskopiot, arkistointi).

(ks. myös zoo, cpio, ar ja bar)

Toimintoja:

r talleta arkiston loppuun (write)
x nouda/pura arkistosta (extract)
t tarkista onko arkistossa (table of contents)
u lisää arkistoon (ellei jo ennestään) (update)
c luo uusi arkisto (create)

Valitsimia:

v tulosta tietoja talletuksista/noudoista (verbose)
f arkistona käytettävä tdsto (file) (ei laite). **-**, jos stdin, stdout
w kysele kjältä varmistuksia (query)
m aseta noudossa tdston aikaleimaksi noutoaika (modification day)
b jakson pituus (1-20) (block)

```
$ tar cvf pruju.tar .          (työhsto alihstoineen tdstoon pruju.tar)
    ... verbosen tulostuksia
$ tar tf pruju.tar             (arkiston tdstonimien listaaminen)
    ... tähän tulee listaus tdstonimistä (hstopolkuineen)
$ tar xvf pruju.tar            (arkiston purku)
    ...verbosen tulostuksia
```

gzip *[-valitsimet] [tdstot]* (compress files)
gunzip *[-valitsimet] [tdstot]* (expand files)
zcat *[-valitsimet] [tdstot]* (display expanded contents)

`gzip` tiivistää tdston pienempään tilaan (Lempel-Ziv koodaus). Esim. tekstitdston koko voi pienentyä 50-60%.

Tiivistetty tdsto (lopussa .gz) korvaa pakkaamattoman tdston. Alkuperäisen tdston tunnistetiedot säilyvät pakatussa tdstossa (omistaja, suojaukset, aikaleimat).

```
$ gzip pruju.tar                (syntyy pruju.tar.gz)
```

`gunzip` purkaa tiivistyksen ja tuottaa alkuperäisen tdston.

```
$ gunzip pruju.tar.gz           (syntyy pruju.tar)
$ tar xf pruju.tar
$ tar xfz pruju.tar.gz          (käytä ensin gunzip:iä)
```

Tiivistetty tdsto voidaan listata pakkaamattomassa muodossa komennolla `zcat` (sama kuin gunzip –c). Itse tdsto säilyy muuttumattomana.

```
$ zcat tdsto.Z
```

compress *[-cfv] [-b bits] [tdstot]* (compress files)
uncompress *[-cv] [tdstot]* (expand files)

Kuten gzip, mutta vanhempi. Tiivistetty tdsto korvaa pakkaamattoman tdston (nimen lopussa .Z).

Käytä mieluummin gzip-ohjelmaa, se tiivistää paremmin.
gunzip osaa käsitellä myös compressilla tiivistettyjä tdstoja.

```
$ tar cvf pruju.tar pruju       (ensin arkistointi)
$ compress pruju.tar            (sitten tiivistys)

$ uncompress pruju.tar.Z
$ tar xvf pruju.tar

$ tar xfZ pruju.tar.Z           (sekä tiivistyksen että arkiston purku)
```


zip *[-valitsimet] [-b bits] [tdstot]* (compress files)
unzip *[-valitsimet] [tdstot]* (expand files)

Arkistopakkauksen luonti ja tiivistys samassa ohjelmassa. Luo uuden tdston, jonka oletusliitteenä .zip. Tästä on myös mm. Windows-versio. Oikea valinta, jos tarvetta siirtää tdstoja järjestelmien välillä.

```
$ zip –h                        (näytä valitsimet)
$ zip words.zip words
$ zip -R foo '*.c'              (recursive, myös kaikki alihakemistot)

$ unzip foo.zip
```

Yhteydet toisiin koneisiin

---

ssh *[-valitsimet] [-l kjätunnus] kone [komento]* (secure shell)
ssh *[-valitsimet] kjätunnus@kone [komento]*

Etäyhteyden muodostaminen toiseen UNIX-koneeseen tai komennon suoritus etäkoneessa.

Korvaa vanhat telnet-, rlogin- ja rsh-komennot (remote). ssh:n välittämä tieto on kryptattua, toisin kuin em. ohjelmissa. Niissä myös salasana kulkee salaamattomassa muodossa, joten älä käytä niitä!

Myös etäkoneessa oltava voimassaoleva tunnus. Jos tunnusta ei erikseen anna komennon yhetydessä, käyttää samaa tunnusta kuin paikallisessa koneessa.

Verkon koneessa voi olla tdsto **/etc/hosts**, josta käy ilmi ne 'luotettavat' koneet, joiden kanssa on yhteiset tunnukset. Kun kjä ottaa yhteyttä omalla tunnuksellaan tällaiseen koneeseen, ei se kysy salasanaa. Lisäksi etäkoneessa kjän kotihakemistossa oleva **.shosts** voi ilmaista koneet ja tunnukset, joilla on oikeus yhteyteen ilman salasanakyselyä. Jos kumpaakaan edellisistä ei voi käyttää, kysyy järjestelmä salasanan.

```
$ ssh -l averell kone.alue.fi
averell@kone.alue.fi's password:
```

Etäistunto päätetään **exit**-komennolla.

Etäkomentoa suoritettaessa paikallinen stdin-syöttövirta toimitetaan etäkoneelle ja siellä suoritettavan komennon stdout-tulosteet kopioidaan edelleen paikalliseen stdoutiin.

Jos etäkomennossa on jokerimerkkejä, on ne laitettava lainausmerkkeihin. Muuten ne evaluoituvat paikallisesti ennen käskyn ssh käynnistystä. (taustaprosessin optio -n: stdin = /dev/null)

```
$ ssh averell@kone.domain.fi ls
$ ssh kone 'gcc -o jummi jammi.c'
$ ssh kone -n 'gcc -o a a.c &> errs' &
```


scp *[-valitsimet] lähde kohde* (secure file copy)

Tdstojen kopiointi koneelta toiselle. Käyttöä, jos koneet eivät käytä yhteistä verkkotiedostojärjestelmää. Tunnuksia koskevat säännöt kuten ssh:ssa.

Lähde ja kohde muodossa *[käyttäjä@][kone:]polkunimi*

Polkunimi on kohdekoneessa suhteellinen kjän kotihakemiston suhteen. Jos etäkoneen tdstonimessä haluaa käyttää jokerimerkkejä, on ne laitettava lainausmerkkeihin.

```
$ scp tdsto averell@kone.alue.fi:tdsto_uusinimi
$ scp kone:tdsto kja@kone:tdsto
$ scp –p kja@kone.alue.fi:nimi tdsto
$ scp kone:'jokerinimi' hakemisto
```

Valitsimia:

-p	säilytä alkuperäiset aikaleimat (preserve)
-r	kopioi hakemisto alihakemistoineen (recursive)
-v	tulosta lisätietoja (debuggaus) (verbose)
-q	älä tulosta tietoa kopioinnin edistymisestä

ftp *[valitsimet][kone]* (file transfer program)

Tdstojen siirto TCP/IP-protokollia käyttävien koneiden välillä. Komennolla voi kirjoittautua toiseen koneeseen ja päästä rajoitettuun käyttöympäristöön.

Kohdekoneen nimen voi antaa komentorivillä. Yhteyden luonnin jälkeen kohdekone kysyy käyttäjätunnusta ja salasanaa ja jatkaa sen jälkeen ftp-komentotilassa.

ftp ei edellytä kjätunnusta kohdekoneessa. Julkiseen käyttöön asetettuja tdstoja voi kopioida kirjoittautumalla tunnuksella **anonymous**. Saattaa pyytää salasanaksi s-postiosoiteen.

ftp-ohjelman merkitys vähentynyt WWW:n myötä.

ftp-komentoja:

help *[knto]*	opastus
?*[knto]*	
quit,bye,close	yhteyden päättäminen

pwd	työhakemisto kohdekoneessa
dir *[hsto][tdsto]*	hakemistolistaus, mahd. tdstoksi
ls *[hsto][tdsto]*	lyhyempi hakemistolistaus
cd *hsto*	työhakemiston vaihto kohdekoneessa
lcd *hsto*	työhakemiston vaihto työkoneessa
binary	siirto binäärisenä
ascii	siirto asciina
get *tdsto [tdsto]*	tdston kopiointi työkoneelle
put *tdsto [tdsto]*	tdston kopiointi kohdekoneelle
!*[knto]*	suorita komento työkoneessa
open *kone*	yhteydenotto, jos ei konetta komentorivillä

Siirron voi keskeyttää ^C:llä.

Käyttäjien välinen kommunikointi

Kelvollisia käyttäjätunnisteita ovat

kjätunnus
kjätunnus@kone
kjätunnus@cs.stadia.fi
Etunimi.Sukunimi@stadia.fi

whoami kertoo millä käyttäjätunnuksella työskentelet

who kertoo kenellä on istunto koneessa

w näyttää kenellä on istunto koneessa ja mikä ohjelma on suorituksessa

finger *[optiot][tunnus[@kone]]*

Antaa tarkempia tietoja käyttäjistä (nimellä tai kjätunnuksella), esim. joutenoloaika, onko lukematonta postia jne., tulostaa myös käyttäjän kotihakemistossa olevan tiedoston .plan (oltava r-oikeus)

```
finger
finger lassara
finger lassara@hylka
```

write *kjätunnus [tty-tunnus]*

Yhtäaikaa samassa koneessa olevien käyttäjien välinen keskustelu. Lähetys riveittäin. Päättyy painamalla `^D` (syötteet loppu, eof). 'Häirintäyritykset' voi kieltää komentamalla **mesg n** (peruminen mesg y). tty-päätetunnuksen näkee esim. komennolla who.

talk *kjätunnus[@kone]*

Yhtäaikaa koneessa olevien käyttäjien välinen keskustelu. Lähetys merkeittäin. Parempi synkronointi kuin edellisessä. Keskustelu päättyy painamalla `^C`.

mail *kjätunnus[@kone]*

BSD-UNIXin alkuperäinen ohjelma postin lähetykseen ja lukemiseen. Kadonnutta kansanperinnettä. Sopii edelleen s'postin 'automaattiseen' lähettämiseen komentotiedostoista.

```
$ mail -s "Otsikko" lassara < tekstitdsto

$ mail lassara@firma.fi
Subject: Kuhan koetan
Aloitin kirjoitella, kun ei ole muutakaan tekemistä.
Lopetan tähän, kun ei ole muutakaan sanomista.
^D
```