Süvateadmiste käsiraamat

Mandriva Linux 2006

http://www.mandriva.com

Süvateadmiste käsiraamat: Mandriva Linux 2006
Avaldatud September 2005

NeoDoc (`http://www.neodoc.biz`)Camille Bégnis, Christian Roy, Fabian Mandelbaum, Roberto Rosselli del Turco, Marco De Vitis, Alice Lafox, John Rye, Wolfgang Bornath, Funda Wang, Patricia Pichardo Bégnis, Debora Rejnharc Mandelbaum, Mickael Scherer, Jean-Michel Dault, Lunas Moon, Céline Harrand, Fred Lepied, Pascal Rigaux, Thierry Vignaud, Giuseppe Ghibò, Stew Benedict, Francine Suzon, Indrek Madedog Triipus, Nicolas Berdugo, Thorsten Kamp, Fabrice Facorat, Xiao Ming, Snature , Guylhem Aznar, Pavel Maryanov, Annie Tétrault, Aurelio Marinho Jargas, Felipe Arruda, Marcia Gawlak Hoshi, Bob Rye, Jean-Luc Borie, ja Roberto Patriarca

Õigusalane teave

Käesoleva käsiraamatu loomisel kasutatud vahendid

Käesoleva käsiraamatu on kirjutanud ja seda hooldab NeoDoc (`http://www.neodoc.biz`). Tõlgete taga seisab NeoDoc, Mandriva ja paljud tõlkijad.
Käesolev käsiraamat on kirjutatud DocBook XML-i kasutades. Käsiraamatu failide haldamiseks kasutati dokumentatsioonihaldussüsteemi Borges (`http://sourceforge.net/projects/borges-dms`). XML-lähtefailide töötlemiseks tarvitati programme xsltproc ja `jadetex`, kasutades Norman Walshi laaditabelite kohandatud versiooni. Ekraanipiltide tegemiseks pruugiti `xwd`'d või GIMP-i ning nende teisendamiseks programmi `convert` (paketis ImageMagick). Kõik mainitud programmid on vaba tarkvara ning neid pakub ka Teie Mandriva Linuxi distributsioon.

Sisukord

Tabelite nimekiri

Eessõna

1. Mandriva Linuxi info

Mandriva Linux on GNU/Linuxi distributsioon, mida toetab ja arendab Mandriva S.A. ja mis sündis Internetis 1998. Selle peamine siht oli ja on pakkuda hõlpsasti kasutatavat ning kasutajasõbralikku GNU/Linuxi süsteemi. Mandriva kaks alussammast on avatud tarkvara ja koostöö.

7. aprillil 2005 muutis senine Mandrakesoft oma nime, võttes liitumise kajastamiseks Brasiilia firmaga Conectiva endale nimeks Mandriva. Kõige olulisem toode, Mandrakelinux, sai pärast kahe firma ühinemist nimeks Mandriva Linux.

1.1. Kontakteerumine Mandriva Linuxi kogukonnaga

Toome järgnevalt ära mõned viidad veebilehekülgedele, kus leiab infot mitmete Mandriva Linuxi aspektide kohta. Kui soovite saada rohkem teada Mandriva firma kohta, uurige meie veebilehekülge (`http://www.mandriva.com/`). Samuti võite tutvuda Mandriva Linuxi distributsiooni veebileheküljega (`http://www.mandrivalinux.com/`) ning selle erinevate variantidega.

Mandriva Expert (`http://www.mandrivaexpert.com/`) on Mandriva tugiplatvorm. See pakub uuelaadset kogemust, mis tugineb usaldusele ning naudingule, mida võib anda teiste väärtustamine nende pakutava panuse eest.

Me kutsume teid ka tellima mitmesuguseid meililiste (`http://www.mandriva.com/community/resources/newsgroups`), mille vahendusel Mandriva Linuxi kogukond näitab oma elujõulisust ja elavust.

Kindlasti võiksite tutvuda ka meie turvalehekülgega (`http://www.mandriva.com/security`). See kogub kõikvõimalikku Mandriva Linuxi distributsiooniga seotud turvainfot. Te leiate siit nii turvalisuse ja vigadega seotud nõuandeid kui ka kerneli uuendamise juhiseid, mitmesuguseid turvalisusele pühendatud meililiste ning Mandriva Online'i (`https://online.mandriva.com`). See on kahtlemata koht, mida iga turvalisuse üle muret tundev administraator või kasutaja peaks külastama.

1.2. Klubiga ühinemine

Mandriva pakub kasutajatele arvukalt mitmesuguseid soodustusi Mandriva Club'i (`http://club.mandriva.com`) vahendusel:

- tavaliselt ainult tasulistes karbiversioonides saada olev kommertstarkvara, näiteks spetsiaalsed draiverid, kommertsrakendused, vabavara ja demoversioonid;
- võimalus välja pakkuda uut tarkvara ja hääletada selle kaasamise poolt spetsiaalse süsteemi vahendusel;
- ligipääs enam kui 50 000 RPM-paketile kõigi Mandriva Linuxi distributsioonide tarbeks;
- allahindlus toodetele ja teenustele, mida pakub Mandriva Store (`http://store.mandriva.com`);
- võimalus kasutada paremaid, spetsiaalselt klubiliikmetele mõeldud peegelsaite;
- võimalus osaleda mitmekeelsetel foorumitel ja lugeda asjalikke artikleid erinevates keeltes;
- võimalus kasutada Mandriva Teadmistebaasi (`http://club.mandriva.com/xwiki/bin/view/KB/`) ehk Wikilaadset veebilehekülge, kust leiab materjale väga paljude teemade kohta (näiteks haldus, võrguühendused, probleemid ja nende lahendamine ja nii edasi);
- võimalus vestelda Mandriva Linuxi arendajatega klubivestlustes (`https://www.mandrivaclub.com/user.php?op=clubchat`);
- võimalus süvendada oma teadmisi GNU/Linuxist Mandriva e-õppetundide abil (`http://etraining.mandriva.com`).

Finantseerides Mandrivat Mandriva Club'i kaudu on Teil võimalus vahetult edendada Mandriva Linuxi distributsiooni ja võimaldada meil pakkuda võimalikult head GNU/Linuxi töölauda kõigile meie kasutajatele.

1.3. Liitumine Mandriva Online'iga

Mandriva pakub välja väga mugava viisi hoida oma süsteem automaatselt alati uuena ning vältida vigu ja turvaauke. Külastage Mandriva Online'i veebilehekülge (`https://online.mandriva.com/`), kus saate teenuse kohta täpsemat teavet.

1.4. Mandriva toodete ostmine

Mandriva Linuxi kasutajatel on võimalik osta meie tooteid meie internetikauplusest Mandriva Store (`http://store.mandriva.com/`). Sealt ei leia mitte ainult Mandriva Linuxi tarkvara, operatsioonisüsteeme ja "live" alglaade-CD-sid (näiteks Move), vaid ka spetsiaalseid tellimispakkumisi, tuge, muude tootjate tarkvara ning litsentse, dokumentatsiooni, GNU/Linuxiga seotud raamatuid, rääkimata juba Mandrivai pakutavatest tarbekaupadest.

1.5. Mandriva Linuxile kaasaaitamine

Kõigi nende paljude nutikate ja osavate inimeste teadmised, kes kasutavad Mandriva Linuxit, tulevad ainult kasuks veel parema Mandriva Linuxi loomisel:

- **Tarkvarapakettide loomine.** GNU/Linuxi süsteem luuakse peamiselt Internetis leitavatest programmidest. Et need aga kõik omavahel korralikult töötaksid, tuleb need korralikult nii-öelda pakendada.
- **Programmeerimine.** Neid programme, mida Mandriva otseselt toetab, on tõesti väga palju: leidke selline, mis Teid kõige enam tundub kütkestavat, ja pakkuge oma abi programmi arendajatele.
- **Internatsionaliseerimine.** Te võite aidata meil tõlkida veebilehekülgi, programme ja nendega kaasasolevat dokumentatsiooni.

Uurige arendusprojektide veebilehekülge (`http://qa.mandriva.com/`), kus saate täpsemalt teada, mil moel on just Teil võimalik anda oma panus Mandriva Linuxi helgesse tulevikku.

2. Süvateadmiste käsiraamatu tutvustus

Süvateadmiste käsiraamat on mõeldud neile, kes soovivad põhjalikumalt tundma õppida oma Mandriva Linuxi süsteemi ja ära kasutada selle kõiki tohutuid võimalusi. Me loodame, et pärast käsiraamatuga tutvumist suudate hõlpsasti tulla toime GNU/Linuxi masinate igapäevase haldamisega. Anname nüüd ülevaate käsiraamatu kahest osast, kirjeldades lühidalt kõiki peatükke:

- Esimeses osas (*Sissejuhatus Linuxisse*) tutvustame Teile GNU/Linuxi süsteemi. Me käsitleme selle arhitektuuri, peamisi failisüsteeme ning mõningaid iseärasusi, näiteks failisüsteemi `/proc`.

  Esimeses peatükis (Peatükk 1) räägime UNIX®-i paradigmast ning spetsiifilisemalt GNU/Linuxist. Me tutvustame failide käsitlemise vahendeid ning mõningaid võimalusi, mida pakub shell. Sellele järgneb peatükk (Peatükk 2), milles tuleb lähemalt juttu kõvaketaste haldamisest GNU/Linuxis ning nende jagamisest ehk partitsioneerimisest.

  Me vaatleme esmalt failipuu korraldust (Peatükk 3). UNIX®-i süsteemid kipuvad sageli väga suureks kasvama, kuid igal failil on oma kindel koht kindlas kataloogis. Selle peatüki läbilugemise järel saate teada, kust mingeid faile otsida vastavalt nende rollile süsteemis.

  Järgmises peatükis on teemaks failisüsteemid (Peatükk 4). Pärast saadaolevate failisüsteemide tutvustamist käsitleme failitüüpe ning veel mõningaid asjakohaseid kontseptsioone ja utiliite, näiteks inoded ja torud. Järgnevas peatükis aga (Peatükk 5) tutvustame GNU/Linuxi erilist (virtuaalset) failisüsteemi nimetusega `proc`.
- Teises osas (*Linux sügavuti*) võtame vaatluse alla praktilisemad teemad. Me räägime failisüsteemide ja haakepunktide seostest, tutvustame, kuidas kasutada käsurida oma igapäevatoimingute sooritamiseks, kuidas redigeerida konfiguratsioonifaile lihtsate, kuid võimsate redaktoritega ja veel palju muud.

  Me räägime ***failisüsteemidest*** ja ***haakepunktidest*** (Peatükk 6). Me selgitame, mida need üldse tähendavad, ning toome nende kohta mitmeid praktilisi näiteid.

Seejärel puudutame käsurea kasutamist (Peatükk 7). Me tutvustame failide haldamise tööriistu, näiteks `mkdir` ja `touch`, ning selgitame, kuidas faile ja katalooge failisüsteemis liigutada, kustutada ja kopeerida. Samuti tulevad jutuks failide atribuudid ning nende käsitlemine selliste käskudega nagu `chown` ja `chgrp`. Me räägime ka metamärkide kasutamisest, ümbersuunamisest ja torudest, lõpetamisest ning elementaarsetest tööde juhtimise võimalustest.

Järgmises peatükis tutvustatakse tekstifailide redigeerimist (Peatükk 8). Et enamik UNIX®-i konfiguratsioonifaile on tekstifailid, tekib Teil tõenäoliselt millalgi soov või vajadus redigeerida neid ***tekstiredaktoris***. Siin saate teada, kuidas kasutada UNIX®-i ja GNU/Linuxi maailma kaht kõige kuulsamat tekstiredaktorit: võimsat Emacs'it, mille kirjutas Richard M. Stallman, ja vana head Vi'd, mille kirjutas juba 1976. aastal Bill Joy.

See peaks võimaldama Teil toime tulla süsteemi kõige põhimõttelisema hooldamisega. Järgmises kahes peatükis tutvustame, kuidas praktiliselt kasutada käsurida (Peatükk 9) ja juhtida protsesse (Peatükk 10).

Seejärel (Peatükk 11) selgitame Mandriva Linuxi algkäivitamise protseduuri ja selle võimalikult tõhusat ärakasutamist. Me räägime protsessist init (see vastutabki süsteemi käivitamise eest) ning erinevatest käivitustasemetest (mille tundmine tuleb eriti kasuks süsteemi hooldamisel). Me selgitame ka lühidalt, kuidas kasutada meie tööriista drakxservices teenuste haldamisel.

Järgmises peatükis (Peatükk 12) selgitame, kuidas pääseda turvaliselt ligi kaugsüsteemile (`ssh` vahendusel) selle hooldamiseks, programmide kasutamiseks vms. Me tutvustame põgusalt ühendusskeemi ning räägime `ssh` serveri ja kliendi põhimõttelisest seadistamisest. Vaatluse all on ka `scp` kasutamine.

Käsiraamat lõpeb peatükiga, mille sisuks on tarkvara haldamine käsureal (Peatükk 13). Selles saab teada, kuidas kasutada utiliite urpmi ja urpme. Samuti selgitame tarkvaraallikate haldamise võimalusi.

3. Toimetaja märkus

Avatud tarkvara üks põhimõtteid on see, et alati on oodatud kõigi kaasalöömine! Mandriva Linuxi dokumentatsiooni uuendamine on õigupoolest väga töömahukas ja nõuab palju abistavaid käsi. Te võite abistada päris mitmel moel, eriti suurt ja pidevat huvi tunneme me aga inimeste vastu, kes suudaksid midagi ära teha järgmistes valdkondades:

- kirjutamine või uuendamine;
- tõlkimine;
- toimetamine;
- XML/XSLT programmeerimine.

Kui Teil peaks olema rohkem aega, võite kirja panna või uuendada terve peatüki; kui Teie emakeel ei ole inglise keel, võite aidata tõlkida käsiraamatuid; kui Teil on ideid sisu parandamiseks, andke sellest teada; kui olete osav programmeerija ja soovite arendada meie dokumentide haldamise süsteemi Borges (`http://sourceforge.net/projects/borges-dms`), siis liituge meiega. Ja ärge mingil juhul kõhelge meile ka sellest teada andmast, kui olete leidnud mõne kirjavea!

Kõikvõimalikku infot Mandriva Linuxi dokumentatsiooni kohta on võimalik saada dokumentatsiooni administraatorile (`mailto:documentation@mandriva.com`) kirjutades või Mandriva Linuxi dokumentatsiooni veebilehekülgedelt (`http://qa.mandriva.com/twiki/bin/view/Main/DocumentationTask/`).

Palun pange tähele, et alates 2004. aasta juunist tegeleb Mandriva Linuxi dokumentatsiooni ja programmi Borges arendamisega NeoDoc (`http://www.neodoc.biz`).

4. Raamatus kasutatavad tähistused

4.1. Tüpograafilised tähistused

Me kasutame mitmesuguseid võtteid, et eristada tavalisest tekstist mitmesuguseid spetsiaalseid sõnu või lõike. Järgnevas tabelis on toodud näited kõigi erisõnade või sõnarühmade kohta, mida eristatakse, ning nende tähendus.

Vormindusnäide	Tähendus
inode	Kasutatakse tehnilise termini rõhutamiseks(mida selgitab Lisa A).
`ls -lta`	Kasutatakse käskude ja nende argumentide korral (vt. Sektsioon 4.2.1).
`fail`	Kasutatakse failinimede korral, samuti RPM pakettide nimedes.
ls(1)	Viitab `man`-leheküljele. Selle lugemiseks kirjutage käsureale lihtsalt `man 1 ls`.
`$ ls *.pid`	Kasutatakse sellise teksti hetkvõteteks, mida võite näha oma arvutiekraanil, sealhulgas arvutipoolsed reaktsioonid, programmide nimekirjad jne.
`localhost`	Andmed, mis ei sobi ühtegi eelnevasse kategooriasse, näiteks konfiguratsioonifaili võtmesõnad.
OpenOffice.org	Rakenduse nimed. Sõltuvalt kontekstist võib rakenduse nimi ja käsk olla isegi ühesugune, kuid nende vormindus erineb. Nii on näiteks käsud enamasti väiketähelised, samas kui rakenduste nimed algavad üldjuhul suure tähega.
Failid	Näitab menüükirjet või graafilise kasutajaliidese nuputeksti. Allajoonitud täht näitab kiirklahvi, mis tähendab, et antud korralduse andmiseks tuleb vajutada klahvi **Alt** ja vajalikku tähte.
Le petit chaperon rouge	Tähistab võõrkeelseid sõnu.
Hoiatus!	Spetsiaalsed hoiatused, mis rõhutavad eriliselt midagi. Selliseid asju tuleks tegelikult endale valjusti ette lugeda.

Tähistab märkust. Enamasti annab see lisainfot mingis konkreetses kontekstis.

Tähistab nõuannet. See võib olla üldine nõuanne mingi toimingu sooritamiseks või vihjata mõnele vahvale võimalusele, mis muudab Teie elu kergemaks (näiteks kiirklahvid).

Sellist ikooni nähes olge väga tähelepanelik. See tähendab alati, et öeldakse midagi äärmiselt olulist mingi konkreetse teema kohta.

4.2. Üldised tähistused

4.2.1. Käsurea kokkuvõte

Järgnev näide näitab, milliseid sümboleid võite näha, kui autor kirjeldab käsu argumente:

```
käsk <muudetav argument> [--option={arg1,arg2,arg3}] [lisaargument ...]
```

Selline tähistus on üldlevinud ja seda võib kohata mitmel pool, näiteks `man`-lehekülgedel.

Sümbolite “<” (väiksem kui) ja “>” (suurem kui) vahel paikneb **kohustuslik** argument, mida saate kasutada vastavalt oma vajadustele. Näiteks `<failinimi>` tähendab faili tegelikku nime. Kui selleks on `suva.txt`, tuleb Teil kirjutada `suva.txt`, mitte aga `<suva.txt>` või `<failinimi>`.

Nurksulgudes (“[ ]”) tuuakse ära lisaargumendid, mida võite vastavalt soovile või vajadusele käsus kasutada või mitte.

Kolmikpunkt (“...”) tähendab, et kasutada võib suvalist arvu argumente.

Looksulgudes (“{ }”) on argumendid, mida saab kasutada just selles konkreetses kohas. Vähemalt üks neist tuleb selles kohas ka anda.

4.2.2. Eritähistused

Aeg-ajalt kasutatakse selgitustes klahvikombinatsioone, näiteks **Ctrl**-**R**, mis tähendab, et Teil on vaja vajutada ning hoida all klahv **Ctrl** ning vajutada seejärel klahvile **R**. Sama käib klahvide **Alt** ja **Shift** kohta.

Me kasutame selliste klahvide tähistamisel suurtähti, kuid see ei tähenda, et need tuleks ka suurtähena anda. Siiski võib olla programme, mille korral **R** ei ole sama mis **r**. Selliste programmide korral antakse Teile seda ka teada.

Ka menüüde korral tähendab näiteks Fail→Laadi uuesti kasutaja seadistused (**Ctrl-R**) seda, et Teil tuleb klõpsata menüüribal kirjele Fail (see asub reeglina akna ülemises vasakus nurgas) ning seejärel valida ilmuvas rippmenüüs käsk Laadi uuesti kasutaja seadistused. Lisaks sellele saate teada, et sama tulemuse võite saavutada klahvikombinatsiooniga **Ctrl**-**R** (nagu eespool toodud käsu juures näidatud).

4.2.3. Tüüpkasutajad

Kui võimalik, kasutame oma näidetes kaht tüüpkasutajat:

Queen Pingusa	queen	See on meie nii-öelda vaikekasutaja, keda kohtab enamikus näidetes.
Peter Pingus	peter	Selle kasutaja võib hiljem luua süsteemiadministraator ning me kasutame seda aeg-ajalt teksti muutmiseks vaheldusrikkamaks.

Peatükk 1. UNIX-i süsteemide põhitõed

Mõiste "UNIX®" on vahest Teile juba tuttav. Täiesti võimalik, et Te isegi kasutate UNIX® süsteemi oma töös. Sellisel juhul suudab käesolev peatükk Teile vaevalt midagi väga huvitavat pakkuda.

Kui Te ei ole aga kunagi UNIX® süsteemi kasutanud, oleks selle peatüki läbilugemine hädavajalik. Siintoodud põhimõtete mõistmine võimaldab leida vastuse üllatavalt paljudele küsimustele, mida ikka ja jälle esitavad **GNU/Linux**i maailma esimest korda sattunud. Ja kui Teil veel ei ole neid küsimusi tekkinud, siis leiate siit usutavasti vastuse ka mitmele oma tulevasele küsimusele või probleemile.

1.1. Kasutajad ja grupid

Käesolevas osas tutvustame selliseid mõisteid, nagu kasutaja ja grupid, mille mõistmine on õieti hädavajalik kõigi järgnevate kontseptsioonide jaoks.

Linux on tõeline ***mitmekasutaja***süsteem ning oma GNU/Linuxi masina kasutamiseks peab Teil olema masinas ***konto***. Kui Te tekitasite paigaldamise ajal kasutaja, siis tegelikult tähendas see konto loomist. Kui Te ei peaks mäletama, siis paluti Teil määrata järgmised asjad:

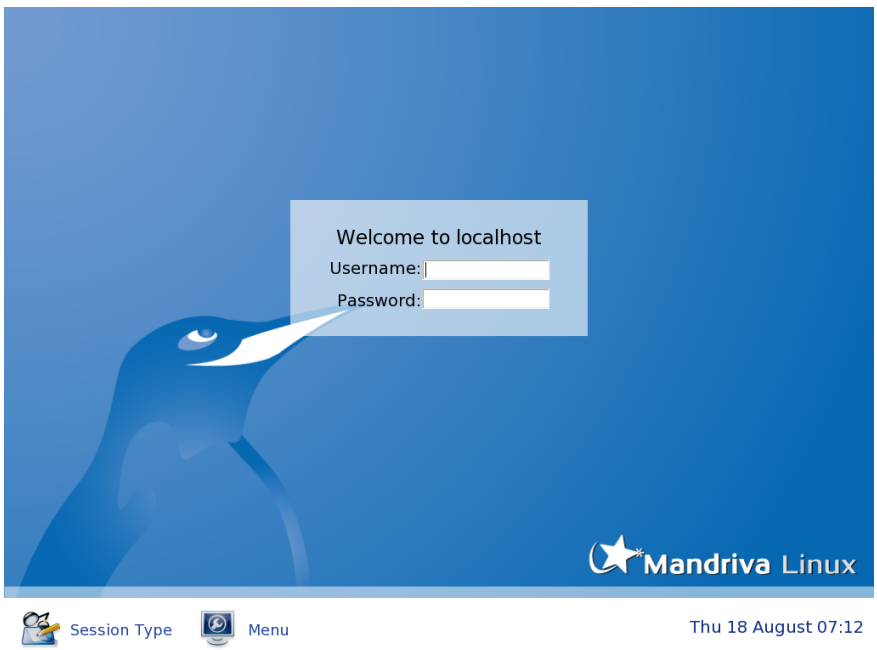
- kasutaja "tegelik nimi" (mis õigupoolest võib olla milline tahes)
- ***kasutajatunnus***
- ja ***parool***.

Kaks kõige tähtsamat parameetrit on seejuures kasutajatunnus (ka kasutajanimi, inglise keeles login) ja parool (inglise keeles password). Süsteemi kasutamiseks lihtsalt peavad need mõlemad olemas olema.

Kasutajat luues luuakse talle ka vaikimisi grupp. Edaspidi näeme, et grupid on väga suureks abiks näiteks failide jagamisel teiste inimestega. Grupis võib olla kui palju tahes kasutajaid ning eriti suurtes süsteemides on nende kasutamine täiesti tavaline. Näiteks ülikoolis võib olla üks grupp igale teaduskonnale, omaette grupid õppejõududele ja nii edasi. Kuid asi võib olla ka vastupidi: üks kasutaja saab korraga kuuluda enam kui ühte gruppi. Matemaatikaõppejõud võib kuuluda näiteks korraga nii õppejõudude gruppi kui ka oma tudengite gruppi.

Kõige elementaarsem öeldud, vaatame nüüd, kuidas sisselogimine tegelikult käib.

Kui alglaadimisel käivitatakse automaatselt graafiline liides, näeb käivitusekraan välja umbes selline, nagu näitab Joonis 1-1.



Joonis 1-1. Graafiline sisselogimine

Sisselogimiseks tuleb esmalt valida nimekirjast oma konto. Siis ilmub uus dialoog, kus tuleb sisestada oma parool . Pange tähele, et parool tuleb kirjutada nii-öelda pimesi, sest tegelikult sisestatud sümbolite asemel näidatakse ainult tärne (`*`). Samas dialoogis saab valida ka seansi tüübi (aknahalduri). Kui olete kõigega valmis, klõpsake nupule Logi sisse.

Kui Te logite sisse konsoolilt ehk "tekstirežiimis", ilmub Teie silmade ette midagi sellist:

```
Mandriva Linux release 2006.0 (koodnimi) for i586
Kernel 2.6.12-6mdk on an i686 / tty1
[masinanimi] login:
```

Sisselogimiseks kirjutage oma kasutajatunnus viibale Login: ja vajutage klahvi **Enter**. Seejärel näitab sisselogimisprogramm (login) viipa Password: ja jääb ootama Teie parooli. Nagu graafilises režiimis, ei näidata ka konsoolis Teie parooli tegelikke sümboleid - õiegi ei näidata seal midagi, isegi mitte tärne.

Pange tähele, et Teil on võimalik ka ühe ja sama kontoga ühes ja samas masinas mitu korda sisse logida erinevatel ***konsoolidel***, sealjuures isegi X'i kasutades. Iga avatav seanss on teistest sõltumatu ning soovi korral võib korraga avatud olla isegi mitu X'i seanssi (kuigi suure ressursivajaduse tõttu pole see üldiselt väga soovitatav). Vaikimisi kasutab Mandriva Linux kuut ***virtuaalset konsooli*** lisaks sellele konsoolile, mis on reserveeritud graafilisele liidesele. Nende vahel saab lülituda klahvikombinatsiooniga **Ctrl-Alt-F<n>**, kus `<n>` on selle konsooli number, millele soovite lülituda. Vaikimisi töötab graafiline liides konsoolil number 7. Niisiis tuleks näiteks teisele konsoolile lülitumiseks vajutada klahve **Ctrl**, **Alt** ja **F2**.

Paigaldamise ajal soovis DrakX Teie käest parooli ka ühele väga erilisele kasutajale, nimelt administraatorile (`root`). See tähistab süsteemiadministraatorit, kelleks kõige tõenäolisemalt olete Teie ise. Süsteemi turvalisuse huvides on mõistlik kaitsta konto `root` alati ühe korraliku, raskesti äraarvatava parooliga!

Kui Te logite sageli sisse administraatorina (***`root`***), võite väga kergesti teha mõne vea, mis halvab kogu süsteemi. Seda võib teha ka üksainus viga! Tasub ka silmas pidada, et kui Te ei määranud kontole `root` parooli, võib **iga** kasutaja muuta **iga** Teie süsteemi osa (ja isegi muid masinais töötavaid operatsioonisüsteeme!). On selge, et see ei ole kohe üldse mõistlik.

Tasub ka mainida, et sisemiselt ei tuvasta süsteem Teid mitte kasutajatunnuse, vaid Teie nimega seotud unikaalse numbri järgi: *User ID* (UID) . Ka kõiki gruppe tuvastatakse nende *Group ID* (GID), mitte aga nime järgi.

1.2. Failide põhitõed

Võrreldes Windows® ja enamiku muude ***operatsioonisüsteemidega*** käsitleb GNU/Linux faile üsna omamoodi. Käesolevas osas tutvustame kõige silmatorkavamaid erinevusi, kui soovite aga põhjalikumat infot, siis seda pakub Peatükk 4.

Suurimate erinevuste taga seisab otseselt asjaolu, et Linux on mitmekasutajasüsteem: iga fail kuulub ainult ja ainuüksi ühele kasutajale ja ühele grupile. Me ei maininud kasutajatest kõneldes, et neil kõigil on oma isiklik kataloog (niinimetatud ***kodukataloog***). Iga kasutaja on oma kodukataloogi ja kõigi selles loodud failide omanik. Samuti tuleb märkida, et need failid on seotud teatud grupiga, milleks on antud kasutaja esmane grupp. Nagu mainisime (Sektsioon 1.1), võib üks kasutaja kuuluda korraga mitmesse gruppi, aga ainult üks neist on tema jaoks esmane.

Siiski jääks pelgalt faili omanikust väheseks. Kuid faili omanikuna saab kasutaja määrata selle kasutamise **õigused**. Õigustega saab teha vahet kolmel kasutajate kategoorial: faili **omanik**, kõik sellised kasutajad, kes kuuluvad failiga seotud **gruppi** (kannab ka nimetust ***omanikugrupp***), aga ei ole selle omanikud, ning **teised**, kelle hulka kuuluvad kõik, kes ei ole ei faili omanikud ega selle omanikugrupi liikmed.

Õigusi on kolme erinevat sorti:

1. **Lugemise** õigus (`r`): võimaldab kasutajal lugeda faili sisu. Kataloogi korral tähendab kasutaja õigust näha selle sisu (s.t. kataloogis leiduvaid faile).
2. **Kirjutamise** õigus (`w`): võimaldab muuta faili sisu. Kataloogi korral tähendab kasutaja õigust lisada sellesse faile või neid sealt eemaldada, isegi kui ta ei ole nende failide omanik.
3. **Käivitamise** õigus (`x`): võimaldab faili käivitada (normaalselt on selline õigus mõttekas ainult käivitatavate failide puhul). Kataloogi korral tähendab kasutaja õigust seda ***läbida*** ehk siis kataloogi kas vahe- või

lõpp-peatusena siseneda. Pange tähele, et see ei ole sama, mis lugemisõigus: Teil võib olla õigus kataloogi läbida, ilma et Teil oleks õigust selle sisu teada saada!

Õigusi on võimalik igati kombineerida. Te võite näiteks anda endale õiguse mingit faili lugeda ja samas keelata kõigile teistele sellele ligipääsu. Faili omanikuna saate muuta ka selle gruppi (aga ainult siis, kui kuulute ise uude gruppi).

Selgitame seda kõike faili ja kataloogi näitel. Allpool on näha käsu `ls -l` sisestamise tulemus ***käsureal***:

```
$ ls -l
total 1
-rw-r-----   1 queen    users           0 Jul  8 14:11 a_file
drwxr-xr--   2 peter    users        1024 Jul  8 14:11 a_directory/
$
```

Käsu `ls -l` tulemused on (vasakult paremale) järgmised:

- Esimesed kümme sümbolit näitavad faili tüüpi ja selle õigusi. Kõige esimene sümbol tähistab faili tüüpi: kui see on tavaline fail, seisab seal kriips (`-`), kui aga kataloog, siis täht `d`. Failitüüpe on veelgi, neist teeme juttu hiljem. Ülejäänud üheksa sümbolit näitavad õigusi. Tegelikult on siin lausa kolme õigusterühmaga seotud kolm sümbolirühma. Neist esimene näitab faili omanikuga seotud õigusi, järgmised kolm sümbolit kõigi omanikugruppi kuuluvate kasutajate õigusi ning viimased kolm sümbolit kõigi ülejäänud kasutajate õigusi. Kriips (`-`) tähendab, et antud õigust ei ole määratud.
- Seejärel on kirjas faili viitade arv. Hiljem näeme, et faili unikaalne identifikaator ei ole mitte selle nimi, vaid number (the ***inode number***) ja et ühel failil võib kettal olla ka mitu nime. Kataloogi korral on viitade arvul spetsiaalne tähendus, millel peatume samuti veidi hiljem.
- Seejärel on näha faili omaniku ja omanikugrupi nimi.
- Lõpuks näidatakse faili suurust (***baitides***) ja selle viimase muutmise aega ning kõige viimasena ka faili või kataloogi nime.

Vaatame nüüd lähemalt failide õigusi. Kõigepealt jätame kõrvale tüüpi tähistava esimese kriipsu, mille järel failil `faili_nimi` on järgmised õigused: `rw-r-----`. Selgitame neid nüüd lähemalt.

- Esimesed kolm sümbolit (`rw-`) näitavad omaniku õigusi, kelleks antud juhul on queen. See tähendab, et kasutajal queen on õigus faili lugeda (`r`) ja selle sisu muuta (`w`), aga mitte seda käivitada (`-`).
- Järgmised kolm sümbolit (`r--`) käivad kõigi kohta, kes pole queen, kuid kuuluvad gruppi `users`. Neil on õigus faili lugeda (`r`), kuid mitte seda muuta ehk kirjutada või käivitada (`--`).
- Viimased kolm sümbolit (`---`) käivad kõigi kohta, kes pole queen ega kuulu gruppi `users`. Neil kasutajatel ei ole antud failile üldse mingeid õigusi ja nende jaoks on fail sisuliselt “nähtamatu”.

Kataloogi `kataloogi_nimi` õigused on `rwxr-xr--`, niisiis:

- peter kui kataloogi omanik võib näha kataloogis leiduvaid faile (`r`), faile kataloogi lisada või neid sealt eemaldada (`w`) ning kataloogi ka läbida (`x`).
- Kõik, kes ei ole peter, aga kuuluvad gruppi `users`, võivad näha kataloogis leiduvaid faile (`r`), kuid ei saa neid eemaldada ega lisada (`-`), küll võivad aga kataloogi läbida (`x`).
- Kõik ülejäänud kasutajad võivad ainult näha kataloogi sisu (`r`). Et neil puuduvad õigused `wx`, ei saa nad faile muuta ega kataloogi siseneda.

Toodud reeglitel on **üks** oluline erand: `root`. `root` võib muuta kõigi failide atribuute (õigusi, omanikku ja omanikugruppi) ka siis, kui ta ei ole ise omanik, ning kuulutada end näiteks ise faili omanikuks! `root` võib lugeda faile, millele tal puudub lugemisõigus, läbida katalooge, kus talle pole sellist õigust antud ja nii edasi. Kui ka `root` on ilma mingi õiguseta, võib ta selle iga kell endale ise võtta. `root` valitseb täielikult süsteemi, mis ühtlasi tähendab, et Teil peab olema tubli usaldus isiku vastu, kellele omistasite administraatori (`root`) parooli.

Lõpuks tasub ära märkida ka erinevusi failinimede käsitlemisel UNIX® ja Windows® maailmas. Kindlasti on UNIX® selles suhtes palju paindlikum ja märksa vähemate piirangutega.

- Failinimi võib sisaldada mis tahes sümbolit, sealhulgas mittenähtavaid, välja arvatud ASCII sümbol 0, mis tähistab stringi lõppu, ja `/`, mida kasutatakse kataloogi eraldajana. Lisaks tuleb arvestada, et kuna UNIX®

on tõstutundlik, on failid `readme` ja `Readme` kaks erinevat faili, sest tähti `r` ja `R` peetakse UNIX® süsteemides **erinevaks**.

- Vahest panite juba tähele, et failinimel ei pea olema laiendit, kui Te seda just ise ei soovi. Faililaiendid ei tuvasta failide sisu ei GNU/Linuxis ega õigupoolest peaaegu üheski operatsioonisüsteemis. Samas on "faililaiendid" üsna mugavad abivahendid. Punkt (`.`) on UNIX® korral lihtsalt üks sümbol teiste seas, kuid sellel on ka eritähendus: nimelt tähistavad UNIX® korral punktiga algavad failinimed "peidetud faile"[1], kusjuures punktiga võivad alata ka kataloogide nimed.

Samas tuleb märkida, et paljud graafilised rakendused (failihaldurid, kontoritöörakendused jne.) kasutavad tõesti faililaiendeid failide tuvastamiseks. Seepärast on mõttekas kasutada faililaiendeid vähemalt selliste rakenduste puhul, kus valitseb nende järele vajadus.

1.3. Protsessid

Protsess tähendab konkreetset käivitatavat programmi ja selle ***keskkonda***. Me mainime siinkohal vaid kõige olulisemaid GNU/Linuxi ja Windows® erinevusi (täpsemalt kõneleb neist Peatükk 10).

Kõige tähtsam erinevus on otseselt seotud juba tutvustatud **kasutaja** kontseptsiooniga: iga protsess käivitatakse selle kasutaja õigustega, kes protsessi käivitab. Seesmiselt tuvastab süsteem protsessi unikaalse numbriga, niinimetatud protsessi ID-ga (*process ID* ehk PID). PID järgi teab süsteem, kes (milline kasutaja) protsessi käivitas ja veel üht-teist ning peab lihtsalt kontrollima protsessi kehtivust. Võtame taas näiteks faili `faili_nimi`. peter võib faili avada ainult ***lugemiseks***, aga mitte ***lugemiseks ja kirjutamiseks***, sest failiga seotud õigused keelavad tal seda teha. Ka siin on reegli erandiks kasutaja `root`.

Seetõttu on ka GNU/Linux sisuliselt kaitstud viiruste eest. Tegutsemiseks peab viirus nakatama käivitatavaid faile. Tavakasutajana ei ole Teil oluliste süsteemsete failide puhul kirjutamisõigust, mis kahandab oluliselt riski. Üldiselt ongi viirused UNIX® maailmas haruharvad külalised. On teada mõned üksikud Linuxile mõeldud viirused ja needki on kahjutud, kui neid käivitab tavakasutaja. Neid käivitades suudab süsteemile kahju tekitada ainult üks kasutaja: `root`.

Kummalisel kombel on viirustevastane tarkvara GNU/Linuxile küll olemas, aga see on eeskätt mõeldud DOS/Windows® failidele! Miks peaks viirusetõrjeprogrammid töötama GNU/Linuxis, kuid puudutama DOS/Windows® faile? Sest üha enam toimivad GNU/Linuxi süsteemid failiserveritena Windows® masinatele, milles neid aitab tarkvarapakett Samba (vt. peatükki Failide ja printerite jagamine *Serverihalduse käsiraamat*us).

Linuxis on protsesside juhtimine väga lihtne. Üks võimalustest on "signaalid", mis lubavad protsessi peatada või tappa sellele vastavat signaali saates. Kuid signaale saab saata ainult omaenda protsessidele. Kui jätta välja `root`, ei luba UNIX® saata signaale mõne muu kasutaja käivitatud protsessile. Kuidas hankida protsessi PID ja saata sellele signaal, tutvustab lähemalt Peatükk 10.

1.4. Kiire sissejuhatus käsurea kasutamisse

Käsurida on kõige vahetum viis saata oma masinale käske. Kui kasutate GNU/Linuxit käsurealt, kogete peagi, et see on märksa võimsam ja võimekam töömeetod kui enamik käsureavõimalusi, millega olete varem kokku puutunud. Kogu selle võimsuse taga seisab asjaolu, et Teil on ligipääs mitte ainult kõigile X'i rakendustele, vaid ka tuhandetele muudele vahenditele just konsoolirežiimis (see vastandub graafilisele režiimile), millel polegi sageli graafilisi vasteid, kõigile nende arvukatele võtmetele ja mitmesugustele kombinatsioonidele, mida oleks tõenäoliselt isegi päris keeruline pruukida nuppude või menüüde abil.

Usutavasti vajab enamik inimesi käsurea kasutamisel alguses veidi abi. Kui Te ei ole seniajani konsoolirežiimis töötanud ja olete kasutanud graafilist liidest, siis võiks esmalt käivitada terminaliemulaatori. Avage peamenüü ning Te leiate lausa mitu emulaatorit menüüs Süsteem+Terminalid. Valige neist meelepärane, näiteks Konsool või RXvt. Sõltuvalt Teie aknahaldurist võite näha terminaliemulaatori ikooni ka paneelil (Joonis 1-2).

1. Vaikimisi ei näidata peidetud faile failihalduris, kui Te just seda spetsiaalselt ei soovi. Terminalis tuleb nii tavaliste kui peidetud failide nägemiseks anda käsk `ls -a`. Enamasti sisaldavad sellised failid seadistusinfot. Vaadake näiteks oma kodukataloogis (`home/`) faile `.mozilla` ja `.openoffice`.

Joonis 1-2. Terminaliikoon KDE paneelil

Terminaliemulaatorit käivitades hakkate tegelikult kasutama shelli. See on programmi nimi, millega Te nüüd suhtlete. Teie ette ilmub ***viip***:

```
[queen@localhost queen]$
```

Siinkohal eeldame, et Teie kasutajanimi on queen ja Teie masina nimi `localhost` (nagu see peaks olema juhul, kui Teie masin ei ole ühendatud mis tahes võrku). Viiba järel asub tühi ruum, kuhu saate kirjutada oma käsud. Pange tähele, et kui olete `root`, asendub viiba sümbol `$` sümboliga `#` (seda küll ainult vaikeseadistuse korral, sest GNU/Linux võimaldab nii seda kui paljusid muid asju oma käe järgi kohandada). Administraatori (`root`) õiguste võtmiseks kirjutage pärast shelli käivitamist `su`.

```
[queen@localhost queen]$ su
# Kirjutage parool; (seda ekraanil ei näidata)
Password:
# käsk exit (või Ctrl+D) toob Teid tagasi oma tavakasutaja kontosse
[root@localhost queen]# exit
[queen@localhost queen]$
```

Kui Te ***käivitate*** shelli esimest korda, satute üldjuhul oma kodukataloogi (`home/`). Kataloogi tuvastamiseks, kus Te parajasti viibite, andke käsk `pwd` (see on väljendi *Print Working Directory* ehk "töökataloogi näitamine" lühend):

```
$ pwd
/home/queen
```

Nüüd vaatame mõningaid äärmiselt kasulikke põhikäske.

1.4.1. cd: kataloogi muutmine

Käsk `cd` esineb täpselt samal kujul ka DOSis, kuid sel on mõned lisajooned. See ei muuda ainult töökataloogi, nagu käsu nimi lubaks arvata. Te võite kasutada ka sümboleid `.` ja `..`, mis tähistavad vastavalt aktiivset kataloogi ja selle ülemkataloogi. Lihtsalt käsu `cd` peale satute tagasi oma kodukataloogi. Käsk `cd -` viib Teid viimati külastatud kataloogi. Aga Te võite ka määrata kasutaja peter kodukataloogi, andes käsu `cd ~peter` (`~` tähendab iseenesest Teie enda kodukataloogi `home/`). Pange tähele, et tavakasutajana ei ole Teil enamasti õigust liikuda mõne muu kasutaja kodukataloogi (`home/`), kui Teil ei ole just spetsiaalselt sellist õigust või kui süsteem ei ole selliselt seadistatud. Küll võib seda teha `root`, nii et võtke endale kasutaja `root` õigused ja proovige järele:

```
$ su -
Password:
# pwd
/root
# cd /usr/share/doc/HOWTO
# pwd
/usr/share/doc/HOWTO
# cd ../FAQ-Linux
# pwd
/usr/share/doc/FAQ-Linux
# cd ../../../lib
# pwd
/usr/lib
# cd ~peter
# pwd
/home/peter
# cd
# pwd
/root
```

Nüüd aga taastage ennast tavakasutajana, andes käsu `exit` ja vajutades klahvi **Enter** (või vajutades lihtsalt klahve **Ctrl-D**).

1.4.2. Mõned keskkonnamuutujad ja käsk echo

Igal protsessil on oma ***keskkonnamuutujad*** ning shell võimaldab neid näha käsuga `echo`. Huvipakkuvamad muutujad on järgmised:

1. `HOME`: see keskkonnamuutuja sisaldab stringi, mis tähistab Teie kodukataloogi.
2. `PATH`: see sisaldab kõigi kataloogide nimekirja, millest shell peab otsima käsu saamisel käivitatavaid faile. Pange tähele, et erinevalt DOS-ist **ei otsi** shell vaikimisi käsku aktiivsest kataloogist!
3. `USERNAME`: see muutuja sisaldab Teie kasutajatunnust.
4. `UID`: see muutuja sisaldab Teie kasutaja ID-d.
5. `PS1`: see määrab, mida viip näitab, ning kujutab endast tihtipeale spetsiaalsete jadade kombinatsiooni. Täpsemalt võib sellest lugeda bash(1) ***manuaalileheküljelt***, milleks andke terminalis käsk `man bash`.

Kui soovite, et shell näitaks muutuja väärtust, peate muutuja nime ette lisama sümboli `$`. Toome siin näite käsu `echo` kohta:

```
$ echo Tere
Tere
$ echo $HOME
/home/queen
$ echo $USERNAME
queen
$ echo Tere $USERNAME
Hello queen
$ cd /usr
$ pwd
/usr
$ cd $HOME
$ pwd
/home/queen
```

Nagu näete, asendab shell muutuja väärtuse enne käsu käivitamist. Vastasel juhul poleks näiteks `cd $HOME` midagi teinud. Tegelikult asendaski shell kõigepealt muutuja `$HOME` selle väärtusega (`/home/queen`), nii et käsurida võttis tegelikkuses kuju `cd /home/queen`, mida meil oligi ju vaja. Sama käib ka käsu `echo $USERNAME` kohta.

Kui mõni keskkonnamuutuja peaks puuduma, saab selle ajutiselt luua käsuga `export KESKKONNAMUUTUJA_NIMI=väärtus`. See tehtud, võite asja üle kontrollida:

```
$ export USERNAME=queen $ echo $USERNAME queen
```

1.4.3. cat: ühe või enama faili sisu näitamine ekraanil

Ega siin palju rohkem polegi öelda: see käsk just seda teebki - saadab ühe või enama faili sisu standardväljundisse, milleks üldjuhul on arvuti ekraan:

```
$ cat /etc/fstab
# This file is edited by fstab-sync - see 'man fstab-sync' for details
/dev/hda2 / ext3 defaults 1 1
/dev/hdc /mnt/cdrom auto umask=0022,user,iocharset=utf8,noauto,ro,exec,users 0 0
none /mnt/floppy supermount dev=/dev/fd0,fs=ext2:vfat,--,umask=0022,iocharset=utf8,sync 0 0
none /proc proc defaults 0 0
/dev/hda3 swap swap defaults 0 0
$ cd /etc
$ cat modprobe.preload
# /etc/modprobe.preload: kernel modules to load at boot time.
#
# This file should contain the names of kernel modules that are
# to be loaded at boot time, one per line.  Comments begin with
# a '#', and everything on the line after them are ignored.
# this file is for module-init-tools (kernel 2.5 and above) ONLY
# for old kernel use /etc/modules
```

```
nvidia-agp
$ cat shells
/bin/bash
/bin/csh
/bin/sh
/bin/tcsh
```

1.4.4. less: lehitseja

Selle programmi nimi kujutab endast väikest sõnamängu: UNIX® kõige esimene lehitseja oli `more`. ***Lehitseja*** on programm, mis võimaldab kasutajal vaadata pisemaid faile lehekülghaaval (täpsemalt küll ekraanihaaval). Põhjus, miks me võtame siinkohal vaatluse alla programmi `less`, mitte aga `more`, on selles, et `less` on märksa kasutajasõbralikum. Programmi `less` on mõtet kasutada, kui soovite vaadata faile, mille sisu ei mahu ekraanile ära. Näide:

```
less /etc/termcap
```

Faili lehitsemiseks kasutage üles ja alla osutavat nooleklahvi. Väljumiseks vajutage klahvi **Q**. `less` suudab küll palju enamatki: vajutage klahvi **H** ja näete kõiki võimalusi.

1.4.5. ls: failide nimekirja näitamine

Käsk `ls` (*LiSt*) teeb sedasama, mida käsk `dir` DOSis, kuid suudab palju enamatki. Õigupoolest on käsk küll seepärast võimekam, et failid ise võimaldavad palju rohkem ette võtta. Käsu `ls` süntaks on selline:

```
ls [võtmed] [fail|kataloog] [fail|kataloog...]
```

Kui käsureal jätta fail või kataloog andmata, näitab `ls` aktiivse kataloogi faile. Käsul on väga palju võtmeid, mistõttu kirjeldame neist vaid mõningaid:

- `-a`: näitab kõiki, sealhulgas ***peidetud faile***. Tuletame meelde, et UNIX® korral on peidetud failid sellised, mille nime ees seisab `.`. Võti `-A` näitab "peaaegu" kõiki faile, mis tähendab kõiki neid faile, mida näitab võti `-a`, välja arvatud "." ja ".."
- `-R`: näitab faile rekursiivselt, s.t. kõiki käsureal antud kataloogide faile ja alamkatalooge.
- `-h`: näitab iga faili järel selle suurust inimsilmale vastuvõetavamal kujul. See tähendab, et failisuuruse juures näidatakse ka ühikut ("K", "M" ja "G"), näiteks "234K" ja "132M". Pange tähele, et suurust arvutatakse arvu 2, mitte arvu 10 astmena. See tähendab, et 1K on tegelikult 1024 baiti, mitte aga 1000 baiti.
- `-l`: näitab failide lisainfot, näiteks selle õigusi, omanikku ja omanikugruppi, faili suurust ja selle viimase kasutamise aega.
- `-i`: näitab faili nime järel selle inode numbrit (faili unikaalset numbrit failisüsteemis, vt. Peatükk 4).
- `-d`: käsitleb käsureal antud katalooge, nagu oleks need tavalised failid, ega hakka nende sisu näitama.

Toome mõned näited:

- `ls -R`: näitab rekursiivselt aktiivse kataloogi sisu.
- `ls -ih images/ ..`: näitab iga faili inode numbrit ja suurust kataloogis `images/` ning aktiivse kataloogi ülemkataloogis.
- `ls -l images/*.png`: näitab kõiki faile kataloogis `images/`, mille nime lõpus seisab `.png`, sealhulgas faili `.png`, kui see on olemas.

1.4.6. Kasulikud kiirklahvid

Ka käsureal saab pruukida paljusid kiirklahve, mis aitavad tublisti aega kokku hoida. Me eeldame siinkohal, et kasutate Mandriva Linuxi pakutavat vaikeshelli bash, kuid toodud kiirklahvid võivad toimida ka muudes shellides.

Kõigepealt nooleklahvid. bash peab meeles varem antud käsud, mida saab näha üles ja alla osutavate nooleklahvidega. Ridu saab kerida maksimaalselt nii palju, kui näeb ette keskkonnamuutuja `HISTSIZE`. Lisaks sellele jäetakse ajalugu ka seansside vahel meelde, nii et Teil on võimalik näha (ja kasutada) ka käske, mida andsite varasemate seansside ajal.

Vasakule ja paremale osutavad nooleklahvid liigutavad kursorit aktiivsel real, mis võimaldab võtta kirjutatud käsus ette muudatusi. Kuid liikuda saab ka teisiti kui ainult ühe sümboli kaupa korraga: **Ctrl**-**A** ja **Ctrl**-**E** näiteks viivad Teid vastavalt aktiivse rea algusse ja lõppu. Klahvid **Backspace** ja **Del** teevad seda, mida nad ikka ja alati teevad. **Ctrl**-**K** kustutab kursori asukohast rea lõpuni, **Ctrl**-**W** aga kursorile eelneva sõna (sama teeb **Alt**-**Backspace**).

Kiirklahvi **Ctrl**-**D** vajutamine tühjal real võimaldab sulgeda aktiivse seansi - märksa kiiremini kui käsku `exit` kirjutades. **Ctrl**-**C** katkestab parajasti käivitatud käsu, välja arvatud siis, kui Te parajasti redigeerisite oma käsurida - sel juhul muutmine peatatakse ja Teie ees seisab taas viip. **Ctrl**-**L** puhastab ekraani. **Ctrl**-**Z** peatab ajutiselt käsu teostamise. See on päris abiks, kui unustasite kirjutamast käsu järele sümboli "&". Näide:

```
$ xpdf MinuDokument.pdf
```

Sellist käsku andes ei saa Te shelli enam kasutada, sest see on antud esiplaanil töötava protsessi xpdf käsutusse. Protsessi seadmiseks taustale ja shelli taaskasutamiseks andke korraldus **Ctrl**-**Z** ja seejärel `bg`.

Lõpuks on ka kiirklahvid **Ctrl**-**S** ja **Ctrl**-**Q**, millega saab peatada ja taastada väljundi näitamise ekraanil. Neid läheb iseenesest harva vaja, aga võib juhtuda, et vajutate **Ctrl**-**S** kogemata (**S** ja **D** on ju klaviatuuril päris kõrvuti). Kui Te peaksite niisiis sattuma olukorda, kus kirjutate midagi, aga Terminal ei näita ühtegi sümbolit, tasuks proovida kiirklahvi **Ctrl**-**Q**. Pange tähele, et kõik sümbolid, mida kirjutasite kogemata vajutatud **Ctrl**-**S** ja **Ctrl**-**Q** vajutamise vahel, ilmuvad ekraanile korraga.

Peatükk 2. Kettad ja partitsioonid

Käesolev peatükk on mõeldud neile, kes tahavad veidi üksikasjalikumalt tundma õppida oma süsteemi tehnilist poolt. Siin antakse täielik ülevaade PC partitsioneerimisest. Seetõttu tuleb peatükk kõige enam kasuks juhul, kui kavatsete oma kõvaketta ise partitsioonideks jagada.

2.1. Kõvaketta struktuur

Kõvaketas on füüsiliselt jagatud sektoriteks. Teatud hulk sektoreid võib moodustada partitsiooni. Põhimõtteliselt on võimalik luua kui palju tahes partitsioone, täpsemalt siiski kuni 67 (3 primaarset partitsiooni ja üks sekundaarne partitsioon, mis sisaldab kuni 64 loogilist partitsiooni): kõiki neid peetakse omaette kõvakettaks.

2.1.1. Sektorid

Lihtsustatult öeldes ongi kõvaketas kõigest rida sektoreid, mis kujutavad endast kõvaketta väikseimaid andmeüksusi. Sektori tüüpiline suurus on 512 baiti. “n” sektoriga kõvakettal on sektorid nummerdatud “0” kuni “n-1”.

2.1.2. Partitsioonid

Mitme partitsiooni kasutamine võimaldab luua füüsilisel kõvakettal mitu virtuaalset kõvaketast. See on mitmes mõttes kasulik:

- Erinevad operatsioonisüsteemid saavad kasutada ketta erinevaid struktuure (neid nimetatakse ***failisüsteemid***eks): nii on see näiteks Windows® ja GNU/Linuxi puhul. Mitme partitsiooni olemasolu ühel füüsilisel kõvakettal lubab sellele paigaldada mitu erinevat operatsioonisüsteemi.
- Jõudluse huvides võib operatsioonisüsteem eelistada erinevate failisüsteemidega erinevaid kettaid, sest neid saab pruukida täiesti erinevate ülesannete jaoks. Selle näiteks on ka GNU/Linux mis nõuab omaette partitsiooni nimetusega saaleala. Seda kasutab virtuaalmälu haldur virtuaalmäluna.
- Isegi kui kõik Teie partitsioonid kasutavad sama failisüsteemi, võib olla kasulik jagada oma OS-i erinevad osad erinevatele partitsioonidele. Üks lihtsamaid näiteid oleks oma failide jagamine kahele partitsioonile: ühel paiknevad Teie isiklikud andmed, teisel programmide andmed. Nii on võimalik näiteks uuendada OS-i programmide partitsiooni isegi täielikult kustutades, kuid säilitada samas omaenda andmed.
- Kuna kõvaketta füüsilised vead tekivad reeglina kõrvutistes sektorites, mitte aga ketta erinevates osades, võib failide jagamine mitme partitsiooni vahel vähendada andmekadu, kui kõvaketas peaks mingil moel füüsiliselt kannatada saama.

Üldiselt määrab partitsiooni tüüp ära failisüsteemi, mida partitsioon sisaldab. Operatsioonisüsteemid ei pruugi alati tunnustada kõiki partitsioonitüüpe. Täpsemalt räägivad sellest Peatükk 6 jaPeatükk 4.

2.1.3. Kõvaketta struktuuri määramine

2.1.3.1. Kõige lihtsam meetod

Me eeldame siin ainult kahe partitsiooni olemasolu: üks on saaleala ja teine failidele[1] nimetusega `root`, mida tähistab `/`.

1. Vaikimisi on Mandriva Linuxis failisüsteemiks `ext3`

Üldjuhul tuleks saalepartitsiooni suuruseks määrata oma RAM mälu kahekordne suurus (s.t. kui Teil on 128 MB RAM mälu, peaks saaleala olema 256 MB). Rohke mälu puhul (512 MB või rohkem) ei ole see siiski nii oluline ja pigem võiks siis jääda väiksema suuruse juurde. Palun arvestage, et saaleala suurust võib piirata ka kasutatav platvorm. Nii on see maksimaalselt 2GB x86, PowerPC ja MC680x0 korral; 512MB MIPS-i korral; 128GB Alpha ja 3TB Ultrasparc-i korral. Arvestage ka seda, et mida suurem saaleala, seda enam vajab OS ressursse (eriti just RAM mälu) selle haldamiseks.

2.1.3.2. Veel üks levinud meetod

Eraldage andmed programmidest. Tõhususe nimel on sageli mõttekas luua rohkem partitsioone süsteemi ja programmide eraldamiseks andmetest. Süsteemipartitsioon sisaldab programme, mida on vaja süsteemi käivitamiseks ja selle elementaarseks hooldamiseks.

Niisiis võime rääkida neljast kõige olulisemast partitsioonist:

Swap

Saaleala ehk saaleala partitsioon, mille suurus peaks üldjuhul olema kaks korda suurem füüsilise RAM mälu suurusest.

Root: /

Kõige tähtsam partitsioon. See sisaldab nii süsteemile kriitilise tähtsusega andmeid ja programme kui ka toimib haakepunktina muudele partitsioonidele (vt. Peatükk 6).

Juurpartitsioon ei pea olema kuigi suur, üldiselt on täiesti piisav 400MB. Kui Te aga kavatsete paigaldada kommertsrakendusi, mille asukohaks on tihtipeale kataloog `/opt`, tuleks juurpartitsiooni vastavalt suurendada. Teine võimalus on luua omaette partitsioon just kataloogi `/opt` tarbeks.

Staatilised andmed: `/usr`

Enamik tarkvarapakette paigutab oma käivitatavad failid ja andmefailid kataloogi `/usr/`. Sellele omaette partitsiooni loomise eeliseks on võimalus jagada kataloogi teiste võrgus olevate masinatega.

Soovituslik suurus sõltub sellest, kui palju pakette soovite paigaldada, ning ulatub 100MB-st väga tagasihoidliku paigalduse puhul kuni mitme GB-ni täieliku paigalduse korral. Sõltuvalt kõvaketta suurusest on tihtipeale piisav 2 kuni 3 GB.

Kodukataloogid: `/home`

See kataloog sisaldab kõigi masina kasutajate isiklikke katalooge. Partitsiooni suurus sõltub mõistetavalt kasutajate arvust ja nende vajadustest.

Veel üks võimalus on **mitte luua** eraldi partitsiooni kataloogi `/usr` failidele: `/usr` võib ka olla lihtsalt kataloog juurpartitsioonil (`/`). Sel juhul tuleb aga vastavalt suurendada juurpartitsiooni (`/`) suurust.

Lõppeks on võimalik luua ka ainult partitsioonid saaleala ja `root` (`/`), eriti kui Te ei ole päris kindel, milleks soovite arvutit kasutada. Sel juhul paikneb Teie kodukataloog (`/home`) juurpartitsioonil (`root`), nagu ka kataloog `/usr` ja muud kataloogid.

2.1.3.3. Eksootilised seadistused

Kui kasutate oma masinat spetsiifilisteks ülesanneteks — näiteks veebiserveri või tulemüürina —, on Teie vajadused kahtlemata täiesti teistsugused kui tavalise tööarvuti korral. Nii on näiteks FTP-serveril tõenäoliselt vaja üsna suurt eraldiseisvat partitsiooni kataloogile `/var/ftp`, samas kui kataloog `/usr` võib olla suhteliselt väike. Sellisel juhul tuleks oma vajadused hoolikalt läbi mõelda juba enne paigaldamise alustamist.

Kui Teil tekib vajadus partitsioonide suurust muuta või ketas kuidagi teisiti jagada, teadke, et enamiku partitsioonide suurust on võimalik muuta ilma süsteemi uuesti paigaldamata ja andmeid kaotamata. Täpsemalt räägib sellest *Partitsioonide haldamine Põhiteadmiste käsiraamat*us.

Veidi harjutades peaksite suutma ka aktiivselt kasutatava partitsiooni liigutada oma vastsoetatud kõvakettale.

2.2. Ketaste ja partitsioonide konventsionaalsed nimed

GNU/Linux kasutab partitsioonide nimetamiseks väga loogilist meetodit. Kõigepealt ignoreeritakse partitsioone nummerdades täielikult selle failisüsteemi. Teiseks antakse partitsioonile nimi vastavalt kettale, kus see asub. Kettad kannavad järgmisi nimetusi:

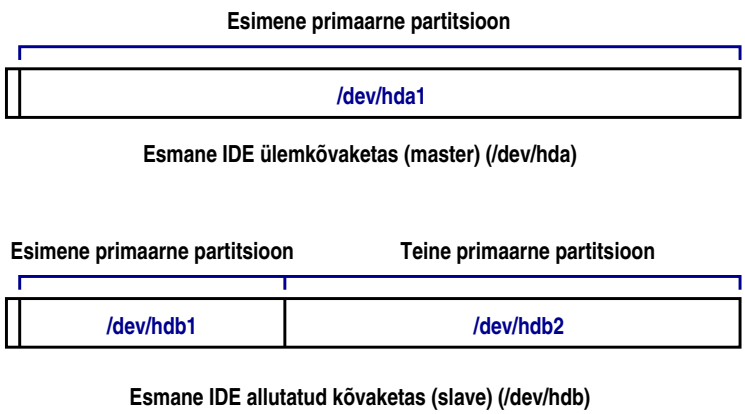
- Primaarne ülem- ja primaarne allutatud IDE seade (olgu see kõvaketas, CD-ROM või mis tahes muu seade) kannavad vastavalt nime `/dev/hda` ja `/dev/hdb`.
- Teisel liidesel kannab ülem nime `/dev/hdc` ja allutatu `/dev/hdd`.
- Kui Teie arvutis on muid IDE liideseid (näiteks IDE liides, mis esineb mõnigatel Soundblasteri kaartidel), kannavad kettad nime `/dev/hde`, `/dev/hdf` jne. Teil võib olla täiendavaid IDE liideseid ka juhul, kui Teil on RAID kontrollerid.
- SCSI kettad kannavad nime `/dev/sda`, `/dev/sdb` ja nii edasi vastavalt oma järjekorrale SCSI ahelas (see sõltub ID-de järjekorranumbrist). SCSI CD-ROM seadmed kannavad nime `/dev/scd0`, `/dev/scd1` ja nii edasi alati vastavalt oma järjekorrale SCSI ahelas.

Kui Teil on SATA IDE kettaid, kehtib neile SCSI nimeskeem.

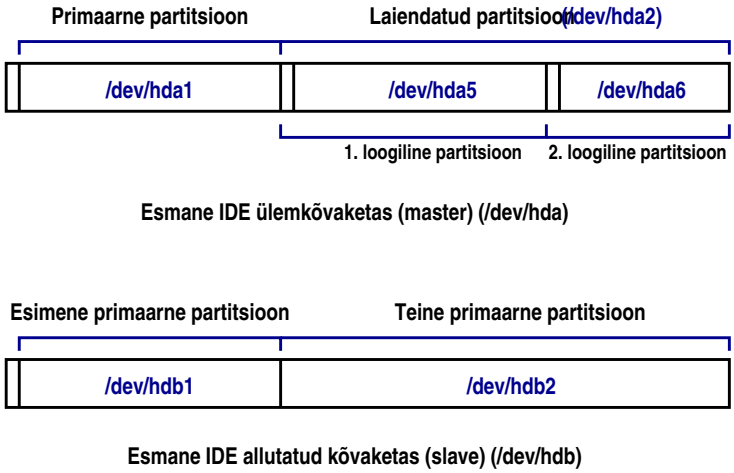
Partitsioonid kannavad nime vastavalt kettale, kus nad paiknevad, järgnevalt (toodud näites kasutame partitsioone primaarselt ülema rollis oleval IDE kettal, kuid see kehtib ka muude ketaste kohta):

- Primaarsed (või laiendatud) partitsioonid kannavad nime `/dev/hda1` kuni `/dev/hda4`.
- Loogilised partitsioonid, kui neid esineb, kannavad nime `/dev/hda5`, `/dev/hda6` ja nii edasi vastavalt nende loomise järjekorrale loogiliste partitsioonide tabelis.

Niisiis annab GNU/Linux partitsioonidele järgmised nimed:



Joonis 2-1. Esimene näide GNU/Linuxi partitsioonide nimede kohta



Joonis 2-2. Teine näide GNU/Linuxi partitsioonide nimede kohta

Toodud teadmistega varustatult peaksite suutma edukalt anda nime oma partitsioonidele ja ketastele, kui Teil tekib selle järgi vajadus. Samuti tasuks tähele panna, et GNU/Linux annab partitsioonile nime isegi siis, kui ei tea, kuidas seda käsitleda (jättes arvesse võtmata asjaolu, et tegemist ei ole loomu poolest GNU/Linuxi partitsioonidega).

Mandriva Linux kasutab nüüd programmi udev (vt. täpsemalt udev FAQ (`http://www.kernel.org/pub/linux/utils/kernel/hotplug/udev-FAQ`)). See tagab täieliku ühilduvuse kirjeldatud nimeskeemi ning standarditega, näiteks sellega, mille on välja töötanud Linux Standards Base Project (`http://www.linuxbase.org/`). Iga seade lisatakse dünaamiliselt süsteemi kohe, kui see on kättesaadav või kui seda vaja läheb.

Peatükk 3. Failipuu ülesehitus

Tänapäeval on üks UNIX® süsteem enamasti väga suur. See kehtib eriti GNU/Linuxi kohta: tohutu kogus tarkvara muudaks süsteemid täiesti juhitamatuks, kui ei valitseks kindlaid reegleid, millised failid kus asuvad.

Selle tunnustatud standardiks on FHS (Failisüsteemi Hierarhia Standard), mille versioon 2.3 nägi ilmavalgust 2004. aasta jaanuaris. Standardit kirjeldava dokumendi leiab väga mitmes vormingus Internetis Pathname'i veebileheküljel (`http://www.pathname.com/fhs/`). Käesolev peatükk pakub ainult lühida kokkuvõtte, kuid sellest peaks piisama teadasaamiseks, millises kataloogis millised failid peaksid asetsema.

3.1. Jagatavad/jagamatud, staatilised/muutuvad andmed

UNIX® süsteemi andmeid võib klassifitseerida järgmiselt: jagatavad andmed võivad olla ühised mitmele võrku kuuluvale arvutile, jagamatud aga mitte. Staatilisi andmeid ei saa tavalisel kasutamisel muuta, muutuvaid andmeid aga küll. Puustruktuuri uurides jagamegi erinevad kataloogid sellistesse kategooriatesse.

Siinne klassifikatsioon on kõigest soovitus ja seda pole sugugi kohustuslik järgida, samas aitab toodud juhistest kinnipidamine kindlasti Teil oma süsteemi palju tõhusamalt hallata. Arvestage ka seda, et staatiliste ja muutuvate andmete eristus käib ainult süsteemi üldise kasutamise, mitte aga konfiguratsiooni kohta. Kui Te näiteks mõnda programmi paigaldate, on Teil enesestmõistetavalt vaja muuta katalooge, mis "normaalselt" on staatilised (näiteks `/usr`).

3.2. Juurkataloog /

Juurkataloog sisaldab kogu süsteemi hierarhiat. Seda ei saa klassifitseerida, sest selle alamkataloogid võivad olla staatilised või jagatavad, aga võivad ka mitte olla. Toome siin ära peamised kataloogid ja alamkataloogid koos klassifikatsiooniga:

- `/bin`: eluliselt vajalikud binaarfailid. Siin on põhikäsud, mida kasutavad kõik kasutajad ja mida on vaja süsteemi toimimiseks: `ls`, `cp`, `login` jne. Staatiline, jagamatu.
- `/boot`: sisaldab faile, mida vajab GNU/Linuxi alglaadur (GRUB või LiLo Inteli, yaboot PPC korral jne.). Siin võib asuda ka kernel, kui see aga nii ei ole, peab kernel asuma juurkataloogis. Staatiline, jagamatu.
- `/dev`: süsteemi seadmefailid (`dev` tähendabki *DEVices* ehk 'seadmed'). Mõned `/dev` failid on kohustuslikud, näiteks `/dev/null`, `/dev/zero` ja `/dev/tty`. Staatiline, jagamatu.
- `/etc`: sisaldab kõiki antud arvutile spetsiifilisi konfiguratsioonifaile. Selles kataloogis ei tohi olla binaarfaile. Staatiline, jagamatu.
- `/home`: siin paiknevad kõigi süsteemi kasutajate isiklikud kataloogid. See kataloog võib olla jagatud või jagamata (mõnes suures võrgus jagatakse seda NFS-i vahendusel). Selles kataloogis asuvad Teie lemmikrakenduste (näiteks e-posti rakenduse või veebilehitseja) konfiguratsioonifailid, mille alguses seisab punkt (".."). Näiteks Mozilla konfiguratsioonifailid asuvad kataloogis `.mozilla`. Muutuv, jagatav.
- `/lib`: selles leiduvad süsteemile olulised teegid, samuti kerneli moodulid alamkataloogis `/lib/modules/KERNELI_VERSIOON`. Siin on kõik teegid, mida vajavad kataloogides `/bin` ja `/sbin` leiduvad binaarfailid. Selles kataloogis peavad olema ka (mittekohustslikud) käitusaja linkija/laadija `ld*` ja dünaamiliselt lingitav C teek `libc.so`. Staatiline, jagamatu.
- `/mnt`: kataloog, mis sisaldab ajutiselt ühendatud failisüsteemide haakepunkte, näiteks `/mnt/cdrom`, `/mnt/floppy` jne. Kataloogi `/mnt` kasutatakse ka ajutiste kataloogide ühendamiseks (näiteks saab USB-kaardi ühendada asukohas `/mnt/removable`). Muutuv, jagamatu.
- `/opt`: sisaldab pakette, mis pole süsteemi toimimisele eluliselt vajalikud. See on mõeldud lisatarkvara jaoks, näiteks paigaldatakse sageli kataloogi /opt Adobe Acrobat Reader jms. FHS soovitab, et staatilised failid (binaarfailid, teegid, manuaalileheküljed jne), mis paigaldatakse `/opt` struktuuri, asetseks kataloogis `/opt/paketi_nimi` ning spetsiifilised konfiguratsioonifailid kataloogis `/etc/opt`.
- `/root`: administraatori (`root`) kodukataloog. Muutuv, jagamatu.

- `/sbin`: sisaldab süsteemi binaarfaile, mis on hädavajalikud süsteemi käivitumisel. Enamikku neist failidest tohib käivitada ainult administraatori (`root`) õigustes. Tavaline kasutaja võib neid samuti käivitada, aga need ei pruugi sel juhul midagi teha. Staatiline, jagamatu.
- `/tmp`: kataloog, mis on mõeldud teatud programmide tekitatavatele ajutistele failidele. Muutuv, jagamatu.
- `/usr`: seda selgitab lähemalt Sektsioon 3.3. Staatiline, jagatav.
- `/var`: andmete asukoht, mida programmid võivad muuta reaalajas (näiteks e-posti serverid, auditiprogrammid, printserverid jne.). Muutuv. Erinevad alamkataloogid võivad olla nii jagatavad kui jagamatud.

3.3. /usr: tõeline isand

Kataloog `/usr` on põhiline rakenduste asukoht. Selle kataloogi binaarfaile ei ole vaja süsteemi käivitamisel ega hooldamisel, mistõttu `/usr` hierarhia võib asuda ja sageli asubki omaette failisüsteemis. Kuna kataloog on tavaliselt väga suur, on `/usr` omaette alamkataloogide hierarhiaga. Mainime siinkohal neist vaid mõningaid:

- `/usr/X11R6`: kogu X Window Systemi hierarhia. Kõik X'i toimimiseks vajalikud binaarfailid ja teegid (kaasa arvatud X'i serverid) peavad asuma siin. Kataloog `/usr/X11R6/lib/X11` sisaldab kõiki X'i seadistusi, mis on erinevatel arvutitel ühesugused. Igale arvutile spetsiifilised konfiguratsioonifailid peaks paiknema kataloogis `/etc/X11`.
- `/usr/bin`: sisaldab enamikku süsteemi binaarfailidest. **Kõik** binaarprogrammid, mida ei ole vaja süsteemi hooldamiseks ja mis ei kujuta endast süsteemi haldamiseks mõeldud programme, peavad asuma selles kataloogis. Ainsaks erandiks on programmid, mida Te ise kompileerite ja paigaldate - need peavad asuma kataloogis `/usr/local`.
- `/usr/lib`: sisaldab kõiki teeke, mida on vaja kataloogides `/usr/bin` ja `/usr/sbin` leiduvate programmide tööks. On ka viit `/usr/lib/X11`, mis viitab kataloogile `/usr/X11R6/lib/X11`, kus leiduvad X Window Systemi teegid (aga ainult juhul, kui X on paigaldatud)[1].
- `/usr/local`: siia tuleb paigaldada kõik rakendused, mida Te ise lähtekoodist kompileerite. Paigaldusprogramm peaks looma rakendusele vajaliku hierarhia.
- `/usr/share`: selles kataloogis leiduvad kõik kirjutuskaitstud, arhitektuurist sõltuvad andmed, mida vajavad kataloogis `/usr` paiknevad rakendused. Muu hulgas on siin ajavööndi ja Teie lokaadi andmed (`zoneinfo` ja `locale`).

Mainime veel ka katalooge `/usr/share/doc` ja `/usr/share/man`, mis sisaldavad vastavalt rakenduste dokumentatsiooni ja süsteemi manuaalilehekülgi.

3.4. /var: reaalajas muutuvad andmed

Kataloog `/var` sisaldab kõiki süsteemis töötavate programmide operatiivandmeid. Erinevalt tööandmetest kataloogis `/tmp` peavad need andmed arvuti taaskäivitamisel jääma puutumata. Siin on mitu alamkataloogi, millest mõnel tasub põgusalt peatuda:

- `/var/log`: sisaldab süsteemi logifaile, mida tasuks lugeda süsteemi probleemide korral (nimetame siinkohal vaid kaht faili: `/var/log/messages` ja `/var/log/kernel/errors`).
- `/var/run`: seda kasutatakse kõigi pärast süsteemi käivitamist kasutatud protsesside jälgimiseks ning see võimaldab Teil nendega midagi ette võtta, kui peaks muutuma süsteemi ***käivitustase*** (vt. Peatükk 11).
- `/var/spool`: sisaldab süsteemi tööfaile, mis ootavad mingit toimingut või töötlemist. Näiteks `/var/spoolcups` sisaldab printserveri tööfaile, `/var/spool/mail` aga e-posti serveri tööfaile (näiteks kõiki süsteemi saabuvaid ja väljuvaid kirju).

1. Arvestage, et Mandriva Linuxis on nüüdseks vaikimisi X Window süsteemina X Window System asemel kasutusel Xorg.

3.5. /etc: konfiguratsioonifailid

Kataloog `/etc` on üks UNIX® süsteemide tähtsamaid katalooge, sest selles paiknevad kõik antud masinale spetsiifilised konfiguratsioonifailid. Ärge **mitte kunagi** kustutage seda näiteks ruumi kokkuhoiu huvides! Kui soovite oma puustruktuuri laiendada mitmele partitsioonile, pidage meeles, et `/etc` ei tohi asuda omaette partitsioonil: seda on vaja süsteemi initsialiseerimisel ja see peab algkäivituse ajal paiknema juurpartitsioonil.

Mõned olulisemad failid:

- `passwd` ja `shadow`: tekstifailid, mis sisaldavad kõiki süsteemi kasutajaid ja nende krüptitud paroole. Te näete faili `shadow` ainult siis, kui kasutatakse variparoole, aga turvakaalutlustel on see paigaldamisel vaikimisi sisse lülitatud.
- `inittab`: see on käsu `init` konfiguratsioonifail, millel on fundamentaalne roll süsteemi käivitamise ajal.
- `services`: see fail sisaldab olemasolevate võrguteenuste nimekirja.
- `profile`: see on shelli süsteemne konfiguratsioonifail. Selle seadistusi saab tühistada shellile spetsiifiliste konfiguratsioonifailidega (näiteks `.bashrc` shelli bash korral).
- `crontab`: käskude perioodilise käivitamise eest hoolt kandva programmi `cron` konfiguratsioonifail.

Programmidele, mis nõuavad rohkemal hulgal konfiguratsioonefaile, on olemas omaette alamkataloogid. Üks selliseid on näiteks X Window System, mille kõik konfiguratsioonifailid on salvestatud kataloogi `/etc/X11`.

Peatükk 4. Linuxi failisüsteem

Teie GNU/Linuxi süsteem paikneb kõvakettal teatud failisüsteemis. Käesolevas peatükis vaatleme GNU/Linuxi failisüsteemide erinevaid aspekte ning tutvustame võimalusi, mida nad Teile pakuvad.

4.1. Mõningate failisüsteemide võrdlus

Paigaldamise ajal on Teil võimalik valida oma partitsioonidele erinevaid failisüsteeme, misjärel partitsioonid vormindatakse erinevate algoritmide kohaselt.

Kui Te ei ole just spetsialist, pole failisüsteemi valik kuigi lihtne. Me anname siin põgusa ülevaate mõnest tänapäevasest failisüsteemist, mida kõike pakub Teile ka Mandriva Linux.

4.1.1. Erinevad failisüsteemid

4.1.1.1. Ext2

Teine laiendatud failisüsteem (Second Extended File System ehk lühendatult ext2FS või lihtsalt ext2) on olnud GNU/Linuxi vaikimisi failisüsteem juba aastaid. See asendas varasema Extended File Systemi (sellest ka "Teine"). ext2 parandas mitmed selle eellasele omased probleemid ja piirangud.

ext2 vastab kõigiti UNIX®-laadsete failisüsteemide tavapärastele standarditele. Juba loomise ajal nähti ette selle edasiareng, kuid samas seati sihiks tubli töökindlus ja hea jõudlus.

Ettevaatust: suuruse muutmiseks tuleb see kõigepealt lahutada.

4.1.1.2. Ext3

Juba nimi Third Extended File System ('Kolmas laiendatud failisüsteem') ütleb, et tegu on ext2 järglasega. See ühildub oma eellasega ning selle suurimaks lisanduseks on ***kirjendamise*** tugi.

"Traditsiooniliste" failisüsteemide (näiteks ext2) üks suuremaid puudujääke on nende suutmatus toime tulla süsteemi äkiliste krahhidega (näiteks voolukatkestuse või tarkvara krahhiga). Üldiselt nõuavad sellist tüüpi sündmused pärast süsteemi taaskäivitamist üsna pikka failisüsteemi struktuuri uurimist, mille käigus üritatakse tekkinud vead parandada. See võib aga vahel kahju isegi suurendada ning tulemuseks on salvestatud andmete osaline või väga halval juhul suisa täielik kaotsiminek.

Selle probleemi lahendab kirjendamine (inglise keeles 'journaling'). Lihtsustatult tähendab see toimingute (näiteks faili salvestamise) kirjapanekut juba **enne** nende sooritamist. Seda võib võrrelda näiteks laevakapteni tööga, kes kasutab logiraamatut päevasündmuste kirjapanemiseks. Tulemus: alati terviklik failisüsteem. Ja kui peakski esinema probleeme, siis on kontroll väga kiire ning võimalik parandus väga piiratud. Seepärast on failisüsteemi kontrollimise aeg proportsionaalne selle tegeliku kasutamise, mitte aga suurusega.

Kokkuvõttes pakub ext3 välja kirjendava failisüsteemi, säilitades samas ajal ext2 struktuuri, mis tagab suurepärase ühilduvuse. Seetõttu on väga lihtne minna ext2'lt üle ext3'le ja vastupidi.

Nagu ext2, tuleb ka see suuruse muutmiseks lahutada.

4.1.1.3. ReiserFS

Erinevalt ext3'st on `reiserfs` kirjutatud nullist peale. See on kirjendav failisüsteem nagu ext3, kuid selle sisemine struktuur erineb radikaalselt, kasutades andmebaasitarkvarast tuntud kahendpuud. Samuti kasutab see failisüsteem muutuvat plokisuurust, mistõttu sobib väga hästi paljude (tuhandede või ka sadade tuhandete) väikeste failide jaoks. Siiski tuleb see hästi toime ka suurte failidega ja on sestap kasutatav väga mitmel otstarbel.

Selle suurust saab muuta "lennult", ilma failisüsteemi lahutamata.

4.1.1.4. JFS

JFS on kirjendav failisüsteem, mille töötas välja ja võttis kasutusele IBM. IBM otsustas algul suletud tarkvarana loodud failisüsteemi vabaks lasta, et kasutada ära vaba tarkvara kogukonna jõudu ja oskusi selle arendamisel. Failisüsteemi sisemine struktuur sarnaneb `reiserfs`'ile.

Selle suurust ei saa GNU/Linuxis muuta.

4.1.1.5. XFS

XFS on kirjendav failisüsteem, mille töötas välja SGI ja mida kasutabki eriti operatsioonisüsteem Irix. SGI otsustas algul suletud tarkvarana loodud failisüsteemi vabaks lasta, et kasutada ära vaba tarkvara kogukonna oskusi ja võimeid. Selle sisemisele struktuurile on omased mitmed erijooned, näiteks reaalaja ribalaiuse toetus, ulatused ja klasterdatud failisüsteemid (küll mitte vabas versioonis).

GNU/Linuxis saab selle suurust ainult suurendada, mite aga vähendada. Suurust saab muuta ainult siis, kui failisüsteem on haagitud.

4.1.2. Failisüsteemide erinevused

	Ext2	**Ext3**	**ReiserFS**	**JFS**	**XFS**
Stabiilsus	Suurepärane	Väga hea	Hea	Keskpärane	Hea
Tööriistad kustutatud failide taastamiseks	Jah (keeruline)	Jah (keeruline)	Ei	Ei	Ei
Taaskäivituse aeg pärast krahhi	Pikk, isegi väga pikk	Kiire	Väga kiire	Väga kiire	Väga kiire
Andmete seisund krahhi korral	Üldiselt hea, aga suur andmete osalise või täieliku kao risk	Väga hea	Keskpärane[a]	Väga hea	Väga hea
ACL-i toetus	Jah	Jah	Ei	Ei	Jah
Märkused: a. Tulemusi krahhi tagajärgede likvideerimisel on võimalik parandada **andmeid**, mitte lihtsalt **metaandmeid** kirjendades, milleks tuleb lisada failis `/etc/fstab` parameeter *`data=journal`*.					

Tabel 4-1. Failisüsteemide iseloomustus

Faili maksimaalne suurus sõltub paljudest teguritest (nt. ext2/ext3 korral ploki suurusest) ning muutub enamasti vastavalt kerneli versioonile ja arhitektuurile.

Kernelis 2.6.X saab plokkseadme piirangust mööda minna, kui kompileerida kernel 'suure plokkseadme' toetusega (`CONFIG_LBD=y`). Täpsemalt lugege selle kohta (ingliskeelseid) materjale Adding Support for Arbitrary File Sizes to the Single UNIX Specification (`http://www.unix.org/version2/whatsnew/lfs.html`), Large File Support in Linux (`http://www.suse.com/~aj/linux_lfs.html`) ja Large Block Devices (`http://www.gelato.unsw.edu.au/IA64wiki/LargeBlockDevices`). Sellise toetuse ja failisüsteemi korral, mis seda toetab, võite saavutada paljude TB suuruse ilma spetsiaalsete failisüsteemi trikkideta, mida teeb näiteks JFS failisüsteemi suuruse suurendamiseks.

4.1.3. Jõudlus

Failisüsteemide jõudlust pole kuigi kerge mõõta ega hinnata. Kõigil testidel on oma piirangud ning nende tulemusi tuleb tõlgendada väga suure ettevaatusega. Isegi mõne kuu või kõigest nädala eest tehtud võrdlused võivad olla tänaseks vananenud (ja enamasti ongi). Ei tasu unustada sedagi, et tänapäeva riistvara (eriti kõvaketaste maht) on nende omavahelisi erinevusi tublisti kahandanud.

Igal süsteemil on oma plussid ja miinused. Õigupoolest sõltub palju sellest, milleks ja kuidas Te oma masinat kasutate. Tavalisele töölauaarvutile sobib väga hästi ext2. Serveri puhul tuleks eelistada kirjendavat failisüsteemi, näiteks ext3. `reiserfs` sobib osaliselt oma algupära tõttu kõige paremini andmebaasiserverisse. JFS-i tasuks eelistada siis, kui kõige olulisem on failide liikumise kiirus süseemis. XFS-i poole tasuks aga silmad pöörata siis, kui Teile pakuvad huvi failisüsteemi täiustatud võimalused. "Tavalise" kasutamise korral on kõigi nelja failisüsteemi tulemused enam-vähem ühesugused, kõigil on aga oma võimalused kohandada neid mingiks kindlaks otstarbeks. Täpsemat infot pakub failisüsteemide enda dokumentatsioon.

4.2. Kõik on fail

*Põhiteadmiste käsiraamat*us tutvustasime juba faili omaniku ja õiguste kontseptsioone, aga UNIX® ***failisüsteemi*** (see käib ka Linuxi failisüsteemide kohta) täielikuks mõistmiseks on vaja uuesti defineerida vastus küsimusele "Mis on fail?"

Siinkohal tähendab "kõik" **tõepoolest** kõike. Kõvaketas, kõvaketta partitsioon, paralleelport, ühendus veebisaidiga, Ethernet-kaart - kõik need on failid. Isegi kataloogid on failid. Linux tunnistab lisaks tavapärastele failidele ja kataloogidele veel paljusid failitüüpe. Pange tähele, et failitüübi all ei mõtle me antud juhul mitte faili **sisu** tüüpi: GNU/Linuxile ja õigupoolest igale UNIX® süsteemile on fail, olgu see siis PNG-pilt, binaarfail või mis tahes muu fail, lihtsalt baidijada. Failide eristamine sisu järgi on rakenduste mure.

4.2.1. Erinevad failitüübid

Käsku `ls -l` andes näete enne kasutamisõigusi sümbolit, mis tähistab failitüüpi. Me oleme juba puutunud kokku kaht tüüpi failidega: tavalised failid (`-`) ja kataloogid (`d`). Kui Te failipuus ringi liigute ja kataloogide sisu endale ette lasete näidata, leiate ka muud tüüpi faile:

1. ***Sümbolseade***: kas spetsiaalsed failid (näiteks `/dev/null`, millega me juba tutvusime) või välisseadmed (jada- või paralleelpordid), millele on ühine see, et nende sisu (kui neil seda üldse on) ei ***puhverdata*** (see tähendab, ei hoita mälus). Selliseid faile tähistatakse tähega `c`.
2. ***Plokkseade***: välisseadmed, mille sisu erinevalt sümbolseadmetest **alati** puhverdatakse. Mõned selle kategooria failid on näiteks kõvakettad, kõvaketta partitsioonid, disketiseadmed, CD-ROM-seadmed ja nii edasi. Plokkseadmed on näiteks `/dev/hda` ja `/dev/sda5`. Selliseid faile tähistatakse tähega `b`.
3. ***Nimeviidad***: neid faile kasutatakse väga palju Mandriva Linuxi süsteemi käivitamisprotseduurides (vt. Peatükk 11). Nagu nimi ütleb, on nende ülesanne linkida failide nimesid, mis tähendab, et nende sisuks ongi viit mingile teisele failile. Nad ei pruugi tingimata viidata olemasolevale failile. Sageli nimetatakse neid ***pehmeteks linkideks*** ning neid tähistatakse tähega `l`.
4. ***Nimega torud***: kui Te huvi tunnete, siis jah, need on väga sarnased torudele, mida kasutavad shelli käsud, aga neid eristabki just see, et neil on nimed. Siiski on nad haruldased ja on üpris väheusutav, et Te retkel failipuus neid üldse kohtate. Selliseid faile tähistatakse tähega `p`. Vt. Sektsioon 4.4.

5. ***Soklid***: see on kõigi võrguühenduste failitüüp, aga ainult mõnel neist on oma nimi. Pealegi on erinevat tüüpi sokleid ja ainult üht tüüpi soklit saab linkida, aga selle käsitlemine väljub käesoleva käsiraamatu raamest. Selliseid faile tähistatakse tähega `s`.

Toome iga faili kohta näite:

```
$ ls -l /dev/null /dev/sda  /etc/rc.d/rc3.d/S20random /proc/554/maps \
/tmp/ssh-queen/ssh-510-agent
crw-rw-rw-    1 root     root       1,   3 May  5  1998 /dev/null
brw-rw----    1 root     disk       8,   0 May  5  1998 /dev/sda
lrwxrwxrwx    1 root     root           16 Dec  9 19:12 /etc/rc.d/rc3.d/
  S20random -> ../init.d/random*
pr--r--r--    1 queen  queen         0 Dec 10 20:23 /proc/554/maps|
srwx------    1 queen  queen         0 Dec 10 20:08 /tmp/ssh-queen/
  ssh-510-agent=
$
```

4.2.2. Infosõlmed

Infosõlmed (inode) on kõrvuti paradigmaga "Kõik on fail" veel üks UNIX® failisüsteemi fundamentaalse tähtsusega osa. Sõna infosõlm ehk ***inode*** on lühend sõnadest "Information NODE", mis otsetõlkes tähendabki infosõlme.

Infosõlmed salvestatakse kettale infosõlmetabelina. Need on olemas igat tüüpi failidele, mida saab failisüsteemis salvestada, sealhulgas kataloogidele, nimega torudele, sümbolseadmetele jne. See toob meid teise kuulsa lause juurde: "Infosõlm on fail". Infosõlmedega identifitseerib UNIX® unikaalselt kõik failid.

Jah, Te saite kõigest õigesti au: UNIX® korral **ei identifitseerita faile mitte nime järgi**, vaid nende infosõlmenumbri järgi[1]. Selle põhjuseks on asjaolu, et ühel failil võib olla mitu nime või siis ei pruugi tal üldse nime olla. UNIX® korral on failinimi lihtsalt kirje kataloogi infosõlmes. Sellist kirjet nimetatakse lingiks. Vaatame nüüd linke lähemalt.

4.3. Lingid

Kõige parem on linke selgitada näite varal. Loome selleks ühe (tavalise) faili:

```
$ pwd
/home/queen/example
$ ls
$ touch a
$ ls -il a
32555 -rw-r--r--  1 queen queen 0 Aug  6 19:26 a
```

Käsu `ls` võti `-i` näitab meile infosõlmenumbrit, mis seisab väljundi esimesel väljal. Nagu näete, ei olnud kataloogis ühtegi faili, enne kui me lõime faili `a`. Veel pakub huvi kolmas väli, mis näitab faili linkide arvu (õigemini küll infosõlme linkide arvu).

Käsu `touch a` võib jagada kaheks eraldi toiminguks:

- infosõlme loomine, millele operatsioonisüsteem annab numbriks 32555 ja mis on tavalise faili tüüpi.
- lingi loomine nimega `a` sellele infosõlmele aktiivses kataloogis (`/home/queen/example`). See tähendab, et fail `/home/queen/example/a` on link infosõlmele numbriga 32555 ja see on praegu ainuke: linkide arvuks näidatakse `1`.

Anname nüüd järgmise käsu:

```
$ ln a b
$ ls -il a b
32555 -rw-r--r--  2 queen queen 0 Aug  6 19:26 a
```

1. **Oluline**: pange tähele, et infosõlmenumbrid on unikaalsed **failisüsteemi sees**, mis tähendab, et sama numbriga infosõlm võib esineda mõnes muus failisüsteemis. Siin tasub tähele panna erinevusi ketta-infosõlmede ja mälu-infosõlmede vahel. Kui kahel ketta-infosõlmel võib olla ühesugune number, kui nad asuvad erinevas failisüsteemis, siis mälu-infosõlmel on unikaalne number failisüsteemist sõltumata kogu süsteemis. Üks viis unikaalsuse saavutamiseks on näiteks räsida ketta-infosõlmede numbrid plokkseadmete identifikaatoritega.

```
32555 -rw-r--r--  2 queen queen 0 Aug  6 19:26 b
$
```

Me loome veel ühe lingi samale infosõlmele. Nagu näete, ei loonud me faili nimega `b`, vaid lisasime samasse kataloogi veel ühe lingi infosõlmele numbriga 32555 ja omistasime sellele lingile nime `b`. Käsu `ls -l` väljund näitab nüüd, et ühe lingi asemel on infosõlmel kaks linki.

Nüüd anname sellise käsu:

```
$ rm a
$ ls -il b
32555 -rw-r--r--  1 queen queen 0 Aug  6 19:26 b
$
```

Näeme, et kuigi me kustutasime "originaalfaili", jäi infosõlm siiski alles. Aga nüüd on ainus link sellele fail nimega `/home/queen/example/b`.

See tähendab, et failil ei pruugi UNIX® korral nime ollagi, vaid see võib olla ka üks või mitu ***linki*** ühes või mitmes kataloogis.

Ka kataloogid salvestatakse infosõlmedes. Nende linkide arv langeb kokku neis leiduvate alamkataloogide arvuga. Selle põhjuseks on asjaolu, et kataloogil on vähemalt kaks linki: kataloog ise (kirje `.`) ja tema eellaskataloog (kirje `..`). Nii on kahe alamkataloogiga kataloogil vähemalt neli linki: `.`, `..` ja mõlema alamkataloogi lingid.

Tüüpiliseks näiteks failidest, mida ei lingita (s.t. neil pole nime), on võrguühendused. Te ei näe näiteks kunagi kuskil oma failipuus faili, mis vastaks Teie ühendusele Mandriva Linuxi veebileheküljega (`http://www.mandrivalinux.com`), kui palju Te ka ei otsiks. Ka siis, kui kasutate shellis ***toru***, on olemas küll torule vastav infosõlm, aga mitte link. Veel üks näide nimeta infosõlme kohta ajutised failid. Te loote ajutise faili, avate selle ja siis eemaldate. Fail on olemas ajal, mil Te seda avatuna hoiate, kuid keegi teine ei saa seda avada (sest pole nime, mida avada). See tähendab ka seda, et kui rakendust peaks tabama krahh, eemaldatakse ajutised failid täiesti automaatselt.

4.4. "Anonüümsed" torud ja nimega torud

Võtame uuesti ette torude näite, sest see on päris huvipakkuv ning aitab ühtlasi paremini selgitada linkide olemust. Kui kasutate toru käsureal, loob shell selle Teie eest ning talitab sellega nii, et enne toru antud käsk kirjutab sinna ja pärast toru antud käsk loeb sealt. Kõik torud, olgu nad siis anonüümsed (mida kasutab näiteks shell) või nimega (vt. allpool), käituvad FIFO (esimesena sisse, esimesena välja) põhimõttel. Me vaatasime juba torude kasutamist shellis, aga antud juhul ei tee paha seda korrata:

```
$ ls -d /proc/[0-9] | head -5
/proc/1/
/proc/2/
/proc/3/
/proc/4/
/proc/5/
```

Üks asi, mida Te antud näite korral ei märka (sest see juhtub liiga kiiresti, et seda oleks võimalik silmaga näha), on see, et torru kirjutamine on olemuselt blokeeriv. See tähendab, et kui käsk `ls` torru kirjutab, on see blokeeritud seni, kuni toru teises otsas olev protsess infot loeb. Selle efekti visualiseerimiseks tuleb Teil luua nimega torud, millel erinevalt torudest, mida kasutab shell, on nimed (s.t. nad on erinevalt shelli torudest lingitud)[2]. Nimega toru saab luua käsuga `mkfifo`:

```
$ mkfifo toruke
$ ls -il
total 0
169 prw-rw-r--  1 queen queen 0 Aug  6 19:37 toruke|
  #
  # Te näete, et linkide arv on 1 ja et väljund näitab,
  # et fail on toru ('p').
  #
  # Siin saab kasutada ka käsku ln:
  #
$ ln toruke sama_toruke
$ ls -il
```

2. Kaht tüüpi torude vahel on muidki erinevusi, aga nende tutvustamine väljub käesoleva käsiraamatu raamest.

```
total 0
169 prw-rw-r--  2 queen queen 0 Aug  6 19:37 toruke|
169 prw-rw-r--  2 queen queen 0 Aug  6 19:37 sama_toruke|
$ ls -d /proc/[0-9] >toruke
  #
  # Protsess on blokeeritud, sest teises otsas pole ühtegi lugejat.
  # Protsessi peatamiseks andke käsk Control Z...
  #
[1]+  Stopped                 ls -F --show-control-chars --color=auto -d /proc/[0-9] >toruke
  #
  # ...Nüüd viige see taustale:
  #
$ bg
[1]+ ls -F --show-control-chars --color=auto -d /proc/[0-9] >toruke &
  #
  # nüüd lugege torust...
  #
$ head -5 <sama_toruke
  #
  # ...kirjutamisprotsess lõpetab
  #
/proc/1/
/proc/2/
/proc/3/
/proc/4/
/proc/5/
[1]+  Done                    ls -F --show-control-chars --color=auto -d /proc/[0-9] >toruke
$
```

Ka lugemine on blokeeriv. Kui anda ülaltoodud käsud vastupidises järjekorras, näete, et `head` on blokeeritud, oodates, et mingi protsess annaks talle midagi, mida lugeda:

```
$ head -5 <toruke
#
# Programm on blokeeritud, peatage see käsuga C-z
#
[1]+  Stopped                 head -5 <toruke
#
# Viige see taustale...
#
$ bg
[1]+ head -5 <toruke &
#
# ...Ja andke talle natuke toitu :)
#
$ ls -d /proc/[0-9] >sama_toruke
/proc/1/
/proc/2/
/proc/3/
/proc/4/
/proc/5/
[1]+  Done                    head -5 <toruke
$
```

Eelmises näites näete ka üht soovimatut efekti: käsk `ls` lõpetab enne seda, kui käsk `head` järje üle võtab. Selle tulemusena ilmub Teie ette kohe taas viip, käsk `head` aga käivitatakse hiljem ja Te näete alles pärast selle töö lõppu selle väljundit.

4.5. Spetsiaalsed failid: sümbolseadmed ja plokkseadmed

Nagu märgitud, loob selliseid faile kas süsteem või Teie masina välisseadmed. Me mainisime ka seda, et plokkseadmefailid puhverdatakse, sümbolseadmefailid aga mitte. Selle selgitamiseks sisestage seadmesse diskett ja andke kaks korda järgmine käsk:

```
$ dd if=/dev/fd0 of=/dev/null
```

Te näete järgmist: esimesel korral loeti kogu disketi sisu. Kui Te andsite käsu teist korda, ei hakatud disketti enam üldse kasutama, sest selle sisu puhverdati juba siis, kui Te andsite käsu esimest korda — ja Te ei võtnud kahe käsu vahel disketiga midagi ette.

Nüüd aga oletame, et tahate lasta trükkida suurt faili sel moel (jah, see on tõesti võimalik):

```
$ cat /suur/fail/kuskil >/dev/lp0
```

Selle käsu täitmine võtab ühepalju aega sõltumata sellest, kas annate käsu ühe, kaks või viiskümmend korda. Põhjuseks on see, et `/dev/lp0` on sümbolseade, mille sisu ei puhverdata.

Sellel, et plokkseadmed puhverdatakse, on üks vahva kõrvalmõju: ei puhverdata mitte ainult lugemisi, vaid ka kirjutamisi. See võimaldab kirjutada ketastele asünkroonselt: kui Te kettale kirjutate, ei võeta kirjutamist otsekohe ette. See sooritatakse alles siis, kui Linux otsustab ette võtta riistvarale kirjutamise. Teatud failisüsteemides saab sellise käitumise muidugi nullida: vaadake lähemalt võtmete `sync` ja `async` kirjeldust manuaalileheküljel mount(8) ning ka Sektsioon 4.7 .

Lisaks on igal spetsiaalsel failil oma ***põhinumber*** ja ***lisanumber***. Käsu `ls -l` väljundis näidatakse neid suuruse asemel, sest selliste failide korral ei ole suurusel õieti ju tähendust:

```
$ ls -l /dev/hdc /dev/lp0
brw-rw----  1 queen cdrom 22, 0 Feb 23 19:18 /dev/hdc
crw-rw----  1 root root 6, 0 Feb 23 19:17 /dev/lp0
```

Siin on `/dev/hdc` põhi- ja lisanumber vastavalt 22 ja 0, `/dev/lp0` omad aga 6 ja 0. Pange tähele, et need numbrid on igal failikategoorial unikaalsed, mis tähendab, et olemas võib olla sümbolseade põhinumbriga 22 ja lisanumbriga 0 ning ka plokkseade põhinumbriga 6 ja lisanumbriga 0. Nende numbrite funktsiooniks on võimaldada kernelil seostada antud failidega (see tähendab, välisseadmetega, millele need failid viitavad) korrektsed operatsioonid: disketiseadet ei saa käsitleda samal moel nagu näiteks SCSI kõvaketast.

4.6. Nimeviidad ja “kõvade” linkide piirangud

Nüüd tuleb meil vastu astuda ühele isegi UNIX® kasutajate seas väga levinud väärarvamusele, nagu oleks lingid, mida me seni käsitlesime (ja mida ekslikult kutsutakse “kõvadeks” linkideks), seotud ainult tavaliste failidega, mis aga pole sugugi nii — isegi nimeviidad on “lingitud”. Selleks tuleb meil aga kõigepealt selgitada, mis siis on nimeviidad ehk “pehmed” lingid (või ka “sümbolviidad”).

Nimeviidad on sellist tüüpi failid, mille ainus sisu on sõne, mis osutab kas olemasolevale või olematule failile. Kui Te kasutate nimeviita käsureal või programmis, siis Te tegelikult kasutate faili, millele see viitab, kui see on olemas. Näide:

```
$ echo Tere >minufail
$ ln -s minufail minulink
$ ls -il
total 4
169 -rw-rw-r--   1 queen  queen  6 Dec 10 21:30 minufail
416 lrwxrwxrwx   1 queen  queen  6 Dec 10 21:30 minulink -> minufail
$ cat minufail
Tere
$ cat minulink
Tere
```

Nagu näete, on faili `minulink` tüübiks `l`, mis tähendabki *Link*. Nimeviida kasutamisõigused pole eriti olulised, need on alati `rwxrwxrwx`. Te näete ka seda, et see **erineb** failist `minufail`, sest nende infosõlmenumber on erinev. Kuid see viitab just tollele failile, mistõttu käsku `cat minulink` andes lasete Te tegelikult näidata faili `minufail` sisu. Selgitamaks, et nimeviit sisaldab suvalist sõnet, võime ette võtta järgmist:

```
$ ln -s "Ma olen olematu fail" teinelink
$ ls -il teinelink
```

```
418 lrwxrwxrwx    1 queen     queen           20 Dec 10 21:43 teinelink
-> Ma olen olematu fail
$ cat teinelink
cat: teinelink: No such file or directory
$
```

Nimeviitade olemasolu mõte on selles, et nii saab vabaneda teatud piirangutest, mis on omased tavalistele ("kõvadele") linkidele:

- Te ei saa luua infosõlme linki kataloogi, mis asub teises failisüsteemis kui antud infosõlm. Põhjus on lihtne: linkide arvestaja peitub infosõlmes endas, infosõlmi aga ei saa jagada failisüsteemide vahel. Nimeviitadele pole see probleem.
- Te ei saa linkida katalooge, sest nii võivad failisüsteemis tekkida tsüklid. Kuid Te võite panna nimeviida osutama kataloogile ja kasutada seda nii, nagu oleks ka tegelikult tegemist kataloogiga.

Niisiis on nimeviidad mitmel juhul väga kasulikud ning sageli kiputakse sel moel linkima faile ka juhul, kui võiks kasutada tavalist linki. Tavalise lingi eeliseks on samas see, et Te ei kaota faili, kui kustutate "originaali".

Ja lõpuks: kui Te jälgisite meie arutelu tähelepanelikult, on lihtne aru saada, et nimeviit on väga väike fail - selle suurus on kõigest vastava sõne suurus.

4.7. Failiatribuudid

Failiatribuute kasutavad nii FAT (arhiiv, süsteemne fail, nähtamatu, kirjutuskaitstud) kui ka GNU/Linuxi failisüsteemid, kuid viimase atribuudid on oma nägu. Me käsitleme neid siin üsna lühidalt, sest enamasti ei ole neil suuremat tähtsust. Kui Te aga soovite tõesti oma süsteemist kõike teada, et seda maksimaalselt turvaliselt hallata, siis tasub ka seda teada.

Failiatribuutidega tegeleb kaks käsku: `lsattr` ja `chattr`. Nagu vahest juba arvasite, näitab ("LiSts") `lsattr` atribuute, `chattr` aga muudab ("CHanges") neid. Atribuute saab määrata ainult kataloogidele ja tavalistele failidele. Atribuudid on järgmised (nende täieliku nimekirja annab chattr(1)):

1. `A` ("no **A**ccess time" ehk 'ilma kasutamisajata'): kui failil või kataloogil on selline atribuut, siis ei uuendata selle viimase kasutamise aega sellele vaatamata, kas seda pärast näidatavat aega loetakse või kirjutatakse. See võib olla kasulik näiteks failide või kataloogide puhul, mida väga sageli loetakse, sest see parameeter on ainus, mis infosõlmes muutub, kui see on lugemiseks avatud.
2. `a` ("**a**ppend only" ehk 'ainult lisamine'): kui sellise atribuudiga fail avatakse kirjutamiseks, on ainus võimalus seda muuta lisada andmeid varasemale sisule. Kataloogi puhul tähendab see, et sinna saab faile ainult lisada, mitte aga olemasolevaid ümber nimetada või kustutada. Sellist atribuuti saab määrata või eemaldada ainult administraator (`root`).
3. `d` ("no **d**ump" ehk 'tõmmistusega'): `dump` on UNIX® standardne varukoopiate loomise utiliit. See teeb tõmmise igast failisüsteemist, mille tõmmisearvuks on failis `/etc/fstab` (vt. Peatükk 6) määratud 1. Kui aga failil või kataloogil on selline atribuut, siis ei arvestata sellega tõmmistamisel. Pange tähele, et kataloogide korral puudutab see ka kõiki selles asuvaid alamkatalooge ja faile.
4. `i` ("**i**mmutable" ehk 'mittemuudetav'): sellise atribuudiga faili või kataloogi ei saa üldse kuidagi muuta: seda ei saa ümber nimetada, sellele ei saa luua uusi linke[3] ja seda ei saa eemaldada. Seda atribuuti saab määrata või eemaldada ainult administraator (`root`). Arvestage, et see takistab ka muutmast kasutamisaega, mistõttu ei ole vaja määrata atribuuti `A`, kui atribuut `i` on juba määratud.
5. `s` ("**s**ecure deletion" 'turvaline kustutamine'): kui sellie atribuudiga fail või kataloog kustutatakse, kirjutatakse plokid, mida see kettal hõlmas, üle nullidega.
6. `S` ("**S**ynchronous mode" ehk 'sünkroonne režiim'): kõik sellise atribuudiga faili või kataloogi tehtud muudatused on sünkroonsed ja kirjutatakse otsekohe kettale.

Nii näiteks võib olla mõttekas anda olulistele süsteemsetele failidele atribuut `i`, et vältida igasuguseid halbu üllatusi. Manuaalilehekülgedele võib olla mõttekas anda atribuut `A`, mis väldib päris paljusid kettaoperatsioone ning võib eriti olla abiks sülearvutite akude toimeea pikenemisel.

3. Tehke endale kindlasti selgeks, mida "lingi lisamine" tähendab nii faili kui kataloogi puhul!

Peatükk 5. /proc failisüsteem

/proc failisüsteem on GNU/Linuxile spetsiifiline nähtus. See on virtuaalne failisüsteem, mistõttu selles kataloogis leiduvad failid ei võta Teie kõvakettal tegelikult üldse ruumi. See on väga mugav viis hankida infot süsteemi kohta, sest enamik antud kataloogi faile on ka tavalisele inimesele päris arusaadavad (nojah, vahest küll väikese abiga). Õigupoolest hangivad paljud programmid infot kataloogis /proc leiduvatest failidest. Sellised on näiteks programmid, mis näitavad infot protsesside kohta (top, ps ja muud sellised). /proc on ka päris hea allikas info leidmiseks oma riistvara kohta ja mitmed programmid ongi tegelikult kataloogis /proc leiduva info liidesed.

Seal paikneb ka spetsiaalne alamkataloog /proc/sys. See võimaldab Teil näha kerneli parameetreid ja neid muuta, kusjuures muutused rakenduvad otsekohe.

5.1. Protsesside info

Kui Te uurite kataloogi /proc sisu, näete seal paljusid katalooge, mille nimeks on mingi arv. Need kataloogid sisaldavad infot parajasti Teie süsteemis töötavate protsesside kohta:

```
$ ls -d /proc/[0-9]*
/proc/1/    /proc/302/  /proc/451/  /proc/496/  /proc/556/  /proc/633/
/proc/127/  /proc/317/  /proc/452/  /proc/497/  /proc/557/  /proc/718/
/proc/2/    /proc/339/  /proc/453/  /proc/5/    /proc/558/  /proc/755/
/proc/250/  /proc/385/  /proc/454/  /proc/501/  /proc/559/  /proc/760/
/proc/260/  /proc/4/    /proc/455/  /proc/504/  /proc/565/  /proc/761/
/proc/275/  /proc/402/  /proc/463/  /proc/505/  /proc/569/  /proc/769/
/proc/290/  /proc/433/  /proc/487/  /proc/509/  /proc/594/  /proc/774/
/proc/3/    /proc/450/  /proc/491/  /proc/554/  /proc/595/
```

Arvestage, et kasutajana saate (täiesti loogiliselt) näha ainult enda, mitte aga teiste kasutajate protsesside infot. Sestap logige sisse administraatorina (root) ja vaadake, milline info käib protsessi 1 kohta, milleks on init, mis vastutab kõigi teiste protsesside käivitamise eest:

```
$ su
Password:
# cd /proc/1
# ls -l
total 0
-r--------  1 root root 0 Aug 15 18:14 auxv
-r--r--r--  1 root root 0 Aug 15 18:14 cmdline
lrwxrwxrwx  1 root root 0 Aug 15 18:14 cwd -> //
-r--------  1 root root 0 Aug 15 18:14 environ
lrwxrwxrwx  1 root root 0 Aug 15 18:14 exe -> /sbin/init*
dr-x------  2 root root 0 Aug 15 18:14 fd/
-rw-r--r--  1 root root 0 Aug 15 18:14 loginuid
-r--r--r--  1 root root 0 Aug 15 18:14 maps
-rw-------  1 root root 0 Aug 15 18:14 mem
-r--r--r--  1 root root 0 Aug 15 18:14 mounts
-rw-r--r--  1 root root 0 Aug 15 18:14 oom_adj
-r--r--r--  1 root root 0 Aug 15 18:14 oom_score
lrwxrwxrwx  1 root root 0 Aug 15 18:14 root -> //
-rw-------  1 root root 0 Aug 15 18:14 seccomp
-r--r--r--  1 root root 0 Aug 15 18:14 stat
-r--r--r--  1 root root 0 Aug 15 18:14 statm
-r--r--r--  1 root root 0 Aug 15 18:14 status
dr-xr-xr-x  3 root root 0 Aug 15 18:14 task/
-r--r--r--  1 root root 0 Aug 15 18:14 wchan
#
```

Kõigis kataloogides on ühesugused kirjed. Tutvustame siin lühidalt neist mõningaid:

1. cmdline: see (pseudo)fail sisaldab kogu protsessi välja kutsunud käsku. See ei ole vormindatud: programmi ja argumentide vahel pole tühikuid, ka puuduvad reavahetused. Selle vaatamiseks tuleks kasutada käsku `perl -ple 's,\00, ,g' cmdline`.
2. cwd: see nimeviit osutab protsessi aktiivsele töökataloogile (inglise keeles 'current working directory', millest ka lühend).

3. `environ`: see fail sisaldab kõiki antud protsessile määratud keskkonnamuutujaid kujul `MUUTUJA=väärtus`. Sarnaselt kirjele `cmdline` ei ole väljund vähimalgi määral vormindatud: muutujaid ei eralda reavahetused, reavahetust pole ka lõpus. Selle vaatamiseks sobib käsk `perl -ple 's,\00,\n,g' environ`.
4. `exe`: see nimeviit osutab käivitatavale failile, mis vastab töötavale protsessile.
5. `fd`: selles alamkataloogis leiduvad protsessi poolt parajasti avatud failide deskriptorid (vt. allpool).
6. `maps`: selle nimega toru sisu näidata lastes (näiteks käsuga `cat`) näete parajasti failiga seostatud protsessi aadressiruumi osasid. Väljad on paremalt vasakule järgmised: seosega seotud aadressiruum, seose õigused, suhtaadress (faili algusest), kus seos algab, seadme põhi- ja lisanumber (16-süsteemis), millel seotud fail asub, faili infosõlmenumber ning faili enda nimi. Kui seade on 0 ning infosõlmenumber või failinimi puuduvad, on tegemist anonüümse seosega. Vaadake mmap(2).
7. `root`: see nimeviit osutab protsessi juurkataloogile. Tavaliselt on see `/`, aga vaadake chroot(2).
8. `status`: see fail sisaldab mitmesugust infot protsessi kohta: käivitatava faili nimi, praegune olek, PID ja PPID, reaalne ja toimiv UID ja GID, mälukasutus ja nii edasi. Pange tähele, et failid `stat` ja `statm` on iganenud: neis sisaldunud infot salvestatakse nüüd failis `status`.

Kui me laseme näidata suvalise protsessi kataloogi `fd` sisu, on tulemus umbes selline:

```
# ls -l /proc/8141/fd/
total 4
lrwx------  1 peter peter 64 Aug  4 09:05 0 -> /dev/tty1
lrwx------  1 peter peter 64 Aug  4 09:05 1 -> /dev/tty1
lrwx------  1 peter peter 64 Aug  4 09:05 2 -> /dev/tty1
l-wx------  1 peter peter 64 Aug  4 09:05 3 -> /home/peter/seti32/lock.sah
#
```

Siin on tegemist protsessi avatud failide deskriptoritega. Iga avatud deksriptorit näitab nimeviit, mille nimeks on deskriptori number ja mis osutab antud deskriptori avatud failile[1]. Pange tähele nimeviitade õigusi: see on ainus koht, kus neil on mingi mõte, sest nad tähistavad õigusi, millega deskriptorile vastav fail avati.

5.2. Riistvara info

Lisaks erinevate protsessidega seotud kataloogidele sisaldab `/proc` veel tohutu hulga infot Teie masina riistvara kohta. Kataloogi `/proc` faile näidata lastes näete midagi sellist:

```
$ ls -d [a-z]*
acpi/      diskstats    iomem      locks       pci            sysvipc/
asound/    dma          ioports    mdstat      scsi/          tty/
buddyinfo  driver/      irq/       meminfo     self@          uptime
bus/       execdomains  kallsyms   misc        slabinfo       version
cmdline    fb           kcore      modules     splash         vmstat
config.gz  filesystems  keys       mounts@     stat
cpuinfo    fs/          key-users  mtrr        swaps
crypto     ide/         kmsg       net/        sys/
devices    interrupts   loadavg    partitions  sysrq-trigger
$
```

Kui nüüd vaadata näiteks `/proc/interrupts` sisu, siis see sisaldab süsteemis parajasti kasutatavate katkestuste nimekirja koos neid kasutavate välisseadmetega. Kataloog `ioports` sisaldab parajasti tegevate sisend/väljundaadressi vahemike nimekirja, `dma` aga sedasama DMA kanalite kohta. Niisiis tasuks konflikti leidmiseks uurida nende kolme faili sisu:

```
$ cat interrupts
           CPU0
  0:     543488          XT-PIC  timer
  2:          0          XT-PIC  cascade
  5:        109          XT-PIC  ohci_hcd:usb2, eth1
  7:          1          XT-PIC  parport0
  8:          0          XT-PIC  rtc
  9:       3432          XT-PIC  acpi, NVidia CK8
 10:      52855          XT-PIC  ehci_hcd:usb3, eth0
 11:       7538          XT-PIC  libata, ohci_hcd:usb1
 12:       1386          XT-PIC  i8042
```

1. Kui Te meenutate, millest rääkis Sektsioon 7.4, peaksite teadma, mida tähendavad deskriptorid `0`, `1` ja `2`. Deskriptor `0` on standardsisend, `1` on standardväljund ja `2` on standardveaväljund.

```
 14:         20          XT-PIC  ide0
 15:       5908          XT-PIC  ide1
NMI:          0
LOC:          0
ERR:          0
MIS:          0
```

```
$ cat ioports
0000-001f : dma1
0020-0021 : pic1
0040-0043 : timer0
0050-0053 : timer1
0060-006f : keyboard
0070-0077 : rtc
0080-008f : dma page reg
00a0-00a1 : pic2
00c0-00df : dma2
00f0-00ff : fpu
0170-0177 : ide1
01f0-01f7 : ide0
0376-0376 : ide1
0378-037a : parport0
037b-037f : parport0
03c0-03df : vesafb
03f6-03f6 : ide0
03f8-03ff : serial
0778-077a : parport0
0970-0977 : 0000:00:0b.0
0970-0977 : sata_nv
09f0-09f7 : 0000:00:0b.0
09f0-09f7 : sata_nv
0b70-0b73 : 0000:00:0b.0
0b70-0b73 : sata_nv
0bf0-0bf3 : 0000:00:0b.0
0bf0-0bf3 : sata_nv
0cf8-0cff : PCI conf1
4000-407f : motherboard
4000-4003 : PM1a_EVT_BLK
4004-4005 : PM1a_CNT_BLK
4008-400b : PM_TMR
4020-4027 : GPE0_BLK
4080-40ff : motherboard
4080-40ff : pnp 00:00
4200-427f : motherboard
4200-427f : pnp 00:00
4280-42ff : motherboard
4280-42ff : pnp 00:00
4400-447f : motherboard
4400-447f : pnp 00:00
4480-44ff : motherboard
44a0-44af : GPE1_BLK
5000-503f : motherboard
5000-503f : pnp 00:01
5100-513f : motherboard
5100-513f : pnp 00:01
9000-9fff : PCI Bus #02
9000-907f : 0000:02:07.0
9000-907f : 0000:02:07.0
ac00-ac0f : 0000:00:0b.0
ac00-ac0f : sata_nv
b000-b07f : 0000:00:0b.0
b000-b07f : sata_nv
b800-b8ff : 0000:00:06.0
b800-b8ff : NVidia CK8
bc00-bc7f : 0000:00:06.0
bc00-bc7f : NVidia CK8
c000-c007 : 0000:00:04.0
c000-c007 : forcedeth
c400-c41f : 0000:00:01.1
f000-f00f : 0000:00:09.0
f000-f007 : ide0
f008-f00f : ide1
```

```
$cat dma
 3: parport0
```

```
 4: cascade
$
```

Veel lihtsam on kasutada käsku `lsdev`, mis kogub neist failidest info ja sordib selle välisseadmete kaupa, mis on kahtlemata märksa mugavam[2]:

```
$ lsdev
Device            DMA   IRQ  I/O Ports
------------------------------------------------
0000:00:01.1                 c400-c41f
0000:00:04.0                 c000-c007
0000:00:06.0                 b800-b8ff bc00-bc7f
0000:00:09.0                 f000-f00f
0000:00:0b.0                 0970-0977 09f0-09f7 0b70-0b73 0bf0-0bf3 ac00-ac0f b000-b07f
0000:02:07.0                 9000-907f     9000-907f
cascade             4     2
CK8                          9
dma                          0080-008f
dma1                         0000-001f
dma2                         00c0-00df
eth0                        10
eth1                         5
forcedeth                    c000-c007
fpu                          00f0-00ff
GPE0_BLK                     4020-4027
GPE1_BLK                     44a0-44af
i8042                       12
ide0                        14  01f0-01f7 03f6-03f6   f000-f007
ide1                        15  0170-0177 0376-0376   f008-f00f
keyboard                     0060-006f
motherboard                  4000-407f 4080-40ff 4200-427f 4280-42ff 4400-447f 4480-44ff 5000-503f 5100-513f
NVidia                       b800-b8ff   bc00-bc7f
ohci_hcd:usb1               11
parport0            3        7  0378-037a 037b-037f 0778-077a
PCI                          0cf8-0cff 9000-9fff
pic1                         0020-0021
pic2                         00a0-00a1
PM1a_CNT_BLK                 4004-4005
PM1a_EVT_BLK                 4000-4003
PM_TMR                       4008-400b
pnp                          4080-40ff   4200-427f   4280-42ff   4400-447f   5000-503f   5100-513f
rtc                          8  0070-0077
sata_nv                      0970-0977   09f0-09f7   0b70-0b73   0bf0-0bf3   ac00-ac0f   b000-b07f
serial                       03f8-03ff
timer                        0
timer0                       0040-0043
timer1                       0050-0053
vesafb                       03c0-03df
$
```

Failide täielik nimekiri oleks liiga pikk, aga kirjeldame siiski mõnda neist:

- `cpuinfo`: nagu nimigi ütleb, sisaldab see fail infot Teie masina protsessori(te) kohta.
- `modules`: see fail sisaldab parajasti kerneli kasutatavate moodulite nimekirja koos infoga nende kasutamise kohta. Õigupoolest on see info, mida märksa loetavamal kujul esitab käsk `lsmod`.
- `meminfo`: see fail sisaldab infot mälukasutuse kohta sel hetkel, mil soovisite seda infot näha. Käsk `free` näitab sama infot loetavamal kujul.
- `apm`: kui Teil on sülearvuti, näitab selle faili sisu Teie arvuti aku olekut. Te näete, kas AC on ühendatud, aku täituvust, seda, kas Teie sülearvuti BIOS toetab APM-i (paraku pole see kaugeltki nii kõigi sülearvutitega), aku järelejäänud tööaega minutites jne. Fail ei ole just väga loetavas vorminduses, mispärast oleks mõttekas kasutada vahetult käsku `apm`, mis esitab sama info märksa loetavamal kujul.

  Pange tähele, et tänapäeva arvutid pakuvad APM-i asemel ACPI toetust. Vaadake allpool.
- `bus`: see alamkataloog sisaldab infot kõigi välisseadmete kohta, mis leitakse Teie masina siinidel. Tavaliselt ei ole info selles failis kuigi hästi vormindatud ning märksa mõttekam oleks see loetavamaks muuta mõne muu utiliidiga: `lspcidrake`, `lspnp` jne.

2. `lsdev` kuulub tarkvarapaketi `procinfo` koosseisu.

- `acpi`: selle kataloogi failid ja kataloogid peaksid huvi pakkuma eriti sülearvutite korral, kus saab valida mitme voolusäästmisrežiimi vahel. Arvestage, et selliseid asju on hõlpsam muuta mõne kõrgema taseme rakendusega, näiteks sellistega, mis kuuluvad tarkvarapaketti `acpid` või `kapacity`.

  Huvipakkuvaimad kirjed on järgmised:

  `battery`

  Näitab, mitu akut sülearvutil on, ning muud asjakohast infot (jäänud tööaeg, maksimaalne mahutavus jne.).

  `button`

  Võimaldab juhtida "spetsiaalsete" nuppudega seotud toiminguid (power, sleep, lid jne.).

  `fan`

  Näitab ventilaatorite olekut, seda, kas need töötavad või mitte, ning võimaldab neid teatud kriteeriumi alusel käivitada/peatada. See, mil määral saab masina ventilaatorite tööd juhtida, sõltub emaplaadist.

  `processor`

  Siin on üks alamkataloog iga masinas leiduva CPU kohta. Juhtimisvõimalused sõltuvad protsessorist. Mobiilprotsessoritel on enamasti rohkem võimalusi, sealhulgas:

  - võimalus kasutada mitut vooluolekut ning leida õige jõudluse ja voolutarbe vahekord
  - võimalus muuta arvuti taktsagedust, mis aitab kahandada CPU voolutarvet

  Arvestage, et mõnedki protsessorid selliseid võimalusi ei paku.

  `thermal_zone`

  Info süsteemi/protsessori temperatuuri kohta.

5.3. Kerneli parameetrite vaatamine ja muutmine

Alamkataloogi `/proc/sys` ülesanne on anda teada kerneli erinevaid parameetreid ning võimaldada Teil neist mõningaid interaktiivselt muuta. Erinevalt muudest kataloogi `/proc` failidest, saab mõningatesse selle kataloogi failidesse ka kirjutada, kuid see õigus on ainult administraatoril (`root`).

Kõigi kataloogide ja failide kirjeldus oleks liiga pikk, pealegi sõltub kataloogide sisu konkreetsest süsteemist ning suur osa faile käib ainult teatud spetsiaalsete rakenduste kohta. Siiski mainime ära kaks levinumat asja, milleks see alamkataloog hea on:

1. Marsruutimise lubamine: kuigi Mandriva Linuxi vaikekernel on marsruutimiseks valmis, tuleb Teil see spetsiaalselt lubada. Selleks tuleb Teil anda administraatorina (`root`) järgmine käsk:

   ```
   $ echo 1 >/proc/sys/net/ipv4/ip_forward
   ```

   Asendage 1 0-ga, kui soovite keelata marsruutimise.

2. IP võltsimise vältimine: IP võltsimisega üritatakse Teid uskuma panna, et kuskilt mujalt saabunud pakett on tegelikult pärit sellelt liideselt, kuhu see saabus. Seda võtet kasutavad väga sageli ***kräkkerid***[3]. Te võite aga lasta kernelil sellised katsed tagasi lüüa. Andke käsk:

   ```
   $ echo 1 >/proc/sys/net/ipv4/conf/all/rp_filter
   ```

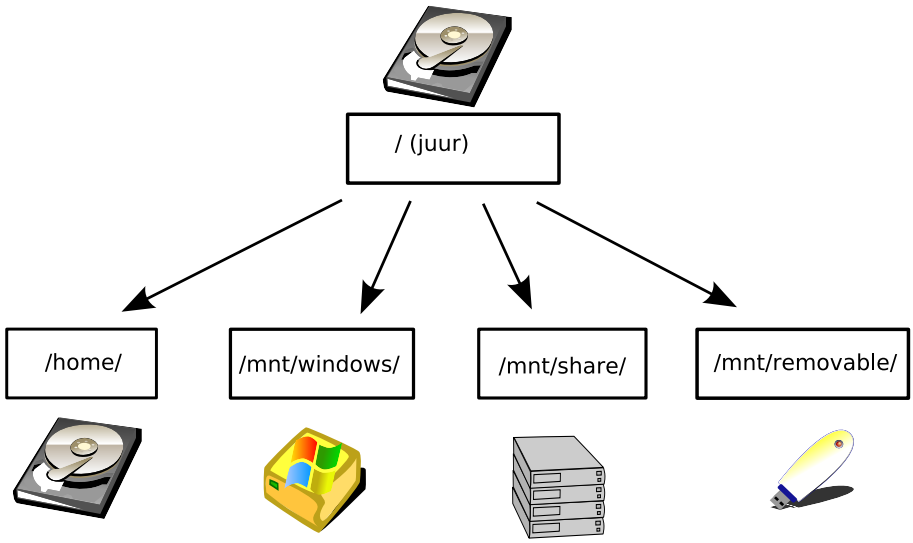
   ja sellist laadi rünnak muutub võimatuks.

Need muudatused toimivad ainult seni, kuni süsteem töötab. Kui teha süsteemile taaskäivitus, hakkavad taas kehtima vaikeväärtused. Et väärtused juba käivitamise ajal võtaks mingi muu väärtuse, tuleb Teil need käsud, mida Te eespool toodud näidete põhjal käsureal andsite, lisada faili `/etc/rc.d/rc.local`. Siis ei pea Te neid iga taaskäivituse ajal jälle ise käsitsi andma. Teine võimalus on muuta faili `/etc/sysctl.conf` (vt. sysctl.conf(5) ja sysctl(8)).

3. Aga mitte ***häkkerid***!

Peatükk 6. Failisüsteemid ja haakepunktid

Nagu selgitas Peatükk 2, on kõik failid süsteemis korraldatud ühte puukujulisse struktuuri. Ka siis, kui me soovime kasutada eemaldatavat seadet, näiteks CD-ROM, või mõnda võrgus paiknevat asukohta, näiteks failiserverit, "poogitakse" selle sisu sõna otseses mõttes kuhugi puusse.



Joonis 6-1. Haakepunktid

Seda näitab Joonis 6-1: juurpartitsioon, mis koosneb GNU/Linuxi partitsioonist, millel asub veel üks Linuxi partitsioon `/home/`, kuid ka Windows® partitsioon, failiserveri jagatud ressurss (kas Windows® või UNIX®) ning USB mälupulk. Tänapäeval võib GNU/Linuxi failisüsteemi haakida paljusid seadmeid, sealhulgas peaaegu kõiki olemasolevaid failisüsteeme, WebDAV-i ja isegi selliseid eksootilisi asju, nagu Google™ Mail...

Kõige paremini saab pealkirjas esitatud kontseptsioone mõista näite varal. Oletame, et hankisite just uhiuue kõvaketta, millel pole mingeid partitsioone. Teie Mandriva Linuxi partitsioon on pungil täis ning selle asemel, et alustada kõike otsast peale, soovite terve puustruktuuri sektsiooni[1] viia uuele kõvakettale. Et uus kõvaketas on palju mahukam, valite selleks välja oma suurima kataloogi: `/usr`.

Me kasutame seda näidet läbi kogu Sektsioon 6.2, aga kõigepealt veidi teooriat.

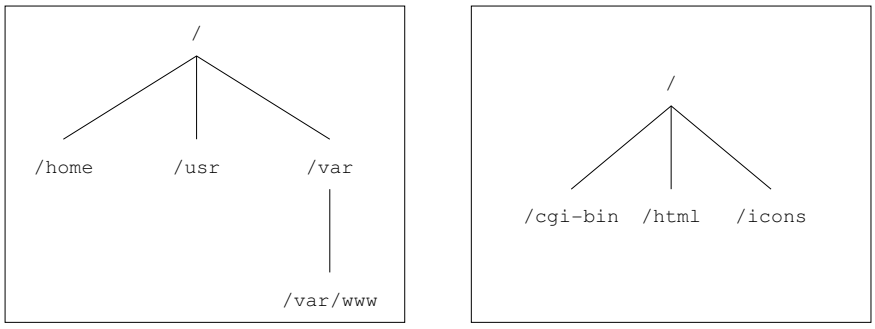
6.1. Põhimõtted

Kõiki kõvakettaid saab jagada partitsioonideks, millel kõigil on oma failisüsteem. Kui Windows® omistab kõigile failisüsteemidele (hmm, tegelikult ainult neile, mida ta ära tunneb) tähe, siis GNU/Linux kasutab unikaalset failipuud ning iga failisüsteem ***haagitakse*** selles puus teatud konkreetsesse kohta.

Kui Windows® vajab `C:` ketast, siis GNU/Linux peab suutma haakida oma failipuu juure (`/`) partitsioonil, mis sisaldab ***juurfailisüsteemi***. Kui juur on haagitud, saate haakida teised puustruktuuri failisüsteemid puu vastavates ***haakepunktides***. Kõik juure all asuvad kataloogid võivad olla haakepunktiks ning Te võite üht ja sama failisüsteemi haakida mitmel korral erinevates haakepunktides.

See võimaldab väga suurt paindlikkust. Kui Te näiteks kavatsete seadistada veebiserveri, on üsna tavaline anda terve partitsioon kataloogile, mis sisaldab veebiserveri andmeid. Andmeid sisaldav kataloog on tavaliselt `/var/www`, mis sellisel juhul ongi antud partitsiooni haakepunkt. Mõttekas oleks luua ka suur partitsioon `/home`, kui kavatsete alla laadida palju tarkvara, salvestada suurel hulgal töö- või eradokumente ja pilte, muusikafaile jne. Seda, kuidas näeb süsteem välja enne ja pärast failisüsteemi haakimist, näitavad Joonis 6-2 ja Joonis 6-3.

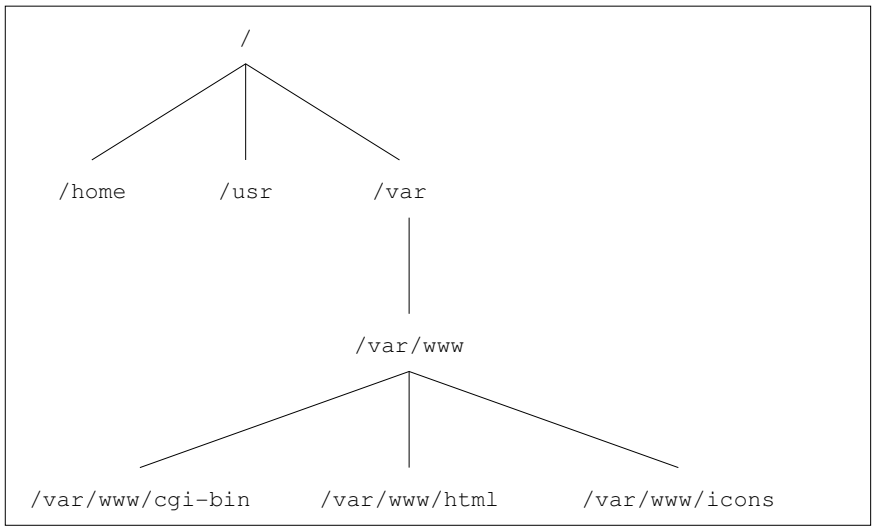
1. Me eeldame siinkohal, et kogu puu asub ühel partitsioonil.



Juurfailisüsteem
(juba haagitud)

Failisüsteem, mis sisaldab
kataloogi "/var/www" faile
(veel haakimata)

Joonis 6-2. Veel haakimata failisüsteem



Joonis 6-3. Haagitud failisüsteem

Nagu arvata võite, on sellel mitmeid eeliseid: puustruktuur on alati ühesugune sõltumata sellest, kas tegemist on ühe failisüsteemiga või ka mitmekümnega. Selline paindlikkus lubab Teil viia näiteks ruuminappuse tekkides puustruktuuri ka olulisi osasid muule partitsioonile, mida me alljärgnevalt demonstreerimegi.

Haakepunktide kohta tasub meelde jätta kaks olulist asja:

1. Haakepunktina toimiv kataloog peab olema olemas.
2. See kataloog peaks **soovitatavalt olema tühi**: kui haakepunktina kasutatavas kataloogis on juba faile ja alamkatalooge, jäävad nad uue failisüsteemi haakimisel lihtsalt "peidetuks". Faile küll ei kustutata, aga neid ei saa kasutada, enne kui Te pole haakepunkti vabastanud.

Õigupoolest on küll võimalik ka uues haagitud failisüsteemis "peidetud" andmetele ligi pääseda. Selleks tuleb lihtsalt peidetud kataloog haakida võtmega `--bind`. Kui Te näiteks haakisite äsja kataloogi asukohas `/peidetud/kataloog` ja tahate selle algsele sisule ligi pääseda uues failisüsteemi `/uus/kataloog`, tuleb Teil anda käsk:

```
mount --bind /peidetud/kataloog/ /uus/kataloog
```

6.2. Kõvaketta jagamine, partitsiooni vormindamine

Käesolevat osa lugedes tuleb meeles pidada kaht asja: kõvaketas on jagatud partitsioonideks ja igal partitsioonil asub failisüsteem. Teie uhiuuel kõvakettal pole ei partitsioone ega failisüsteeme, seepärast alustame kõvaketta jagamisega. Selleks tuleb Teil sisse logida administraatorina (`root`).

Kõigepealt tuleb teada kõvaketta "nime" (s.t. millise failina seda süsteemis tuntakse). Oletame, et uus ketas on paigaldatud Teie esmase IDE liidese alamana. Sellisel juhul on ketta nimeks `/dev/hdb`[2]. Palun vaadake *Põhiteadmiste käsiraamat*us *Partitsioonide haldamine*, kus selgitatakse ketta jagamist. DiskDrake loob ka Teie eest failisüsteemid, nii et kui olete ketta jagamise ja failisüsteemide loomisega valmis, liigume edasi.

6.3. Käsud mount ja umount

Nüüd, kus failisüsteem on loodud, on aeg partitsioon haakida. Algul on see tühi, sest süsteem ei ole veel failisüsteemile ligi pääsenud, et sinna faile lisada. Failisüsteeme haagitakse käsuga `mount` ja selle süntaks on järgmine:

```
mount [võtmed] <-t tüüp> [-o haakimisvõtmed] <seade> <haakepunkt>
```

Antud juhul soovime ajutiselt haakida oma partitsiooni asukohas `/mnt/uus` (või mis tahes muus Teile sobivas haakepunktis - peaasi, et see oleks ikka olemas). Meie uue partitsiooni saab haakida käsuga:

```
$ mount -t ext3 /dev/hdb1 /mnt/uus
```

Võtmega `-t` määratakse, milline failisüsteem partitsioonil asub. Levinumad failisüsteemid on ext2FS (GNU/Linuxi failisüsteem) ja ext3FS (ext2FS täiustatud versioon kirjendamisvõimalustega), VFAT (peaaegu kõik DOS/Windows® partitsioonid: FAT 12, 16 või 32), NTFS (uuemad Windows® versioonid) ja ISO9660 (CD-ROM-i failisüsteem). Kui Te tüüpi ei määra, üritab `mount` partitsiooni failisüsteemi ise kindlaks määrata selle superploki alusel.

Võtmega `-o` saab määrata mõne haakimisvõtme. Need erinevad mõneti vastavalt failisüsteemile. Täpsemalt tasuks uurida manuaalilehekülge mount(8).

Nüüd, kus uus partitsioon on haagitud, on aeg kopeerida sinna kogu kataloog `/usr`:

```
$ (cd /usr && tar cf - .) | (cd /mnt/uus && tar xpvf -)
```

Kui failid on kopeeritud, võib partitsiooni lahutada. Selleks kasutage käsku `umount`. Selle süntaks on lihtne:

```
umount <haakepunkt|seade>
```

Oma uue partitsiooni lahutamiseks tuleb niisiis anda käsk:

```
$ umount /mnt/uus
```

või:

```
$ umount /dev/hdb1
```

Vahel võib juhtuda, et seade (eriti CD-ROM) on tegevuses. Enamik kasutajaid kipub selle puhul eelistama arvutit taaskäivitada. Kui näiteks käsk `umount /dev/hdc` ebaõnnestub, võiks proovida kasutada "laiska" `umount`'i. Ka selle süntaks on lihtne:

```
umount -l <haakepunkt|seade>
```

See käsk lahutab seadme ja sulgeb kõik seadmega seotud avatud toimingud, kui vähegi võimalik. Tavaliselt saab ketta väljastada käsuga `eject <haakepunkt|seade>`. Kui aga `eject` ei toimi ja Te ei soovi arvutit taaskäivitada, proovige laiska lahutamist.

Kuna me soovime oma partitsioonist "teha" enda uue `/usr`-kataloogi, peame selle soovi ka süsteemile teatavals tegema. Selleks tuleb redigeerida faili `/etc/fstab`. See võimaldab automatiseerida teatud failisüsteemide

2. Ketta nime määramist selgitab Sektsioon 2.2.

haakimist eriti süsteemi käivitamise ajal. Selles on mitu rida, mis kirjeldavad failisüsteeme, nende haakepunkte ja muid valikuid. See fail võib välja näha näiteks selline:

```
/dev/hda2 / ext3 defaults 1 1
/dev/hdd /mnt/cdrom auto umask=0022,user,iocharset=utf8,noauto,ro,exec,users 0 0
/dev/fd0 /mnt/floppy supermount dev=/dev/fd0,fs=ext2:vfat,--,umask=0022,iocharset=utf8,sync 0 0
/dev/hda1 /mnt/windows ntfs umask=0,nls=utf8,ro 0 0
none /proc proc defaults 0 0
/dev/hda3 swap swap defaults 0 0
```

Igal real seisab:

- failisüsteemi sisaldav seade
- haakepunkt
- failisüsteemi tüüp
- haakimisvõtmed
- utiliidi `dump` varundamise ***lipp***
- `fsck` (*FileSystem ChecK* ehk failisüsteemi kontroll) kontrollimisjärjekord

Alati on olemas juurfailisüsteemi kirje. saaleala partitsioonid on spetsiaalsed, sest neid ei ole näha puustruktuuris. Nende partitsioonide haakepunktiväljal esineb võtmesõna `swap`. Failisüsteemi `/proc` kirjeldab põhjalikumalt Peatükk 5. Veel üks spetsiaalne failisüsteem on `/dev/pts`.

Arvestage, et Teie süsteem võib sellesse faili kirjeid atuomaatselt lisada ja neid sealt eemaldada. Seda teeb käsk `fstab-sync`, mis saab spetsiaalseid sündmusi riistvara abstraktsioonikihilt (HAL) ning töötleb vastavalt faili `/etc/fstab`. Täpsemalt räägib sellest manuaalilehekülg fstab-sync(8).

Naastes oma failisüsteemi vahetamise juurde, oleme praeguseks hetkeks viinud kogu oma `/usr` hierarhia partitsioonile `/dev/hdb1` ja tahame nüüd, et see partitsioon haagitaks süsteemi käivitamisel asukohas `/usr`. Selleks tuleb lisada faili `/etc/fstab` järgmine kirje:

```
/dev/hdb1 /usr ext3 defaults 1 2
```

Nüüd haagitakse partitsioon iga süsteemi käivitamise ajal ja vajaduse korral kontrollitakse, ega sellel vigu esine.

Kui Teie partitsiooni tüübiks ei ole ext3FS, tuleb Teil mõistagi anda õige tüüp. Tavalisemad võimalused on siin `ext2` ja `reiserfs`. Pange tähele, et viimase välja väärtuseks on `2`. See tähendab, et seda kontrollitakse pärast kirjeid, mille väärtuseks on `1`, ja pärast teisi sama prioriteediga failisüsteeme samal kõvakettal, mis seisavad failis `/etc/fstab` eespool. Ainult juurpartitsioonil (`/`) tohib olla väärtus `1`.

On olemas kaks spetsiaalset võtit: `noauto` ja `users`. Võti `noauto` tähendab, et failisüsteemi ei haagita süsteemi käivitamise ajal, vaid alles siis, kui selleks vajalik korraldus antakse. Võti `users` määrab, et failisüsteemi võib haakida ja lahutada iga kasutaja. Mõlemat võtit tarvitatakse tavaliselt CD-ROM-i ja disketiseadme puhul. Võtmeid on veelgi, neist annab ülevaate `/etc/fstab` `man`-lehekülg (fstab(5)).

Üks faili `/etc/fstab` kasutamise eeliseid on see, et nii lihtsustub tublisti käsu `mount` süntaks. Failis kirjeldatud failisüsteemi haakimiseks võib anda viite haakepunktile või seadmele. Disketi haakimiseks näiteks andke käsk:

```
$ mount /mnt/floppy
```

või:

```
$ mount /dev/fd0
```

Oma partitsiooninäite lõpetuseks vaatame üle, mida me oleme ära teinud. Me kopeerisime `/usr` hierarhia ja muutsime faili `/etc/fstab`, et uus partitsioon haagitaks kohe süsteemi käivitamisel. Praegu on siiski veel vanad `/usr` failid kettal oma endises asukohas ja kui me tahame ruumi juurde saada - see meid ju kõike kirjeldatut ette võtma panigi -, tuleb need kustutada.

- Selleks tuleb kõigepealt lülituda ainukasutaja režiimi, andes käsureal korralduse `telinit1`. See peatab kõik teenused ja takistab kasutajatel masinaga ühendust loomast.
- Seejärel kustutame kõik kataloogi `/usr` failid. Pange tähele, et siin käib jutt ikka veel "vanast" kataloogist, sest uus ja suurem pole ju veel haagitud. `rm -Rf /usr/*`.
- Lõpuks haagime uue `/usr`-kataloogi: `mount /usr`.

Ja ongi kõik. Nüüd naaske mitmekasutajarežiimi (`telinit3` tavalise tekstirežiimi või `telinit5` graafilise režiimi korral) ja kui Teil rohkem haldusülesandeid ees ei ole, võite rahumeeli administraatori (`root`) kontolt välja logida.

Peatükk 7. Sissejuhatus käsurea kasutamisse

Eelnevalt selgitas Peatükk 1, kuidas käivitada shell. Käesolevas peatükis selgitame, kuidas sellega tööd teha.

Peamine shelli väärtus on selle jaoks loodud abivahendite hulk: neid on lausa tuhandeid, igaüks täitmas edukalt mingit konkreetset ülesannet. Me võtame vaatluse alla neist ainult üksikud. UNIX® maailma üks suuremaid väärtusi on võimalus kõiki neid vahendeid omavahel kombineerida (nagu edaspidi korduvalt näete).

7.1. Failide käsitlemise vahendid

Antud kontekstis mõistame failide käsitlemise all failide kopeerimist, liigutamist ja kustutamist. Hiljem vaatame ka seda, kuidas muuta failide atribuute (omanik, õigused).

7.1.1. mkdir, touch: tühjade kataloogide ja failide loomine

Käsuga `mkdir` (*MaKe DIRectory*) saab luua katalooge. Selle süntaks on lihtne:

```
mkdir [võtmed] <kataloog> [kataloog ...]
```

Käsu võtmetest väärib märkimist ennekõike `-p`. See on mõeldud kaheks asjaks:

1. see loob ülemkataloogid, kui neid ei peaks eksisteerima. Ilma selle võtmeta käsk `mkdir` lihtsalt ebaõnnestub ja teatab, et vajalikku ülemkataloogi ei ole olemas;
2. see lõpetab oma töö vaikides, kui kataloog, mida soovisite luua, on juba olemas. Kui aga jätate võtme `-p` andmata, tagastab `mkdir` veateate, mille sisuks on teadaanne kataloogi olemasolu kohta.

Toome mõned näited:

- `mkdir foo`: loob kataloogi `foo` selles kataloogis, kus Te parajasti viibite;
- `mkdir -p pildid/muud dokud`: loob kataloogis `pildid` kataloogi `muud`. Kõigepealt luuakse esimesena mainitud kataloog, kui seda veel ei ole (`-p`); samuti luuakse kataloog `dokud` selles kataloogis, kus Te parajasti viibite.

Algselt ei olnud käsk `touch` mõeldud üldse mitte failide loomiseks, vaid failide kasutamise ja muutmise aja uuendamiseks[1]. Kuid käsk `touch` loob ka tühjad failid, kui antud faile ei ole eelnevalt olemas. Käsu süntaks on järgmine:

```
touch [võtmed] fail [fail...]
```

Nii loob käsk:

```
touch fail1 pildid/fail2
```

tühja faili nimega `fail1` kataloogis, kus parajasti asute, ja tühja faili nimega `fail2` kataloogis `pildid`, kui neid faile veel ei olnud olemas.

7.1.2. rm: failide või kataloogide kustutamine

Käsk `rm` (*ReMove*) asendab DOS-i käske `del` ja `deltree` ning võimaldab kasutada rohkem võtmeid. Käsu süntaks on järgmine:

```
rm [võtmed] <fail|kataloog> [fail|kataloog...]
```

Tähtsamad võtmed on järgmised:

- `-r` või `-R`: kustutab rekursiivselt. See võti on **kohustuslik** kataloogi kustutamise korral sõltumata sellest, kas kataloog on tühi või mitte. Tühja kataloogi kustutamiseks võib küll kasutada ka käsku `rmdir`.

1. UNIX® korral on igal failil kolm erinevat ajatemplit: faili viimase kasutamise kuupäev (`atime`), s.t. aeg, mil fail viimati avati lugemiseks või kirjutamiseks; viimane kuupäev, mil muudeti inode atribuute (`ctime`); ning viimane kuupäev, mil muudeti faili **sisu** (`mtime`).

- `-i`: kinnituse nõudmine enne iga kustutamisoperatsiooni. Pange tähele, et vaikimisi on Mandriva Linuxis käsk `rm` õieti ***alias*** ja tegelik käsk ongi turvakaalutlustel `rm -i` (samasugused aliased on käskudele `cp` ja `mv`). Võib muidugi juhtuda, et Te ei pea selliseid aliaseid mõttekaks. Kui soovite need eemaldada, looge tühi fail `~/.alias`, mis takistab süsteemsete aliaste kasutamist. Teine võimalus on redigteerida faili `~/.bashrc` ja tühistada seal mõningad süsteemsed aliased järgmist rida lisades: `unalias rm cp mv`
- `-f`: vastupidiselt võtmele `-i` sunnib faile või katalooge kustutama ka siis, kui kasutajal ei ole nendele failidele kirjutamisõigust[2].

Mõned näited:

- `rm -i pildid/*.jpg fail1`: kustutab kõik failid, mille nime lõpus seisab `.jpg`, kataloogist `pildid`, ning faili `file1` kataloogist, kus Te parajasti viibite, nõudes iga kustutamise eel kinnitust. Kusutamise lubamiseks vajutage klahvile **y**, sellest loobumiseks klahvile **n**.
- `rm -Rf pildid/muud/ fail*`: kustutab kinnitust nõudmata kogu kataloogi `muud/` kataloogis `pildid/` ning kõik failid kataloogis, kus Te parajasti asute, mille nime alguses seisab `fail`.

Käsuga `rm` hävitatakse failid **jäädavalt**. Neid ei ole enam võimalik taastada! (Õigupoolest küll on, aga see ei ole sugugi lihtne ega kiiresti teostatav ülesanne.) Väga soovitatav oleks kasutada võtit `-i`, mis aitab tagada, et Te ei kustuta midagi lihtsalt kogemata.

7.1.3. mv: failide liigutamine või ümbernimetamine

Käsu `mv` (*MoVe*) süntaks on järgmine:

```
mv [võtmed] <fail|kataloog> [fail|kataloog ...] <sihtkoht>
```

Arvestage, et mitme faili liigutamisel peab sihtkoht olema kataloog. Faili ümbernimetamiseks piisab aga sihtkohana uue failinime andmisest.

Mõned võtmed:

- `-f`: sunnib operatsiooni peale. Hoiatust ei anta, kui see nõuab näiteks olemasoleva faili ülekirjutamist.
- `-i`: vastupidine võti, mis soovib saada kasutaja kinnitust enne olemasoleva faili ülekirjutamist.
- `-v`: niinimetatud jutukas (***verbose***) režiim, mis annab teada kõigist muudatustest ja toimingutest.

Mõned näited:

- `mv -i /tmp/pics/*.png .`: liigutab kõik failid kataloogis `/tmp/pics/`, mille nime lõpus seisab `.png`, kataloogi, kus Te parajasti viibite (`.`), kuid nõuab kinnitust enne samanimeliste failide ülekirjutamist.
- `mv foo bar`: annab failile `foo` nimeks `bar`. Kui kataloog `bar` on juba olemas, tähendab see käsk faili `foo` või kogu kataloogi (kataloogi ning rekursiivselt kõigi selles leiduvate failide ja kataloogide) liigutamist kataloogi `bar`.
- `mv -vf fail* pildid/ prügikast/`: liigutab kinnitust nõudmata kõik aktiivse kataloogi failid, mille nime alguses seisab `fail`, ning kogu kataloogi `pildid/` kataloogi `prügikast/` ja näitab iga sooritatud operatsiooni.

2. Piisab, kui kasutajal on kirjutamisõigus **kataloogis**. Sellisel juhul võib ta kustutada selles leiduvaid faile ka juhul, kui ta ei ole nende failide omanik.

7.1.4. cp: failide ja kataloogide kopeerimine

Käsuga `cp` (*CoPy*) saab kopeerida katalooge. See sarnaneb DOS-i käskudega `copy` ja `xcopy`, kuid on võimekam. Selle süntaks on järgmine:

```
cp [võtmed] <fail|kataloog> [fail|kataloog ...] <sihtkoht>
```

Toome mõned levinumad käsu `cp` võtmed:

- `-R`: rekursiivne kopeerimine; **kohustuslik** kataloogi kopeerimisel ka siis, kui tegemist on tühja kataloogiga.
- `-i`: kinnituse nõudmine enne iga võimalikku faili ülekirjutamist.
- `-f`: vastupidiselt võtmele `-i` asendab kõik olemasolevad failid ilma kinnitust küsimata.
- `-v`: jutukas režiim, mis näitab kõiki käsu `cp` sooritatavaid toiminguid.

Mõned näited:

- `cp -i /tpildid/* pildid/`: kopeerib kõik failid kataloogist `/tpildid/` kataloogi `pildid/`, mis asub parajasti aktiivses kataloogis. Kui peaks tekkima vajadus kirjutada olemasolev fail üle, küsitakse Teie käest kinnitust.
- `cp -vR dokud/ /shared/mp3s/* minukraam/`: kopeerib kogu kataloogi `dokud` ning kõik failid kataloogis `/shared/mp3s` kataloogi `minukraam`.
- `cp foo bar`: loob failist `foo` koopia nimega `bar` aktiivses kataloogis.

7.2. Failiatribuutide käsitlemine

Alljärgnevalt toodavate käskudega saab muuta faili omanikku või gruppi või õigusi. Erinevatest õigustest kõneles juba eespool Peatükk 1.

7.2.1. chown, chgrp: ühe või enama faili omaniku ja grupi muutmine

Käsu `chown` (*CHange OWNer*) süntaks on järgmine:

```
chown [võtmed] <kasutaja[:grupp]> <fail|kataloog> [fail|kataloog...]
```

Tähtsamad võtmed on järgmised:

- `-R`: rekursiivne. See võimaldab muuta mingi kataloogi kõigi failide ja alamkataloogide omanikku.
- `-v`: jutukas režiim. Näitab kõiki käsu `chown` sooritatavaid toiminguid, andes teada, millised failid muutsid käsu tõttu omanikku, millised aga mitte.
- `-c`: nagu `-v`, kuid annab teada ainult failidest, mida muudeti.

Mõned näited:

- `chown nobody /shared/raamat.tex`: muudab faili `/shared/raamat.tex` omanikuks **`nobody`**.
- `chown -Rc queen:music *.mid kontserdid/`: muudab kõigi aktiivse kataloogi failide, mille nime lõpus seisab `.mid`, ning kõigi kataloogi `kontserdid/` failide ja alamkataloogide omanikuks `queen` ja grupiks `music`, andes samas teada ainult failidest, mida käsuga muudeti.

Käsk `chgrp` (*CHange GRouP*) võimaldab muuta faili või failide gruppi. Selle süntaks on väga sarnane käsuga `chown`:

```
chgrp [võtmed] <grupp> <fail|kataloog> [fail|kataloog...]
```

Selle käsu võtmed sarnanevad käsu `chown` ning nende kasutaminegi on väga sarnane. Nii muudab käsk `chgrp disk /dev/hd*` kõigi selliste failide gupiks kataloogis `/dev`, mille nime alguses seisab `hd`, grupi `disk`.

7.2.2. chmod: failide ja kataloogide õiguste muutmine

Käsul `chmod` (*CHange MODe*) on väga eriline süntaks. See näeb välja esmapilgul suhteliselt tavaline:

```
chmod [võtmed] <muutmisrežiim> <fail|kataloog> [fail|kataloog...]
```

Eriliseks muudab selle aga mitu erinevat viisi, milles muutmisrežiim võib väljenduda. Põhimõtteliselt saab see olla kahesugune:

1. kaheksandsüsteemis. Omaniku õigused vastavad sel juhul arvule kujul `<x>00`, kus `<x>` tähistab omistatud õigust: 4 lugemise, 2 kirjutamise ja 1 käivitamise õigust. Grupi õigused esitatakse kujul `<x>0` ning "teiste" õigused kujul `<x>`. Seejärel tuleb vajaliku režiimi leidmiseks kõik need kolm arvu kokku liita. Niisiis vastab õigustele `rwxr-xr--` 400+200+100 (omaniku õigused `rwx`) +40+10 (grupi õigused `r-x`) +4 (teiste õigused `r--`) = 754. See väljendab õigusi absoluutse väärtusena, mis tähendab, et igasugused varasemad õigused asendatakse ilma igasuguste mööndusteta;
2. avaldised. Õigusi väljendatakse sel juhul komadega eraldatud avaldiste jadana. Avaldis võtab niisiis järgmise kuju: `[kategooria]<+|-|=><õigused>`.

   Kategooria võib olla üks järgmistest:

   - `u` (*User*, omaniku õigused);
   - `g` (*Group*, omanikugrupi õigused);
   - `o` (*Others*, õigused "teiste" jaoks).

   Kui kategooria on määramata, rakendatakse muudatusi kõigile kategooriatele. Sümbol + lisab õiguse, − eemaldab õiguse ja = määrab õiguse. Õigus ise võib olla üks järgmistest:

   - `r` (*Read*, lugemise õigus),
   - `w` (*Write*, kirjutamise õigus) või
   - `x` (*eXecute*, käivitamise õigus).

Peamised võtmed on üsna sarnased käskude `chown` ja `chgrp` võtmetega:

- `-R`: muudab õigusi rekursiivselt.
- `-v`: jutukas režiim. Näitab iga failiga sooritatud toiminguid.
- `-c`: nagu `-v`, aga näitab ainult faile, mille puhul käsk midagi muutis.

Näited:

- `chmod -R o-w /shared/dokud`: eemaldab rekursiivselt `teiste` kirjutamisõiguse kõigile failidele ja alamkataloogidele kataloogis `/shared/dokud/`.
- `chmod -R og-w,o-x privaat/`: eemaldab rekursiivselt `grupi` ja `teiste` kirjutamisõiguse kogu kataloogile `privaat/` ning eemaldab `teiste` käivitamisõiguse.
- `chmod -c 644 muud/fail*`: muudab kõigi failide õiguseks kataloogis `muud/`, mille nime alguses seisab `fail`, `rw-r--r--` (s.t. lugemisõigus kõigile ja kirjutamisõigus ainult omanikule) ning annab teada ainult käsuga muudetud failidest.

7.3. Metamärkide kasutamine shellis

Tõenäoliselt oled juba kokku puutunud ***metamärkidega***, isegi kui sa ei tea, mida see täpselt tähendab. Kui näiteks soovid Windows® kasutades faili otsida või avada, võid kasutada suvalise faili leidmiseks sümbolit `*`. Näiteks `*.txt` leiab kõik failid, mille nime lõpus seisab `.txt`. Me kasutasime seda üsna palju ka viimases osas. Kuid metamärgid võimaldavad palju enam kui pelgalt `*` sellist kasutamist.

Kui Te kirjutate käsu `ls *.txt` ja vajutate klahvi **Enter**, ei soorita selliste failide otsimist, mis vastaksid mustrile `*.txt`, mitte käsk `ls`, vaid shell ise. Siinkohal tasub veidi seletada, kuidas shell tõlgendab käsurida. Kui Te kirjutate:

```
$ ls *.txt
readme.txt  recipes.txt
```

jagatakse käsurida kõigepealt sõnadeks (`ls` ja `*.txt` antud näites). Kui shell näeb sõnas sümbolit `*`, tõlgendab see kogu sõna metamärki kasutava mustrina ning asendab selle kõigi sobivate failide nimedega. Seepärast omandab käsk enne seda, kui shell selle teostab, kuju `ls readme.txt recipe.txt`, mis annabki meile oodatud tulemuse. Ka mõned muud sümbolid panevad shelli selliselt käituma:

- `?`: sobib üks ja ainult üks sümbol sõltumata sellest, milline see sümbol on;
- `[...]`: sobib iga nurksulgudes leiduv sümbol. Sümboleid võib anda kas sümbolite vahemikuna (s.t. `1-9`) või ***diskreetsete väärtustena*** või mõlemal kujul. Näide: `[a-zBE5-7]` korral sobivad kõik sümbolid `a` ja `z` vahel, `B`, `E`, `5`, `6` või `7`;
- `[!...]`: sobib iga sümbol, mida **ei leidu** nurksulgudes. Näiteks `[!a-z]` korral sobib iga sümbol, mis ei ole väiketäht[3];
- `{c1,c2}`: sobib `c1` või `c2`, kus `c1` ja `c2` on omakorda samuti metamärke kasutavad mustrid, mis tähendab, et võite näiteks kirjutada `{[0-9]*,[acr]}`.

Toome siin ära mõningad mustrid ja nende tähendused:

- `/etc/*conf`: kõik failid kataloogis `/etc`, mille nime lõpus seisab `conf`. Sobivad `/etc/inetd.conf`, `/etc/conf.linuxconf` **ja ka** `/etc/conf` , kui see on olemas. Pidage meeles, et `*` võib tähendada ka tühja stringi.
- `pildid/{autod,kosmos[0-9]}/*.jpg`: kõik failid, mille nime lõpus seisab `.jpg`, kataloogides `pildid/autod`, `pildid/kosmos0`, (...), `pildid/kosmos9`, kui need kataloogid on olemas.
- `/usr/share/doc/*/README`: kõik failid nimega `README` kõigis kataloogi `/usr/share/doc` otsestes alamkataloogides. See tähendab, et sobib näiteks `/usr/share/doc/mandriva/README`, aga mitte `/usr/share/doc/minuprogramm/doc/README`.
- `*[!a-z]`: kõik failid aktiivses kataloogis, mille nime lõpus **ei seisa** väiketäht.

7.4. Ümbersuunamine ja torud

7.4.1. Veel veidi protsessidest

Ümbersuunamise ja torude mõistmiseks peame rääkima taas veidi protsessidest. Enamik UNIX® protsesse (nende hulka kuuluvad ka graafilised rakendused, aga ei kuulu enamik deemoneid) kasutavad vähemalt kolme faili deskriptorit: standardsisend, standardväljund ja standardveaväljund. Neid tähistavad vastavalt numbrid 0, 1 ja 2. Üldiselt on need deskriptorid seotud terminaliga, kus protsess käivitati, ning sisendiks on klaviatuur. Ümbersuunamise ja torude eesmärk ongi deskriptorid ümber suunata. Paremini aitavad seda kontseptsiooni mõista käesolevas osas toodud näited.

7.4.2. Ümbersuunamine

Oletame näiteks, et tahate näha nimekirja kõigi failidega, mille lõpus seisab `.png`[4], ja mis asuvad kataloogis `pildid`. Nimekiri tuleks väga pikk, mistõttu oleks mõttekas see salvestada faili, et siis sellega sobival ajal rahulikult tutvuda. Selleks võib anda järgmise käsu:

```
$ ls pildid/*.png 1>faili_nimekiri
```

See tähendab, et antud käsu standardväljund (`1`) suunatakse ümber (`>`) faili nimega `faili_nimekiri`. Operaator `>` ongi väljundi ümbersuunamise operaator. Kui faili, kuhu väljund suunatakse, ei ole olemas, see luuakse,

3. Ettevaatust! See võib küll nii olla paljude keelte korral, aga mitte tingimata Teie keele puhul (`lokaat`). See sõltub niinimetatud **põimejärjekorrast**. Mõnes keeles vastab määratlusele `[a-z]` a, A, b, B, (...), z. Ja me ei hakka mainimagi, et mõnes keeles on näiteks täpitähed...
4. Te võite arvata, et on mõttetu rääkida "failidest, mille lõpus seisab `.png`", ja et selle asemel võiks öelda pigem "PNG pildid". Kuid meenutame taas, et UNIX® korral on failidel laiend ainult moepärast: laiendid ei määra mingil moel faili tüüpi. Fail, mille lõpus seisab `.png`, võib väga hästi olla ka JPEG pilt, mõne rakenduse fail, tekstifail või mis tahes muu fail. Õigupoolest on see nii isegi Windows® korral!

kui see aga peaks eksisteerima, kirjutatakse selle varasem sisu lihtsalt üle. Pange tähele, et antud operaatoriga vaikimisi ümbersuunatav deskriptor ongi standardväljund, mistõttu seda ei ole vaja spetsiaalselt käsureal määrata. Niisiis võite käsu anda ka lihtsamal kujul:

```
$ ls pildid/*.png >faili_nimekiri
```

ning tulemus on täpselt sama. Seejärel võite vaadata tulemuseks saadud faili mõne tekstifailide näitajaga, näiteks `less`.

Oletame nüüd, et Te soovite teada, kui palju selliseid faile üldse on. Te ei pea neid käsitsi üle lugema, vaid võite kasutada sõnade loendamise utiliiti `wc` (*Word Count*) võtmega `-l`, mis näitab standardväljundis ridade arvu failis. Üks võimalus on selline:

```
$ wc -l 0<faili_nimekiri
```

mis annab just vajaliku tulemuse. Operaator < on sisendi ümbersuunamise operaator ning vaikimisi ümbersuunatav deskriptor ongi standardsisend, s.t. `0`, mistõttu võite ka anda lihtsalt käsu:

```
$ wc -l <faili_nimekiri
```

Nüüd aga oletame, et soovite eemaldada kõigi failide "laiendid" ning näha tulemust omaette failis. Seda aitab teha abivahend `sed` (*Stream EDitor*). Te saadate lihtsalt programmi `sed` standardsisendi faili `faili_nimekiri` ja suunate selle väljundi tulemusfaili, s.t. faili `nimekiri`:

```
$ sed -e 's/\.png$//g' <faili_nimekiri >nimekiri
```

misjärel vajalik nimekiri luuaksegi ja Te võite seda vaadata just meelepärase failinäitajaga.

Mõnikord võib olla kasulik suunata ümber ka standardveaväljund. Näiteks võib Teil tekkida soov teada, millistele kataloogidele kataloogis `/shared` Te ligi ei pääse. Üks võimalus on lasta näidata kataloogi sisu rekursiivselt, kuid suunata vead faili, mitte aga lasta neid näidata standardväljundis (s.t. ekraanil):

```
$ ls -R /shared >/dev/null 2>vead
```

mis tähendab, et standardväljund suunatakse ümber (>) spetsiaalsesse faili `/dev/null`, mis tühistab kõik, mida Te kirjutate (s.t. standardväljundit ei näidata), ning standardveaväljund (`2`) suunatakse ümber (>) faili `vead`.

7.4.3. Torud

Torud on teatud mõttes sisendi ja väljundi ümbersuunamise kombinatsioon. Põhimõte sarnaneb reaalsete torudega (sellest ka nimi): üks protsess saadab andmed toru ühte otsa ja teine protsess loeb need toru teises otsas. Toru operaatoriks on sümbol `|`. Võtame uuesti ette ülaltoodud failinimekirja. Oletame, et soovite otsekohe teada vajalike failide arvu, ilma et selleks peaks nende nimekirja salvestama ajutisse faili. Seda teeb järgmine käsk:

```
$ ls pildid/*.png | wc -l
```

mis tähendab, et käsu `ls` standardväljund (s.t. failide nimekiri) suunatakse ümber käsu `wc` standardsisendisse. See annab seejärel Teile vajaliku tulemuse.

Järgmise käsuga võite vahetult koostada nimekirja "ilma laiendita" failidest:

```
$ ls pildid/*.png | sed -e 's/\.png$//g' >nimekiri
```

või kui soovite nimekirja otse uurida seda failina salvestamata:

```
$ ls pildid/*.png | sed -e 's/\.png$//g' | less
```

Torude ja ümbersuunamise kasutusala ei piirdu kaugeltki tekstiga, mida inimsilm suudaks lugeda. Kui näiteks anda järgmine käsk Terminalis:

```
$ xwd -root | convert - ~/minu_desktop.png
```

saadetakse Teie töölauast tehtud pilt faili `minu_desktop.png`[5] Teie kodukataloogis.

5. Jah, see on nüüd tõesti PNG pilt (ainult et paigaldatud peaks olema ka tarkvarapakett ImageMagick...).

7.5. Käsurea lõpetamine

Lõpetamine on äärmiselt mugav asi, mida võimaldavad kõik moodsad shellid (sealhulgas bash). Selle mõte on võimaldada kasutajal vajaliku tööga võimalikult ruttu valmis saada. Kõige parem on lõpetamist selgitada näidete varal.

7.5.1. Näide

Oletame, et Teie kodukataloogis leidub `fail_nii_jube_pika_nimega_mis_kuidagi_meelde_ei_jää`, aga Te siiski soovite seda uurida. Oletame ka, et Teil on samas kataloogis veel fail `fail_tekst`. Te viibite parajasti kodukataloogis, nii et andke käsk:

```
$ less fa<TAB>
```

(s.t. kirjutage `less fa` ja vajutage seejärel tabulaatoriklahvi **TAB**). Nüüd avab shell ise Teie eest käsurea võimalused:

```
$ less fail_
```

ning näitab ühtlasi võimalikke valikuid (vähemalt vaikeseadistuse korral; seda saab ka muuta). Seejärel kirjutage käsureale veel täht n ja vajutage uuesti tabulaatoriklahvi:

```
$ less fail_w<TAB>
```

ning shell täidab käsurea just Teile vajaliku tulemusega:

```
$ less fail_nii_jube_pika_nimega_mis_kuidagi_meelde_ei_jää
```

Nüüd tuleb Teil vaid vajutada kinnituseks klahvi **Enter** ja võitegi asuda faili lugema.

Failinäitajast väljumiseks vajutage klahvi **q**.

7.5.2. Muud lõpetamismeetodid

Klahv TAB pole sugugi ainus viis lõpetamise aktiveerimiseks, kuigi ilmselt levinuim. Reeglina lõpetatakse käsureal esimesena antud sõna korral käsu nimetus (`nsl<TAB>` annab tulemuseks `nslookup`) ning ülejäänud sõnade korral faili nimi, kui just sõna ees ei seisa mõni "maagiline" sümbol, näiteks `~`, `@` või `$`, mis sunnib shelli üritama lõpetada vastavalt kasutajanime, masinanime või keskkonnamuutujat[6]. Lisaks on olemas maagiline sümbol ka failinime lõpetamiseks (`/`) ning käsk, millega kutsuda välja käsk ajaloost (`!`).

Veel kaks võimalust lõpetamine aktiveerida on klahvikombinatsioonid **Esc-<x>** ja **Ctrl-X-<x>**, kus **<x>** on üks äsjamainitud maagilistest sümbolitest. **Esc-<x>** üritab hakkama saada ühe ja unikaalse lõpetamisega. Kui see ei õnnestu, lõpetatakse sõna valikunimekirja suurimat võimalikku alamstringi sisaldava sõnaga. ***Piiks*** tähendab, et valik ei olnud unikaalne või et vajalikku sõna ei leitudki. Klahvikombinatsioon **Ctrl-X-<x>** näitab võimalike valikute nimekirja ilma lõpetamist üritamata. Vajutamine klahvile **TAB** on õigupoolest sama, mis üksteise järel kombinatsioonide **Esc-<x>** ja **Ctrl-X-<x>** vajutamine, kus maagiline sümbol sõltub kontekstist.

Niisiis on üks võimalus näha kõiki defineeritud keskkonnamuutujaid anda tühjal real käsk **Ctrl-X-$**. Teine näide: kui soovite näha käsu `nslookup` man-lehekülge, kirjutage lihtsalt `man nsl`, seejärel vajutage **Esc-!** ning shell lõpetab Teie eest automaatselt käsu `man nslookup`.

6. Pidage meeles, et UNIX® teeb vahet suur- ja väiketähtedel. Keskkonnamuutuja `HOME` ja muutuja `home` ei ole sugugi üks ja sama asi.

7.6. Taustaprotsesside käivitamine ja käsitlemine: tööde juhtimine

Arvatavasti olete juba märganud, et kui annate Terminalis käsu, peate tavaliselt ootama, kuni see täidetakse, enne kui shell Teil uuesti lubab tegutsema asuda. See tähendab, et olete saatnud käsu ***esiplaanile***. Siiski pole see sugugi alati soovitatav ega soovitud.

Oletame näiteks, et soovite kopeerida üsna suure kataloogi kuhugi mujale. Samuti soovite ignoreerida vigu, mistõttu suunate veaväljundi faili `/dev/null`:

```
cp -R pildid/ /shared/ 2>/dev/null
```

Sellise käsu teostamisele võib kuluda mitu minutit. Seepärast on Teil kaks võimalust, kui soovite vahepeal käsureal edasi tegutseda: esimene on vägivaldne käsu peatamine (tapmine) ning selle taaskäivitamine millalgi, kui aega on rohkem. Selleks vajutage klahve **Ctrl**-**C**: see katkestab protsessi ja toob Teie silmade ette taas viiba. Kuid Te ei pruugi sugugi nii talitada! Lugege edasi.

Oletame, et soovite jätta käsu tegutsema, samal ajal kui ise võtate midagi muud ette. Seda võimaldab protsessi saatmine ***taustale***. Selleks vajutage **Ctrl**-**Z** protsessi ajutiseks peatamiseks:

```
$ cp -R images/ /shared/ 2>/dev/null
# Vajutage nüüd Ctrl+z
[1]+  Stopped                 cp -R pildid/ /shared/ 2>/dev/null
$
```

ja Teie ees seisab taas viip. Protsess on nüüd ootel, oodates Teilt käsku taas tegevust alustada (seda näitab võtmesõna `Stopped`). Seda Te muidugi soovitegi, aga muu tegevuse taustal. Soovitud tulemuse saavutamiseks andke käsk `bg` (*BackGround*):

```
$ bg
[1]+ cp -R pildid/ /shared/ 2>/dev/null &
$
```

Protsess käivitatakse nüüd uuesti, aga juba taustal töötava ülesandena, millele osutab rea lõpus seisev sümbol `&` (ampersand). Teie ees aga seisab taas viip ning Te võite oma asjadega edasi tegutseda. Protsessi, mis töötab taustaülesandena või taustal, nimetataksegi ***taustatööks***.

Mõistagi on võimalik panna protsesse ka juba taustal käivituma, kui lisada käsu lõppu sümbol **&**. Nii võite näiteks käivitada kataloogi kopeerimise käsu taustal sel moel:

```
$ cp -R pildid/ /shared/ 2>/dev/null &
```

Soovi korral võite protsessi uuesti esiplaanile tuua ja jääda ootama selle lõppemist käsuga `fg` (*ForeGround*). Uuesti taustale saatmiseks andke korraldused **Ctrl**-**Z** ja `bg`.

Nii saab käivitada ka mitu tööd: igale käsule antakse sel juhul omaette töö number. shelli käsk `jobs` näitab kõiki töid, mis on seotud aktiivse shelliga. Kui tööle eelneb märk `+`, on see see käivitatud taustaülesandena. Mõne konkreetse töö taastamiseks esiplaanil andke korraldus `fg <n>`, kus `<n>` on töö number, näiteks `fg 5`.

Pange tähele, et sel moel saab peatada või käivitada ka ***täisekraanirežiimis*** rakendusi, näiteks `less` või tekstiredaktor Vi, ning soovi korral neid esiplaanile taastada.

7.7. Lõppsõna

Nagu näete, on shell väga mitmekülgne ja selle maksimaalselt tõhus kasutamine eeldab teatud harjutamist. Selja taha jäänud suhteliselt pikas peatükis jõudsime mainida vaid väheseid võimalikke käske: Mandriva Linux pakub tuhandeid utiliite ning isegi kõige kogenenumad kasutajad ei pruugi enamasti neid kõiki.

Utiliite on igale maitsele ja igaks eesmärgiks: mõned võimaldavad käsitleda pilte (näiteks juba mainitud `convert`, aga ka GIMPi ***pakktöötluse*** režiim ja kõik ***pikselrasterfailide*** käsitlemise utiliidid), mõned tegelevad heliga (Ogg Vorbis kodeerijad, audio CD mängijad), CD-de kirjutamisega, mõned on e-posti kliendid, FTP kliendid või isegi veebilehitsejad (näiteks `lynx` või `links`), rääkimata juba arvukatest haldusvahenditest.

Isegi siis, kui on olemas sarnaste funktsioonidega graafilised rakendused, kujutavad need enamasti lihtsalt käsureavahendite alusele ehitatud graafilist liidest. Pealegi on käsureautiliitide eeliseks see, et nad võivad tegutseda mitteinteraktiivselt: Te võite näiteks käivitada CD kirjutamise ja siis kas või süsteemist välja logida, olles kindel, et kirjutamine igal juhul toimub (vt. nohup(1)man-lehekülge või screen(1)man-lehekülge).

Peatükk 8. Teksti redigeerimine: Emacs ja VI

Nagu sissejuhatuses märgitud, on teksti redigeerimine[1] UNIX® süsteemi kasutamisel fundamentaalse tähendusega. Kaks redaktorit, mida me järgnevalt põgusalt tutvustame, on vahest esmapilgul keerulised kasutada, aga nende põhitõdede omandamise järel veendute kindlasti, et tegemist on väga võimsate tööriistadega. Eriti tulevad kasuks erinevad redigeerimisrežiimid, mis pakuvad spetsiifilisi redigeerimisvõimalusi vastavalt failitüübile (Perl, C++, XML jne.).

8.1. Emacs

Emacs on arvatavasti üldse maailma kõige võimsam tekstiredaktor. See suudab teha absoluutselt kõige ja seda saab lõputult laiendada rakendusse sisseesitatud lispi-põhise programmeerimiskeele abil. Emacs võimaldab Teil liikuda veebis, lugeda e-kirju, osaleda Useneti uudistegruppides, teha kohvi ja nii edasi. See ei tähenda muidugi, et Te saaksite käesolevast peatükist seda kõike teada, sest mahupiirangud lihtsalt ei võimalda täielikku ülevaadet anda, Küll aga saate siit teada, kuidas Emacs käima panna, üht või ka rohkemat faili redigeerida, need salvestada ja Emacsist väljuda.

Kui te selle lõigu järel soovite Emacsi kohta rohkem teada saada, võiks ette võtta GNU Emacsi sissejuhatava õppematerjali (`http://www.lib.uchicago.edu/keith/tcl-course/emacs-tutorial.html`) (see on küll inglise keeles).

8.1.1. Lühitutvustus

Emacsi käivitamine käsurealt käib nii:

```
emacs [fail1] [fail2...]
```

Emacs avab iga argumendina antud faili omaette puhvris. Kui käsureal anda enam kui kaks faili, jagatakse aken automaatselt kaheks, millest ühes osas näidatakse viimasena antud faili, teises aga saadaolevate puhvrite nimekirja. Kui käivitada Emacs käsureal faile andmata, satute puhvrisse, mis kannab nimetust `scratch*`. Kui töötate X'i keskkonnas, saab menüüsid kasutada hiirega, kui tekstirežiimis, siis saab menüüsid kasutada klahviga **F10**. Käesolevas peatükis piirdume ainult klaviatuuril töötamise käsitlemisega ega puuduta menüüsid.

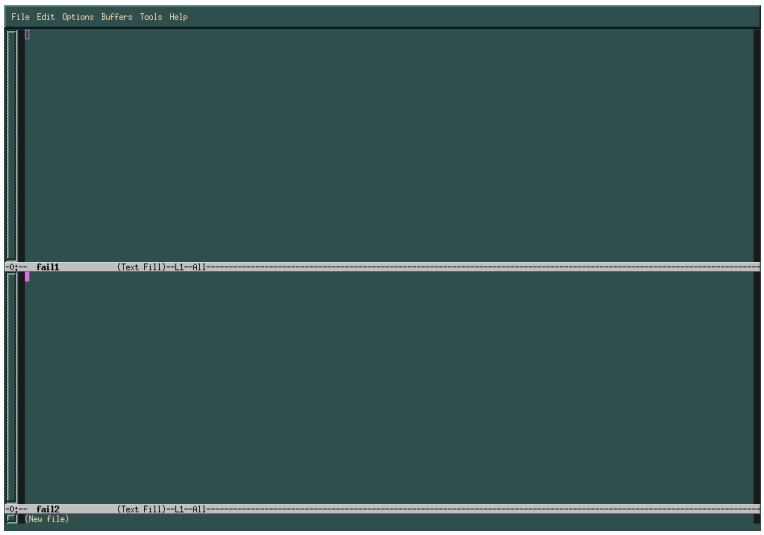
8.1.2. Alustamine

Võtame nüüd ette praktika. Alustame kõigepealt kaht faili avades, milleks on meie näites `fail1` ja `fail2`. Kui neid ei peaks olemas olema, luuakse need kohe, kui annate käsureal sellise korralduse:

```
$ emacs fail1 fail2
```

Seejärel ilmub nähtavale järgmine aken:

1. "Teksti redigeerimine" tähendab siinkohal ainult tähti, numbreid ja kirjavahemärke sisaldavate failide sisu muutmist. See ei puuduta paigutusega seotud asju nagu fondid või joondamine jms. Mainitud failideks võivad olla e-kirjad, lähtekood, dokumendid või isegi konfiguratsioonifailid.



Joonis 8-1. Kahe faili redigeerimine korraga

Nagu näete, loodi kaks puhvrit. Õigupoolest on näha ka kolma puhver ekraani allosas, kus seisab `(New file)`. See on minipuhver, mida ei saa vahetult kasutada. Seda kutsub teid kasutama Emacs ise interaktiivse suhtluse ajal. Aktiivse puhvri muutmiseks andke korraldus **Ctrl-X-O**. Teksti saab sisestada nagu "normaalses" redaktoris, märkide kustutamiseks on mõeldud klahv **Del** või **Backspace**.

Liikumiseks saab kasutada nooleklahve või siis järgmisi klahvikombinatsioone: **Ctrl-A** liikumiseks rea algusesse, **Ctrl-E** liikumiseks rea lõppu, **Alt-<** või **Ctrl-Home** liikumiseks puhvri algusesse ja **Alt->** või **Ctrl-End** liikumiseks puhvri lõppu. Selliseid kombinatsioone on veel päris palju, sealhulgas kõigile nooleklahvidele [2].

Kui olete valmis tehtud muudatusi salvestama, andke käsk **Ctrl-XCtrl-S** või **Ctrl-XCtrl-W**, kui soovite puhvri sisu salvestada mõne muu failina. Emacs pärib Teie käest, milline nime soovite salvestatavale failile anda. Selleks võite kasutada ka ***lõpetamist*** **Tab**-klahviga (täpselt nagu lõpetamise korral bash'is).

8.1.3. Puhvrite kasutamine

Soovi korral võite lasta ekraanil näidata ainult üht puhvrit. Seda saab teha kahel viisil:

- Kui viibite puhvris, mida soovite ekraanil varjata, andke käsk **Ctrl-X0**.
- Kui viibite puhvris, mida soovite ekraanile jätta, andke käsk **Ctrl-X1**.

Puhvri saab tagasi ekraanile tuua kahel viisil:

- andke käsk **Ctrl-XB** ja kirjutage vajaliku puhvri nimi või
- andke käsk **Ctrl-XCtrl-B**. See avab uue puhvri nimetusega `*Buffer List*`. Selles puhvris saab liikuda, kui anda korraldus **Ctrl-XO**, valida vajalik puhver ja vajutada klahvi **Enter** või kirjutada puhvri nimi minipuhvrisse. Kui olete valiku langetanud, kaob puhver `*Buffer List*` taas taustale.

Kui olete töö faili kallal lõpetanud ja soovite vastavast puhvrist lahti saada, andke käsk **Ctrl-XK**. Emacs pärib seejärel, milline puhver sulgeda. Vaikimisi on selleks puhver, kus Te parasjagu viibite. Kui soovite lahti saada mõnest muust puhvrist, andke selle nimi või vajutage **Tab**-klahvi. Emacs avab siis uue puhvri nimetusega `*Completions*`, kus on näha saadaolevad võimalused. Kinnitage oma valik klahvi **Enter** vajutamisega.

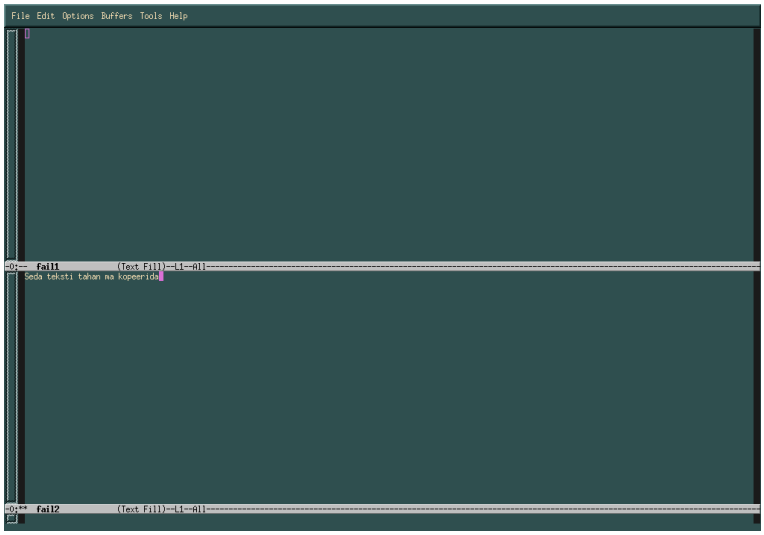
Te võite alati tekitada ekraanile kaks nähtavat puhvrit. Selleks andke käsk **Ctrl-X2**. Vaikimisi luuakse uus puhver aktiivse puhvri koopiana (mis muu hulgas lubab teil suuri faile "ühel ja samal ajal" mitmes erinevas kohas redigeerida). Puhvrite vahel liikumiseks kasutage eelpool mainitud käske.

Te võite alati avada uusi faile, andes käsu **Ctrl-XCtrl-F**. Emacs pärib Teie käest faili nime ning soovi korral võite ka siin kasutada lõpetamist.

2. Emacs on loodud töötama väga mitmesugustel masinatel, ka sellistel, mille klaviatuuril puuduvad noolenupud. Täpselt samad sõnad kehtivad Vi kohta.

8.1.4. Kopeerimine, lõikamine, asetamine, otsimine

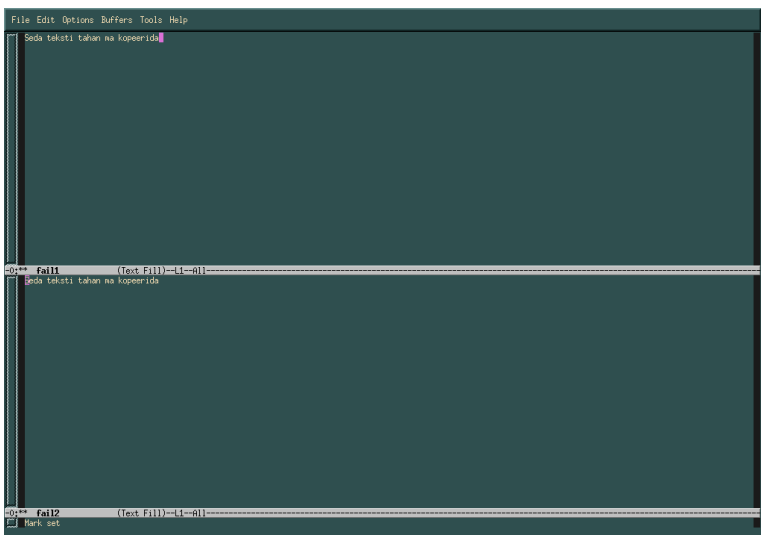
Oletame, et Teie ees seisab järgmine situatsioon; Joonis 8-2.



Joonis 8-2. Emacs enne tekstiploki kopeerimist

Kõigepealt tuleb valida tekst, mida soovite kopeerida. Antud näites soovime kopeerida terve lause. Selleks tuleb esmalt asetada märge vajaliku ala algusesse. Eeldusel, et kursor on seal, kus teda näitab Joonis 8-2, peab käsk olema **Ctrl**-**Tühik** (**Control** pluss tühikuklahv). Emacs näitab seepeale minipuhvris teadet `Mark set`. Seejärel liikuge käsuga **Ctrl**-**A** rea algusesse. Kopeerimiseks või lõikamiseks märgitakse kogu ala, mis jääb märke ja kursori hetkeasukoha vahele, niisiis antud juhul terve tekstirida. Nüüd saab kasutada kaht käsku: **Alt**-**W** (kopeerimine) või **Ctrl**-**W** (lõikamine). Kopeerimise korral naaseb Emacs mõne hetke pärast märke asukohta, nii et näete valitud ala.

Lõpuks liikuge puhvrisse, kuhu soovite teksti kopeerida, ja andke käsk **Ctrl**-**Y**. See annab järgmise tulemuse:



Joonis 8-3. Teksti kopeerimine Emacsis

Sisuliselt kopeerisite te eelneva operatsiooniga teksti Emacsi ***surmaringi***. Surmaring sisaldab kõiki pärast Emacsi käivitamist kopeeritud või lõigatud alasid. **Kõik** alad kopeeritakse või lõigatakse surmaringi esiotsa. Käsk **Ctrl**-**Y** "asetab" ainult kõige ees asuva ala. Kui soovite asetada mõnda muud ala, vajutage **Ctrl**-**Y** ja seejärel **Alt**-**Y**, kuni jõuate vajaliku tekstini.

Teksti otsimiseks liikuge vajalikku puhvrisse ja andke käsk **Ctrl**-**S**. Emacs pärib, millist sõnet otsida. Sama stringi uueks otsimiseks aktiivses puhvris andke lihtsalt uuesti käsk **Ctrl**-**S**. Kui Emacs jõuab puhvri lõppu ega leia rohkem antud sõne esinemisi, võite anda veel kord käsu **Ctrl**-**S**, misjärel otsingut alustatakse puhvri algusest. Klahvile **Enter** vajutades saab otsingu peatada.

Otsimiseks ja asendamiseks andke käsk **Alt-%**. Emacs pärib, millist sõnet otsida ja millega see asendada ning küsib iga sobiva sõne leidmisel enne asendamist Teie käest kinnitust.

Toimingute tagasivõtmiseks andke käski **Ctrl-XU** või **Ctrl-Shift--**, nii saate tühistada viimase toimingu. Te võite tagasi võtta nii palju toiminguid, kui vaid soovite.

8.1.5. Emacsist väljumine

Kiirkorraldus Emacsist väljumiseks on **Ctrl-XCtrl-C**. Kui Te ei ole tehtud muudatusi salvestanud, pärib Emacs, kas soovite puhvrid salvestada või mitte.

8.2. Vi: esivanem

Vi oli üldse esimene täisekraaniredaktor. See on üks tähtsamaid programme, millele osutavad nii UNIX® põlgurid kui ka selle kaitsjad: kuigi programmi omandamine ei pruugi olla kõige kergem, on see põhivõtete omandamisel siiski äärmiselt võimas tööriist. Mõne üksiku klahvivajutusega võib Vi kasutaja sõna otseses mõttes mägesid liigutada - ja kui Emacs välja jätta, polegi õigupoolest peaaegu ühtegi tekstiredaktorit, mis midagi samasugust suudaks.

Mandriva Linux pakub Teile tegelikult varianti Vim ehk niinimetatud täiustatud VI'd (*VI iMproved*), kuid käesolevas peatükis kasutame siiski läbivalt nimetust Vi.

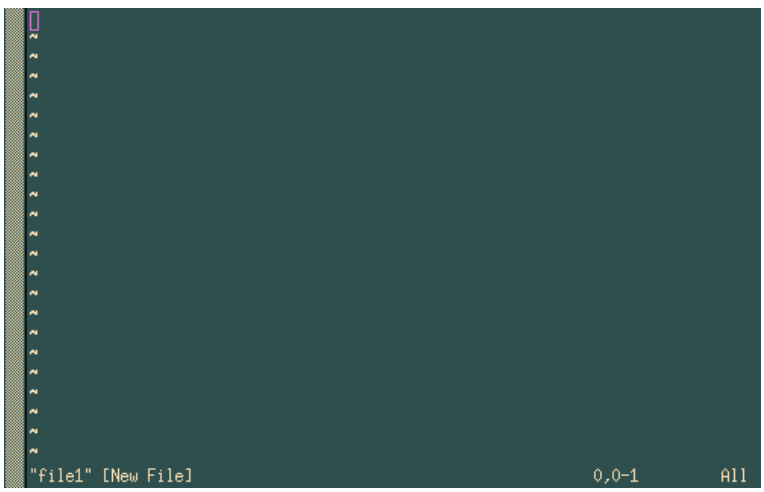
Kui soovite Vi kohta rohkem teada saada, uurige (inglise keeles) sissejuhatust Hands-On Introduction to the Vi Editor (`http://www.library.yale.edu/wsg/docs/vi_hands_on/`) või Vim'i kodulehekülge (`http://www.vim.org/`) (eesti keeles on võimalik vi'ga põgusalt tutvust teha PingviiniWikis (`http://wiki.linux.ee/phpwiki/vi`)).

8.2.1. Lisamisrežiim, käsurežiim, ex-režiim...

Vi kasutamiseks käivitame selle käsurealt sarnaselt Emacsiga. Võtame siis taas ette meie kaks näitefaili ja anname korralduse:

```
$ vi fail1 fail2
```

Seejärel ilmub Teie ette selline aken:



Joonis 8-4. VIM alustamishetkel

Te olete nüüd ***käsurežiimis*** esimese avatud faili alguses. Selles režiimis ei ole faili võimalik teksti sisestada. Selleks tuleb Teil lülituda ***lisamisrežiimi***.

Toome mõned kiirkorraldused teksti lisamiseks:

Pange tähele, et kiirkorraldused tuleb anda täpselt nii, nagu siin näidatud, sest Vi teeb käskudes vahet suur- ja väiketähtedel. Sestap on näiteks käsud **a** ja **A** täiesti erinevad asjad.

- **a** ja **í**: teksti lisamine kursori ette ja järele (**A** ja **I** võimaldavad lisada teksti vastavalt aktiivse rea lõppu ja algusesse);
- **o** ja **O**: teksti lisamine aktiivse rea alla ja kohale.

Lisamisrežiimis on ekraani allosas näha sõne `---INSERT---`, mis annab Teile teada, millises režiimis parajasti viibite. See on ainus režiim, milles saab teksti sisestada. Käsurežiimi naasmiseks vajutage klahvi **Esc**.

Lisamisrežiimis saate klahvidega **Backspace** ja **DEL** teksti kustutada. Nooleklahvidega saab tekstis liikuda nii käsu- kui lisamisrežiimis. Käsurežiimis on veel mitmeid klahvikombinatsioone, mida tutvustame allpool.

`ex`-režiimi saab siseneda käsureale **:** kirjutades. Märk **:** ilmub ekraani alumisse vasakusse serva ja selle järel asetseb kursor. Vi peab kõike, mida Te nüüd enne klahvile **Enter** vajutamist kirjutate, `ex`-käsuks. Kui kustutate käsu ja **:**, naasete käsurežiimi ning kursor suundub taas oma varasemasse kohta tekstis.

`ex`-režiimis saab kasutada ka lõpetamist: kirjutage käsust esimesed sümbolid ja vajutage selle automaatseks lõpetamiseks klahvi **Tab**.

Faili tehtud muudatuste salvestamiseks andke käsurežiimis korraldus **`:w`**. Kui soovite salvestada puhvri sisu mõnda muusse faili, andke korraldus **`:w <failinimi>`**.

8.2.2. Puhvrite kasutamine

Liikumiseks samas puhvris failide vahel, mille nimed andsite käsureal, andke järgmisele failile liikumiseks korraldus **`:next`** ja eelmisele failile liikumiseks **`:prev`**. Te võite kasutada ka käsku **`:e <failinimi>`**, millega saate liikuda vajalikule failile, kui see on juba avatud, või avada uue faili. Kasutada saab ka lõpetamist.

Nagu Emacsi korral, saab ka vi's lasta ekraanil näidata mitut puhvrit korraga. Selleks kasutage käsku **`:split`**.

Puhvri vahetamiseks andke käsk **Ctrl**-**wj** alumisse või **Ctrl**-**wk** ülemisse puhvrisse liikumiseks. **j** ja **k** asemel võib kasutada ka üles ja alla osutavaid nooleklahve. Käsk **`:close`** peidab puhvri, käsk **`:q`** aga sulgeb.

Tasub silmas pidada, et kui üritate puhvri peita või sulgeda ilma sellesse tehtud muudatusi salvestamata, ei võeta käsku arvesse ja Vi näitab sellist teadet:

```
No write since last change (use ! to override)
```

Sellisel juhul tehke nii, nagu öeldud, ja andke käsk **`:q!`** või **`:close!`**.

8.2.3. Teksti redigeerimine ja liikumiskäsud

Lisaks redigeerimisrežiimis kasutatavatele klahvidele **Backspace** ja **Del** võimaldab Vi kasutada veel tervet rida käske teksti kustutamiseks, kopeerimiseks, asetamiseks ja asendamiseks käsurežiimis. Alltoodud käsud on jagatud õigupoolest kahte lehte: sooritatavad toimingud ja nende mõju. Toimingud võivad olla järgmised:

- **c**: asendamine (*Change* ehk 'muutmine'). Redaktor kustutab soovitud teksti ja naaseb käsu täitmise järel lisamisrežiimi.
- **d**: kustutamine (*Delete*).
- **y**: kopeerimine ("Yank" ehk 'paigalttõmbamine'). Sellest räägib lähemalt järgmine alajaotus.
- **.**: viimase toimingu kordamine.

Mõju määrab selle, millisele märgigrupile käsk mõjub.

- **h**, **j**, **k**, **l**: üks märk vasakule, alla, üles, paremale[3]
- **e**, **b**, **w**: aktiivse sõna lõppu ja algusesse ning järgmise sõna algusesse.
- **^**, **0**, **$**: aktiivse rea esimese mittetühimärgist märgini, aktiivse rea algusesse ja aktiivse rea lõppu.
- **<x>**: märgi **<x>** järgmisele esinemisele. Näiteks **fe** viib kursori märgi `e` järgmisele esinemiskohale.
- **/<sõne>**, **?<sõne>**: sõne või regulaaravaldise `<sõne>` järgmisele ja eelmisele esinemisele. Näiteks **/suva** viib kursori sõna `suva` järgmisele esinemiskohale.
- {, }; aktiivse lõigu algusesse ja lõppu.
- **G**, **H**: faili lõppu, ekraani algusesse.

Kõigile sellistele "mõjumärkidele" ehk liikumiskäskudele võib eelneda niinimetatud kordusearv. Näiteks **G** ("Go" ehk 'mine') korral tähistab see reanumbrit failis. Nii on võimalik moodustada kõikvõimalikke kombinatsioone.

Toome mõned näited:

- **6b**: liigub 6 sõna tagasi.
- **c8fk**: kustutab kogu teksti kuni märgi `k` kaheksanda esinemiseni ja naaseb siis lisamisrežiimi.
- **91G**: liigub faili 91. reale.
- **d3$**: kustutab aktiivse rea lõpuni ja veel kaks järgnevat rida.

Kuigi paljud käsud ei tundu esmapilgul väga loogilised ja näivad keerulistena, jäävad nad praktilisel kasutamisel üpris hõlpsasti külge. Aga kahtlemata andis juba seegi pisuke ülevaade aimu, et väljend "paari klahvivajutusega mägesid liigutama" pole tõest sugugi kaugel.

8.2.4. Lõikamine, kopeerimine, asetamine

Teksti kopeerimiseks kasutab Vi käsku, millega me juba tutvusime: **y**. Teksti lõikamiseks kasutage lihtsalt käsku **d**. Teksti salvestamiseks on ette nähtud 27 mälu ehk puhvrit: anonüümne mälu ja 26 mälu, mis on tähistatud inglise tähestiku väiketähtedega.

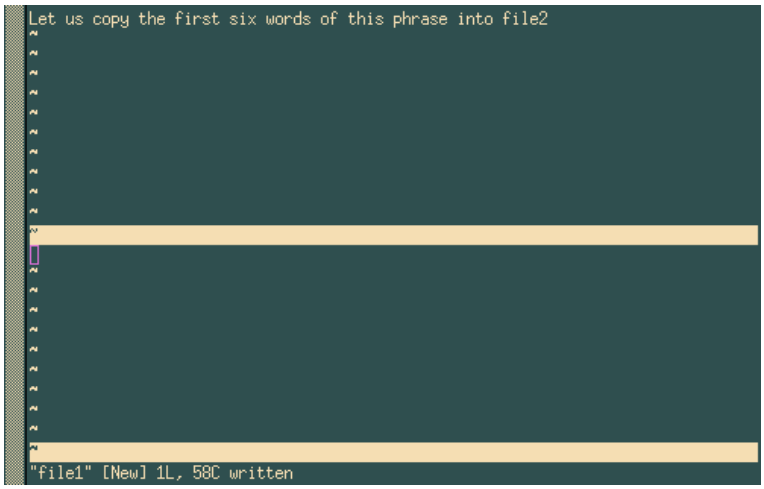
Anonüümse mälu kasutamiseks andke käsk "nii, nagu see on". See tähendab, et käsk **y12w** kopeerib anonüümsesse mällu 12 kursorile järgnevat sõna[4]. Kui soovite selle ala aga lõigata, andke käsk **d12w**.

26 nimega mälust mõne kasutamiseks lisage käsu ette **"<x>**, kus **<x>** määrab ära mälu nime. See tähendab, et eelmises näites toodud 12 sõna kopeerimiseks mällu `k` tuleb anda käsk **"ky12w** või **"kd12w** (kui soovite need sõnad lõigata).

Anonüümse mälu sisu asetamiseks kasutage käsku **p** või **P** (*Paste* ehk 'asetamine') teksti lisamiseks vastavalt kursori järele või ette. Nimega mälu sisu asetamiseks kasutage vastavalt käske **"<x>p** ja **"<x>P** (näitkes **"dp** asetab mälu `d` sisu kursori järele).

Vaatame seda näite varal:

3. **dl** (kustuta üks märk edasi) lühem variant on **x**; **dh** (kustuta üks märk tagasi) lühivariant on **X** ja **dd** kustutab aktiivse rea.
4. Aga ainult siis, kui kursor asub esimese sõna alguses!

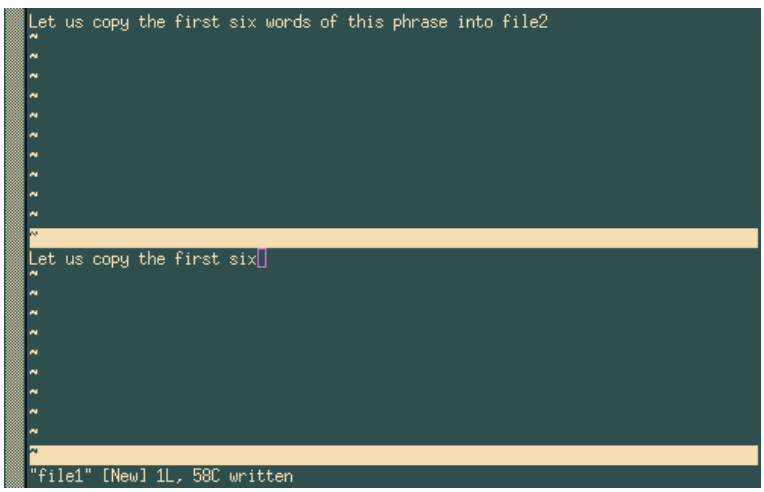


Joonis 8-5. VIM enne tekstiploki kopeerimist

Kirjapandud toimingu teostamiseks:

- kopeerige lause esimesed kuus sõna näiteks mällu `r`: **`"ry6w`**[5].
- liikuge allasuvasse puhvrisse `fail2`: **Ctrl-wj**.
- asetage mälu `r` sisu kursori ette: **`"rp`**.

See annab oodatud tulemuse, mida näitab Joonis 8-6.



Joonis 8-6. VIM pärast tekstiploki kopeerimist

Teksti otsimine on väga lihtne: käsurežiimis andke lihtsalt käsk **/** ja selle järel sõne, mida soovite otsida, ning vajutage klahvi **Enter**. Näiteks käsuga **`/pidu`** saab alates kursori asukohast asuda otsima sõnet `pidu`. Klahvi **n** vajutamisel saate liikuda sõne järgmisele esinemisele ja kui jõuate faili lõppu, alustatakse otsinguga uuesti selle algusest. Otsimiseks kursorist tahapoole kasutage käsu **/** asemel käsku **?**.

8.2.5. Vi'st väljumine

Väljumiskäsk on **`:q`** (õigupoolest sulgeb see käsk aktiivse puhvri, nagu juba eespool nägime, aga kui tegemist on ainsa avatud puhvriga, pannakse kinni ka Vi). Enamasti ongi tegemist ju üheainsa faili redigeerimisega. Sestap saate väljumiseks kasutada kas käsku

- **`:wq`** või **`:x`** muudatuste salvestamiseks ja väljumiseks (veel kiirem on **ZZ**) või
- **`:q!`** väljumiseks ilma salvestamata.

5. **`y6w`** ehk "Yank 6 words" tähendab otsetõlkes 'tõmba paigast 6 sõna'.

Pange tähele, et kui Teil on mitu puhvrit, salvestab ja sulgeb käsk **`:wq`** ainult aktiivse puhvri.

8.3. Viimane sõna...

Me oleme vahest rääkinud rohkemgi kui hädapärast vaja (lõppeks oli ju siht redigeerida tekstifaili), aga vahest saime siiski demonstreerida väbemalt mõningaid võimalusi, mida pakuvad käsitletud redaktorid. Nende kohta võiks kirjutada terveid raamatuid - mida on ka tehtud: mõlema redaktori kohta on lausa palju väga põhjalikke käsitlusi.

Võtke endale aega kogu siinse informatsiooni läbiseedimiseks, valige välja sobivamana paistev redaktor ja hakake seda kasutama, olgu siis kas või ainult vahetevahel huvi pärast. Igal juhul peaksite teadma, et kui Te vaid soovite edasi minna, siis on Teie ees kõik teed lahti.

Peatükk 9. Käsureautiliidid

Käesoleva peatüki eesmärk on tutvustada mõningaid käsureatööriistu, mis võivad tulla kasuks Teie igapäevatöös.

GNU/Linuxi üks suuremaid tugevusi on võimalus teha lihtsate tööriistadega ära ka väga keerulised ülesanded. Me rääkisime juba, kuidas siduda omavahel kokku mitu käsku ja puhastada väljund, nii et seda oleks hea vaadata (vt. Sektsioon 7.4). Nüüd on käes aeg õppida tundma mõningaid kasulikke tööriistu, mis lubavad Teil kõike märksa paremini kontrollid ja seeläbi tootlikkust suurendada.

Käesolev peatükk on mõeldud omamoodi harjutusena, et saaksite kõigi käskude funktsioonidest ja kasutamisest võimalikult hästi aru. Seepärast on iga käsu juures toodud ära vähemalt üks näide. Kui Teile peaks midagi siiski arusaamatuks jääma, vaadake kindlasti vastava käsu manuaalilehekülge. Need lõpevad alati sektsiooniga "VAATA KA" (või inglise keeles SEE ALSO), kus leiab viited muudele asjakohastele käskudele. See on koht, kus minna süvitsi oma uurimisretkel GNU/Linuxi maailmas!

9.1. Failioperatsioonid ja filtreerimine

Enamasti tegeldakse käsureal failidega. Käesolevas osas tutvustame, kuidas jälgida ja filtreerida faili sisu, kuidas hankida failist vajalik info üheainsa käsuga ja kuidas vähesega vaevaga sortida faili sisu.

9.1.1. cat, tail, head, tee: faili näitamise käsud

Neil käskudel on peaaegu täiesti ühesugune süntaks: `käsu_nimi [võtmed] [failid]` ja neid võib kasutada torus. Kõigi nendega saab lasta näidata faili mingit osa vastavalt teatud kriteeriumidele.

`cat` ühendab ehk konkateneerib failid ja näitab tulemusi standardväljundis, milleks on tavaliselt Teie arvuti ekraan. See on üks levinumaid käske. See võib välja näha näiteks selline:

```
# cat /var/log/mail/info
```

mis näitab e-posti deemoni logifaili sisu standardväljundis[1]. Käsul `cat` on üks väga kasulik võti (`-n`), millega saab näha ka reanumbreid.

Mõned failid, näiteks töötavate deemonite logifailid, võivad tihtipeale olla väga suured [2] ja nende täielik näitamine ekraanil sestap üsna mõttetu. Tavaliselt pakubki Teile ju huvi vaid mingi osa neist, teatud read. Selleks saab kasutada käsku `tail`. Nii näitab järgmine käsk vaikimisi faili `/var/log/mail/info` 10 viimast rida:

```
# tail /var/log/mail/info
```

Selliseid failid nagu logid muutuvad tavaliselt dünaamiliselt, sest antud logiga seotud deemon lisab logifaili pidevalt uusi toiminguid ja sündmusi. Kui soovite muudatusi interaktiivselt vaadata, kasutage võtit `-f`:

```
# tail -f /var/log/mail/info
```

Antud juhul näidatakse kõiki failis `/var/log/mail/info` toimuvaid muudatusi otsekohe ekraanil. Käsu `tail` kasutamine võtmega `-f` on üsna kasulik, kui soovite teada, kuidas Teie süsteem töötab. Nii saate näiteks logifaili `/var/log/messages` jälgides olla kursis süsteemi ja mitmesuguste deemonite teadetega.

Kui kasutate käsku `tail` enam kui ühe faili jaoks, näidatakse enne sisu omaette real faili nime. Ka selle korral saab kasutada võtit `-f` ning sellest on suur abi nägemaks, kuidas süsteemi erinevad osad omavahel suhtlevad.

Võtmega `-n` saab lasta näidata faili viimast n rida. Näiteks kahe viimase rea näitamiseks tuleb anda käsk sellisel kujul:

```
# tail -n2 /var/log/mail/info
```

Nagu käskude puhul ikka, saab korraga kasutada ka mitut võtit. Nii näiteks võite anda võtmed `-n2` ja `-f`, mis tähendab, et Teile näidatakse faili kaht viimast rida ja sealjuures uuendatakse neid vastavalt sellele, kuidas lisandub ridu jälgitavasse faili.

---

1. Mõnes käesolevas osas toodud näites kasutatakse reaalseid töö- ja serverilogide faile (teenused, deemonid jne.). Kontrollige nende järgimisel, et syslogd (see võimaldab logida deemoni tegevust) ja vastav deemon (meie näiteks Postfix) töötavad ja et te tegutsete administraatorina (`root`). Te võite muidugi alati ka toodud näiteid rakendada muude failide peal.
2. Fail `/var/log/mail/info` sisaldab infot kõigi saadetud kirjade kohta, teateid POP-protokolli kasutavate kasutajate kirjade tõmbamise kohta jne.

Käsk `head` on äärmiselt sarnane käsuga `tail`, ainult et sellega saab lasta näidata faili esimesi ridu. Järgmine käsk näitab vaikimisi faili `/var/log/mail/info` esimest 10 rida:

```
# head /var/log/mail/info
```

Nagu käsu `tail` puhul, saab ka siin võtmega `-n` määrata näidatavate ridade arvu. Näiteks kahe esimese rea näitamiseks andke käsk sellisel kujul:

```
# head -n2 /var/log/mail/info
```

Käske võib ka koos kasutada. Kui soovite näiteks lasta näidata ainult rida 9 ja 10, võite koostada käsu, milles `head` valib kõigepealt faili esimesed 10 rida ning edastab need siis toru kaudu käsule `tail`.

```
# head /var/log/mail/info | tail -n2
```

Viimane osa valib kaks viimast rida ja näitab neid ekraanil. Samalaadse koondkäsuga saab näidata näiteks 20. rida alates faili lõpust:

```
# tail -n20 /var/log/mail/info |head -n1
```

Selles näites anname käsule `tail` korralduse valida faili 20 viimast rida ja saata need toru kaudu käsule `head`. Käsk `head` näitab seejärel ekraanil saadud andmete esimest rida.

Oletame, et soovite näha ekraanil just viimati toodud näite rida ja selle siis failina `tulemused.txt` salvestada. Siin tuleb appi utiliit `tee`. See süntaks on järgmine:

```
tee [võtmed] [fail]
```

Nüüd võime modifitseerida varasemat käsku:

```
# tail -n20 /var/log/mail/info |head -n1|tee tulemused.txt
```

Toome veel ühe näite. Soovime valida 20 viimast rida ja salvestada need faili `tulemused.txt`, aga lasta ekraanil näidata 20 valitud reast ainult esimest. Selleks tuleb käsk anda järgmisel kujul:

```
# tail -n20 /var/log/mail/info |tee tulemused.txt |head -n1
```

Käsuga `tee` saab tarvitada väga kasulikku võtit `-a`, mis lubab lisada andmed juba olemasoleva faili lõppu.

Järgmises osas vaatame, kuidas kasutada käsku grep filtrina Postfixi teadete eraldamisel muudelt teenustelt pärit teadetest.

9.1.2. grep: sõnede leidmine failides

Ei käsu nimi ega lahtiseletus (“General Regular Expression Parser” ehk üldine regulaaravaldiste parsija) pole just paljuütlevad, kuid käsk ise on õigupoolest väga lihtne ja väga võimas: grep otsib Teie antud mustrit ühes või rohkemas failis. Süntaks on järgmine:

```
grep [võtmed] <muster> [üks või rohkem faili]
```

Kui anda mitu faili, siis tulemust näidates lisatakse alatu iga sobiva rea ette ka failinimi. Võtmega `-h` saab failinime näitamise keelata, võtmega `-l` aga lasta näidata ainult failinimesid. Muster kujutab endast regulaaravaldist, kuigi see võib olla ja enamasti ongi lihtsalt sõna või sõnaosa. Levinumad võtmed on järgmised:

- `-i`: tõstutundetu otsing (s.t. ei arvestata väike- ja suurtähe erinevust).
- `-v`: vastupidine otsing. Näidatakse ridu, mis **ei vasta** mustrile.
- `-n`: näidatakse iga leitud rea numbrit.
- `-w`: grep’ile antakse korraldus leida ainult mustriga sobivad terviksõnad.

Võtame nüüd uuesti ette e-posti deemoni logifaili. Oletame, et soovite leida failis `/var/log/mail/info` kõik read, millel leidub muster `postfix`. Selleks tuleb anda käsk:

```
# grep postfix /var/log/mail/info
```

Kui soovite leida kõik read, kus EI LEIDU mustrit `postfix`, tuleb kasutada võtit `-v`:

```
# grep -v postfix /var/log/mail/info
```

Torus saab kasutada käsku grep.

Oletame, et soovite leida kõik teated e-kirjade eduka saatmise kohta. Sel juhul tuleb filtreerida kõik read, mille lisas logifailile e-posti deemon (sisaldab mustrit `postfix`), kusjuures need peavad sisaldama teadet eduka saatmise kohta (`status=sent`)[3]:

```
# grep postfix /var/log/mail/info |grep status=sent
```

Antud juhul kasutatakse käsku grep kaks korda. See on küll lubatud, aga mitte just väga elegantne... Sama tulemuse võib saada utiliidiga `fgrep`. `fgrep` on õigupoolest seesama, mis käsk `grep -F`. Kõigepealt tuleb luua fail, milles leiduksid ühekaupa ridadele kirjutatud mustrid. Selle faili saab luua nii (me anname siinkohal failile nimeks `musrid.txt`):

```
# echo -e 'status=sent\npostfix' >./mustrid.txt
```

Kontrollige tulemust käsuga `cat`. `\n` on spetsiaalne muster tähendusega "new line" (ehk 'uus rida').

Seejärel kutsuge välja järgmine käsk, mis kasutab grep'i "topeltväljakutsumise" asemel faili `mustrid.txt` ja utiliiti `fgrep`:

```
# fgrep -f ./mustrid.txt /var/log/mail/info
```

Failis `./mustrid.txt` võib olla nii palju mustreid kui vaja. Näiteks selleks, et valida kirjad, mis saadeti edukalt aadressile `peter@mandriva.com`, tuleb see lihtsalt lisada faili `./mustrid.txt` järgmise käsuga:

```
# echo 'peter@mandriva.com' >>./mustrid.txt
```

Mõistagi om võimalik grep kombineerida käskudega `tail` ja `head`. Kui soovite näiteks leida teated eelviimase aadressile `peter@mandriva.com` saadetud kirja kohta, tuleb anda käsk:

```
# fgrep -f ./mustrid.txt /var/log/mail/info | tail -n2 | head -n1
```

Siin rakendatakse filter, nagu eespool kirjeldatud, ja edastatakse tulemus torusse käskudele `tail` ja `head`. Need üheskoos valivadki andmetest välja eelviimase väärtuse.

9.1.3. Regulaaravaldised ja filtreerimine egrep'iga

Käsu grep korral oleme seotud mustrite ja fikseeritud andmetega. Kuidas aga leida näiteks kõik e-irjad, mida on saadetud igale firma "ABC" töötajale? Kõigi nende aadresside ületähendamine pole sugugi lihtne, sest väga kergesti võib mõni puudu jääda või siis tuleb lausa logifaili käsitsi kaevuda.

Nagu `fgrep`'i korral, pakub grep välja omaette versiooni ka käsule `grep -E`: `egrep`. See kasutab mustrite asemel regulaaravaldisi, mis muudavad teksti "greppimise" palju tundlikumaks.

Lisaks metamärkide kasutamisele (neist rääkis lähemalt Sektsioon 7.3) saab kasutada veel mõningaid regulaaravaldis:

- `[:alnum:]` (kõik tähed ja kõik numrbrid), `[:alpha:]` (kõik suur- ja väiketähed) ja `[:digit:]` (kõik numbrid) on kasutatavad märgiklasside vahetu määratlemise asemel. Sellel on veel üks pluss: see arvestab ka muid kui inglise tähestiku tähti, samuti süsteemi lokaati.
- `[:print:]` tähistab kõiki märke, mida saab näidata ekraanil.
- `[:lower:]` ja `[:upper:]` tähistavad vastavalt kõiki väike- ja suurtähti.

Klasse on rohkemgi, nendega aitab Teil tutvuda manuaalikirje egrep(1). Siintoodud on lihtsalt kõige levinumad.

Regulaaravaldisele võib järgneda mõni kordusoperaator:

?

Eelnev element ei ole kohustuslik ja peab esinema mitte enam kui üks kord.

3. Kuigi on võimalik ka filtreerida lihtsalt staatuse mustri järgi, proovige ka meie näidet, sest me soovime siin näidata uut käsku.

*

Eelnev element peab esinema null või enam korda.

+

Eelnev element peab esinema üks või enam korda.

{n}

Eelnev element peab esinema täpselt `n` korda.

{n,}

Eelnev element peab esinema `n` või rohkem korda.

{n,m}

Eelnev element peab esinema vähemalt `n`, aga mitte rohkem kui `m` korda.

Kui panna regulaaravaldis sulgudesse, saab seda hiljem uuesti kasutada. Oletame, et määrasite avaldiseks `[:alpha:]+`. See võib tähistada näiteks sõna. Kui soovite tuvastada kaks korda esinevad sõnad, võite panna selle sulgudesse ja kasutada uuesti kordajaga `\1`, kui tegemist on esimese grupiga. Te saate ära kasutada kuni 9 sellist "mälu".

```
$ echo -e "abc def\nabc abc def\nabc1 abc1\nabcdef\nabcdabcd\nabcdef abcef" > testfile
$ egrep "([[:alpha:]]+) \1" testfile
abc abc def
$
```

Märgid `[` ja `]` kuuluvad grupinimesse, mistõttu need tuleb kaasata antud märgiklassi. Esimene `[` annab teada, et me kasutame märgigruppi, teine on osa grupinimest ning seejärel tulevad mõlemale vastavad sulgevad märgid `]`.

Ainuke tagastatud rida ongi selline, mis ainsana vastas kahele tühikuga eraldatud tähegrupile. Ükski teine grupp regulaaravaldisega ei sobinud.

Kasutada võib ka märki `|` kas vasakul pool `|` või sellest paremal pool asuva avaldise sobivuse leidmiseks. See operaator ühendab need avaldised. Püüame eespool loodud faili `testfail` kasutades leida avaldised, mis sisaldavad ainult topeltsõnu või topeltsõnu numbritega:

```
$ egrep "([[:alpha:]]+) \1|([[:alpha:][:digit:]]+) \2" testfail
abc abc def
abc1 abc1
$
```

Pange tähele, et teise sulgusid kasutava grupu puhul oli vajalik kasutada `\2`, sest muidu poleks leitud meile vajalikke sobivusi. Õigupoolest oleks antud juhul tõhusam regulaaravaldis selline:

```
$ egrep "([[:alnum:]]+) \1" testfail
abc abc def
abc1 abc1
$
```

Lõpuks tuleb arvestada, et teatud märkide leidmiseks tuleb need "varjestada" ehk nende ette längkriips lisada. Neiks märkideks on: `?`, `+`, `{`, `|`, `(`, `)` ning mõistagi ka `\`. Nende leidmiseks tuleb kirjutada: `\?`, `\+`, `\{`, `\|`, `\(`, `\)` ja `\\`.

See lihtne nõks aitab Teil vältida korduvate sõnade kirjutamist "Teie Teie" tekstis.

Kõik tööriistad järgivad regulaaravaldiste kasutamisel toodud või nendega väga sarnaseid reegleid. Kui Te kulutate veidi aega nende omandamisele, aitab see Teid tunduvalt paljude tööriistade, näiteks `sed`'i juures. `sed` võimaldab Teil teksti töödelda ja muu hulgas seda muuta ka regulaaravaldisi kasutades.

9.1.4. wc: faili elementide loendamine

Käsku `wc` (*Word Count* ehk 'sõnade arv') kasutatakse faili ridade, sõnede ja sõnade arvu leidmiseks. Sellest on abi ka pikima rea pikkuse leidmisel. Süntaks on järgmine:

```
wc [võtmed] [failid]
```

Kasutada saab järgmisi võtmeid:

- `-l`: ridade arvu näitamine.
- `-w`: sõnade arvu näitamine.
- `-m`: märkide koguarvu näitamine.
- `-c`: baitide arvu näitamine.
- `-L`: teksti pikima rea pikkuse näitamine.

Vaikimisi näitab `wc` ridade, sõnade ja märkide arvu. Toome mõned näited:

Kui me soovime teada saada süsteemi kasutajate arvu, on käsk selline:

```
$ wc -l /etc/passwd
```

Kui me soovime teada saada süsteemi CPU-de arvu, on käsk selline:

```
$ grep "model name" /proc/cpuinfo |wc -l
```

Eelmises osas hankisime teadete nimekirja edukalt failis `./mustrid.txt` loetletud aadressidele saadetud e-kirjade kohta. Kui soovime teada, mitu teadet see sisaldab, võib filtri tulemuse toru kaudu suunata käsule `wc`:

```
# fgrep -f ./mustrid.txt /var/log/mail/info | wc -l
```

9.1.5. sort: faili sisu sortimine

Võimsa sortimisutiliidi süntaks on järgmine[4]:

```
sort [võtmed] [failid]
```

Vaatleme faili `/etc/passwd` sortimist. Nagu näete, pole see sorditud:

```
$ cat /etc/passwd
```

Kui me soovime selle sortida kasutajanime välja (`login`) alusel, tuleb anda käsk:

```
$ sort /etc/passwd
```

Käsk `sort` sordib andmed vaikimisi kasvavas järjestuses esimese välja alusel (meie näites on selleks kasutajanimi ehk `login`). Kui soovime sortida andmed kahanevas järjestuses, tuleb kasutada võtit `-r`:

```
$ sort -r /etc/passwd
```

Kõigil kasutajatel on oma `UID`, mis on pandud kirja faili `/etc/passwd`. Järgmise käsuga saab sortida faili kasvavas järjestuses välja `UID` järgi:

```
$ sort /etc/passwd -t":" -k3 -n
```

Me kasutasime järgmisi käsu `sort` võtmeid:

- `.t":"`; annab käsule `sort` teada, et väljaeraldajaks on sümbol `":"`.
- `-k3`: sortimine tuleb sooritada kolmanda veeru järgi.
- `-n`: annab teada, et sortimine tuleb ette võtta arvandmetega, mitte tähestiku järgi.

4. Me tutvustame käsku `sort` siin ainult lühidalt, kuigi selle kohta võiks kirjutada terve paksu raamatu.

Sama käsu võib anda ka teistpidi järjestuses:

```
$ sort /etc/passwd -t":" -k3 -n -r
```

Pange tähele, et käsul `sort` on veel kaks tähtsat võtit:

- `-u`: range järjestus - topeltväljad jäetakse näitamata.
- `-f`: tõstu eiramine (väiketähti peetakse võrdseks suurtähtedega).

Kui me tahame leida suurima `UID`-ga kasutajat, tuleb anda käsk:

```
$ sort /etc/passwd -t":" -k3 -n |tail -n1
```

millega me sordime faili `/etc/passwd` kasvavas järjestuses veeru `UID` järgi ning suuname tulemusel toru kaudu käsule `tail`. See näitab sorditud nimekirja esimest väärtust.

9.2. find: failide leidmine teatud kriteeriumi alusel

`find` on pika ajalooga UNIX® utiliit. Selle funktsioon on uurida rekursiivselt läbi üks või rohkem kataloogi ning leida sealt failid, mis vastavad teatud kriteeriumidele. Kuigi see on väga kasulik käsk, on selle süntaks mõnevõrra keeruline ja nõuab harjutamist. Üldiselt näeb süntaks välja selline:

```
find [võtmed] [kataloogid] [kriteerium1] ... [kriteeriumN] [toiming]
```

Kui kataloogi ei anta, otsib `find` läbi aktiivse kataloogi. Kui kriteerium andmata jätta, arvestatakse vaikimisi kriteeriumiks “tõene”, mis tähendab, et leitakse üles kõik failid. Võtmeid, kriteeriume ja toiminguid on nii palju, et siinkohal on võimalik neist ainult mõnel peatuda. Kõigepealt mõned võtmed:

- `-xdev`: otsingut ei sooritata muude failisüsteemide kataloogides.
- `-mindepth <n>`: enne failide otsimist liigutakse vähemalt `n` taset määratud kataloogist allapoole.
- `-maxdepth <n>`: otsitakse faile, mis asuvad määratud kataloogist maksimaalselt `n` taset allpool asuvates kataloogides.
- `-follow`: kui nimeviidad osutavad kataloogidele, järgitakse neid. Vaikimisi `find` viitu ei järgi.
- `-daystart`: ajaga seotud testide korral (vt. allpool) arvestatakse ajatemplina käesoleva päeva algust, mitte vaikeväärtust (24 tundi enne käesolevat hetke).

Kriteeriumiks võib olla üks või mitu ***atomaarset*** testi. Mõned kasulikumad testid:

- `-type <failitüüp>`: antud tüübiga faili otsimine. `failitüüp` võib olla: `f` (tavaline fail), `d` (kataloog), `l` (nimeviit), `s` (sokkel), `b` (plokkseadmefail), `c` (sümbolseadmefail) või `p` (toru).
- `-name <muster>`: antud mustrile vastava nimega failide otsimine. Selle võtme puhul koheldakse mustrit ***metamärgina*** (vt. Sektsioon 7.3).
- `-iname <muster>`: nagu `-name`, aga ei arvesta tõstu ehk tähesuurust.
- `-atime <n>`, `-amin <n>`: failide otsimine, mida on viimati kasutatud `n` päeva eest (`-atime`) või `n` minuti eest (`-amin`. Samuti võib anda `<+n>` või `<-n>`, mille korral otsitakse faile, mida on kasutatud vastavalt kas hiljem või varem kui `n` päeva/minuti eest.
- `-anewer <fail>`: failide otsimine, mida on kasutatud hiljem kui faili `fail`.
- `-ctime <n>`, `-cmin <n>`, `-cnewer <fail>`: sama, mis `-atime`, `-amin` ja `-anewer`, kuid otsitakse faile, mille sisu on vastavatel aegadel muudetud.
- `-regex <muster>`: sama, mis `-name`, kuid `muster` tähendab antud juhul ***regulaaravaldist***.
- `-iregex <muster>`: sama, mis `-regex`, aga tõst jäetakse arvestamata.

Teste on veel päris palju, neist annab ülevaate manuaalilehekülg find(1). Testide kombineerimiseks võib kasutada üht järgnevatest võimalustest:

- `<k1> -a <k2>`: tõene, kui nii `k1` kui ka `k2` on tõesed; `-a` on tegelikult vaikimisi eeldatav, mistõttu võib ka kirjutada lihtsalt `<k1> <k2> <k3>`, kui soovite, et sobiksid kõik testid, nii `k1`, `k2` kui ka `k3`.
- `<k1> -o <k2>`: tõene, kui tõene on kas `k1` või `k2` või mõlemad. Arvestage, et võtmel `-o` seisab ***tehtejärjestuses*** madalamal kui `-a`, seepärast tuleb näiteks juhul, kui soovite leida faile, mis sobiksid kriteeriumile `k1` või `k2` ja ka kriteeriumile `k3`, kasutada sulge ning kirjutada see kujul `( <k1> -o <k2> ) -a <k3>`. Teil tuleb ühtlasi sulud ***varjestada*** (mitteaktiivseks muuta), sest muidu üritab shell neid oma reeglite järgi tõlgendada!
- `-not <k1>`: test `k1` tagurpidi, mis tähendab, et `-not <k1>` on tõene juhul, kui `k1` on väär.

Lõpuks saab määrata ka toimingu, mida leitud faili(de)ga ette võtta. Sagedasemad toimingud on järgmised:

- `-print`: kõigi failide nime näidatakse standardväljundis. See on vaiketoiming.
- `-ls`: standardväljundis näidatakse iga leitud faili kohta seda, mida näitaks käsk `ls -ilds`.
- `-exec <käsurida>`: käivitab kõigi leitud failide puhul käsu `käsurida`. Käsurida `käsurida` peab lõppema märgiga `;`, mis tuleb varjestada, et shell seda omapäi tõlgendama ei asuks. Faili asukohta märgib `{}`. Vt. kasutamisnäiteid.
- `-ok <käsk>`: sama, mis `-exec`, aga pärib iga käsu korral kinnitust.

Ilmselt parim viis kõiki võtmeid ja parameetreid tundma õppida on näidete varal. Oletame, et soovime leida kõik kataloogid kataloogis `/usr/share`. Selleks tuleb anda käsk:

```
find /usr/share -type d
```

Oletame, et Teil on HTTP-server. Kõik Teie HTML-failid asuvad kataloogis `/var/www/html`, kus Te ka parasjagu viibite. Te soovite leida kõik failid, mille sisu pole muudetud kuu aja jooksul. Kuna Teil on mitmeid kirjutamisõigusega kasutajaid, kannavad osad failid laiendit `html`, osad aga `htm`. Te soovite linkida leitud failid kataloogi `/var/www/obsolete`. Selleks tuleb Teil anda käsk[5]:

```
find \( -name "*.htm" -o -name "*.html" \) -a -ctime -30 \
-exec ln {} /var/www/obsolete \;
```

See on suhteliselt keeruline näide, mis vajab ilmselt seletust. Kriteerium on järgmine:

```
\( -name "*.htm" -o -name "*.html" \) -a -ctime -30
```

ja teeb just seda, mis vaja: leiab kõik failid, mille nime lõpus seisab kas `.htm` või `.html` ("`\( -name "*.htm" -o -name "*.html" \)`"), **ja** (`-a`) mida ei ole muudetud viimase 30 päeva jooksul, mis ongi enam-vähem kuu aega (`-ctime -30`). Pange tähele sulge: need on siin vajalikud, sest `-a` seisab tehtejärjestuses kõrgemal. Kui neid ei oleks, leitkas kõik failid, mille lõpus seisab `.htm`, ning lisaks kõik failid, mille lõpus seisab `.html` ja mida ei ole muudetud viimase kuu aja jooksul, mis aga ei vastaks kuidagi meie soovile. Pange tähele ka seda, et sulud on shelli eest varjestatud: kui kirjutada `\( ... \)` asemel `( ... )`, hakkaks shell neid oma reeglite järgi tõlgendama ning üritaks käivitada alamshellis `-name "*.htm" -o -name "*.html"`... Teine võimalus oleks panna sulud topelt- või ühekordsete jutumärkide vahele, aga längkriips on soovitatavam, sest sel juhul saab läbi ajada ainult ühe märgiga.

Lõpuks on siin ka käsk, mida rakendatakse kõigile failidele:

```
-exec ln {} /var/www/obsolete \;
```

Ka siin tuleb märk `;` shell'i eest varjestada. Vastasel juhul tõlgendaks shell seda käsueraldajana. Kui Teil peaks see meelest minema, hoiatab `find` teid, et `-exec` on argumendita.

Viimane näide: Teil on tohutu kataloog (`/shared/pildid`), kus leidub kõikvõimalikke pilte. Te kasutate regulaarselt käsku `touch` kataloogis leiduva faili `tempel` aja uuendamiseks, nii et Teil on olemas aeg, millele toetuda. Te soovite nüüd leida kõik ***JPEG***-pildid, mis on uuemad kui fail `tempel`, aga kuna Te olete hankinud pilte erinevatest allikatest, on neil laienditeks nii `jpg`, `jpeg` kui ka `JPG` ja `JPEG`. Samuti soovite jätta otsingul kõrvale kataloogi `vanad`. Lisaks soovite, et failinimekiri saadetakse Teile e-kirjana (ning Teie kasutajanimi on **`peter`**):

```
find /shared/pildid -cnewer     \
     /shared/pildid/tempel       \
     -a -iregex ".*\.jpe?g"     \
```

5. Arvestage, et toodud näite korral on vajalik, et `/var/www` ja `/var/www/obsolete` asuksid samas failisüsteemis!

```
    -a -not -regex ".*/vanad/.*" \
      | mail peter -s "Uued pildid"
```

Selline käsk on mõistagi tülikas, kui Te peate iga kord kogu joru käsitsi kirja panema - ja Teil läheb seda sageli vaja. Sellise probleemi lihtsaks lahenduseks on lasta käsul perioodiliselt tööle hakata cron-deemoni abil, millest tuleb juttu järgmises osas.

9.3. Käskude käivitamise ajastamine

9.3.1. crontab: oma crontab-faili uurimine või muutmine

`crontab` võimaldab Teil käivitada käske regulaarse ajavahemiku järel, kusjuures Te ei pea isegi olema ennast sisse loginud. `crontab` võib käsu väljundi e-postiga Teile saata. Te saate määrata intervalli minutites, tundides, päevades või isegi kuudes. Sõltuvalt võtmetest toimib `crontab` omajagu erinevalt:

- `-l`: olemasoleva `crontab`-faili näitamine.
- `-e`: `crontab`-faili redigeerimine.
- `-r`: olemasoleva `crontab`-faili eemaldamine.
- `-u <kasutaja>`: mõne mainitud võtme rakendamine kasutajale `<kasutaja>`. Seda saab teha ainult administraator (`root`).

Alustame kohe faili `crontab` redigeerimisega. Kui annate käsu `crontab -e`, ilmub Teie lemmiktekstiredaktor, kui olete selle määranud keskkonnamuutujaga `EDITOR` või `VISUAL`, vastasel juhul avatakse Vi. Faili `crontab` rida koosneb kuuest väljast. Esimesed viis välja märgivad vastavalt minutite, tundide, päevade, kuude ja nädalapäevade intervalle. Kuues väli sisaldab käivitatavat käsku. Ridasid, mille alguses seisab `#`. peetakse kommentaariks `crond` (programm, mis vastutab `crontab`-failides kirjapandu täitmise eest) ignoreerib neid. Faili ülesehitus on natuke teistsugune süsteemse `crontab`-faili puhul (selle asukohaks on `/etc/crontab`). Selles seisab kuuendal väljal kasutajanimi, kelle õigustes käivitatakse seitsmendal väljal antud programm. Seda tuleks kasutada ainult haldamisülesanneteks ning selliste kasutajate tööde käivitamiseks, mis ongi mõeldud süsteemi turvalisuse suurendamiseks (näiteks viirustõrje-kasutaja või siis andmebaasiserveri käimashoidmiseks loodud kasutaja). Toome näite `crontab`-faili kohta:

Loetavaks muutmiseks tuli meil pikad read murda. Seepärast tuleb arvestada, et mõned osad peavad tegelikult asuma ühel real. Kui rida lõpeb märiga `\`, tähendab see, et rida jätkub. Sellist konventsiooni kasutatakse nii `Makefile`-failides ja shellis kui ka mitmel pool mujal. Samuti on siin eestindatud ka sellised kommentaarid, mis tavaliselt esinevad inglise keeles.

```
# Kui Te ei soovi saada e-kirja, muutke
#   järgmine rida kommentaariks
#MAILTO="Teie_e-posti_aadress"
#
# Iga 2 päeva tagant antakse 14.00 teada uutest piltidest,
#   nagu eespool selgitatud - seejärel antakse failile
#   "tempel" taas käsk "touch". Märki
#   "%" tõlgendataksereavahetusena, mis lubab
#   anda ühel real mitu käsku.
0 14 */2 * *  find /shared/pildid              \
  -cnewer /shared/pildid/tempel                \
  -a -iregex ".*\.jpe?g"             \
  -a -not -regex                               \
    ".*/old/.*"%touch /shared/images/stamp
#
# Igal jõululaupäeval mängitakse muusikat :)
0 0 25 12 * mpg123 $HOME/sounds/merryxmas.mp3
#
# Iga teisipäev 17.00 trükitakse välja ostunimekiri...
0 17 * * 2 lpr $HOME/ostunimekiri.txt
```

Intervalle saab mõistagi määrata väga mitmel moel ja toodud näide on ainult üks võimalus. Te võite määrata ***diskreetsed väärtused***, mille eraldamiseks tuleb kasutada komasid (`1,14,23`) või vahemiku (`1-15`) või ka

neid kombineerida (`1-10,12-20`), määrates soovi korral ka sammu (`1-12,20-27/2`). Ja siis tuleb lihtsalt leida käsud, mida nende vahvate intervallide järel käivitada!

9.3.2. at: käsu ajastamine ainult üheks korraks

Võimalik, et soovite käivitada mingi käsu konkreetsel päeval, aga mitte regulaarselt. Võib-olla soovite näiteks tuletada endale meelde, et Teid ootab õhtul kell kuus ees kohtumine. Teil töötab X, paigaldatud on tarkvarapakett `X11R6-contrib` ning Te soovite, et Teile antaks 17.30 teada, et on aeg liikuma hakata, et mitte kohtumisele hilineda. Siin astub mängu `at`:

```
$ at 5:30pm
 # Teie ees seisab nüüd "at" viip
at> xmessage "Aeg minna! Kohtumine on kell 18.00"
 # Väljumiseks andke käsk CTRL-d
at> <EOT>
job 1 at 2005-02-23 17:30
$
```

Aega saab määrata mitmel moel:

- `now + <intervall>`: see tähendabki hetke 'praegu' ja intervalli (lisavõimalus; intervalli puudumine tähendab lihtsalt praegust hetke). Intervalli süntaks on `<n> (minutes|hours|days|weeks|months)` vastavalt minutite, tundide, päevade, nädalate ja kuude jaoks. Nii võib intervalliks määrata näiteks `now + 1 hour` (ühe tunni pärast), `now + 3 days` (kolme päeva pärast) jne.
- `<kellaaeg><päev>`: daatumi täielik määramine. Parameeter `<kellaaeg>` on kohustuslik. `at` on selle puhul üsna liberaalne: Te võite kellaajaks anda `0100`, `04:20`, `2am`, `0530pm`, `1800` või mõne kolmest eriväärtusest: `noon` (keskpäev), `teatime` (16.00) või `midnight` (kesköö). Parameeter `<päev>` pole kohustuslik. Ka selle võib määrata mitmel kujul: `12/20/2001` (20.detsember 2001) või `20.12.2001`. Te võite ka aasta ära jätta, aga siis saate kasutada ainult viimast, niinimetatud Euroopa märkimisviisi: `20.12`. Kuu võib määrata ka tähtedega: sobivad nii `Dec 20` kui ka `20 Dec`.

`at` tunnustab ka mitmeid võtmeid:

- `-l`: näitab parajasti järjekorras tööde nimekirja, kus esimesel väljal seisab töö number. See on sama, mis käsk `atq`.
- `-d <n>`: järjekorrast eemaldatakse töö numbriga `<n>`. Numbri saab tuvastada käsuga `atq`. See on sama, mis käsk `atrm <n>`.

Nagu ikka, tutvustab kõiki võtmeid ja muud lähemalt `manuaalilehekülg` at(1).

9.4. Arhiveerimine ja andmete tihendamine

9.4.1. tar: lindiarhiveerija (Tape ARchiver)

Nagu `find`, on ka `tar` väga vana UNIX® utiliit, mistõttu selle süntaks on omajagu spetsiifiline. See näeb välja selline:

```
tar [võtmed] [failid...]
```

Toome siin ära mõned võtmed. Pange tähele, et kõigil neil on ka vastavad pikemad analoogid, aga selleks vaadake juba manuaalilehekülge tar(1).

`tar` ei kasuta enam lühivõtmete puhul eelnevat kriipsu `-`, seda kasutatakse ainult pikkade võtmete korral.

- `c`: uute arhiivide loomiseks.

- `x`: failide ekstraktimiseks olemasolevast arhiivist.
- `t`: olemasoleva arhiivi failide nimekirja loomiseks.
- `v`: niinimetatud jutukuse suurendamine. Näidatakse arhiivi lisatavate või sealt ekstraktitavate failide nimesid. Kui seda kasutada koos võtmega `t` (vt. eespool), näidatakse lühikese failide nimekirja asemel pikka.
- `f <failinimi>`: arhiivi loomine nimega `failinimi`, ekstraktib failid arhiivist `failinimi` või näitab arhiivi `failinimi` failide nimekirja. Kui see parameeter ära jätta, on vaikefailiks `/dev/rmt0`, mis on üldjuhul spetsiaalne ***striimeriga*** seotud fail. Kui failiparameetriks on – (kriips), seostatakse sisend või väljund (sõltuvalt sellest, kas loote arhiivi või ekstraktite) standardsisendi või -väljundiga.
- `z`: annab `tar`'ile teada, et loodav arhiiv tuleb tihendada `gzip`'iga või et ekstraktitav arhiiv on tihendatud `gzip`'iga.
- `j`: sama, mis `z`, ainult et tihendamiseks kasutatakse programmi `bzip2`.
- `p`: failide ekstraktimisel arhiivist säilitatakse kõik faili atribuudid, kaasa arvatud omanik, viimase kasutamise aeg ja nii edasi. Väga kasulik failisüsteemi tõmmiste korral.
- `r`: lisab käsureal antud failide nimekirja olemasolevale arhiivile. Arvestage, et arhiiv, millele soovite faile lisada, **ei tohi** olla tihendatud!
- `A`: lisab käsureal antud arhiivid arhiivi, mis antakse võtmega `f`. Sarnaselt võtmele `r` ei tohi ka selle puhul arhiivid olla tihendatud.

Võtmeid on veel terve hulk, kõigi nendega aitab tutvuda manuaalilehekülg tar(1). Vaadake näiteks võtit `d`.

Võtame nüüd ette näite. Oletame, et soovite luua arhiivi kõigi kataloogis `/shared/pildid` leiduvate piltidega, tihendada selle programmiga `bzip2`, anda sellele nimeks `pildid.tar.bz2` ning paigutada loodud faili kataloogi `/home`. Selleks tuleb anda selline käsk:

```
 #
 # Märkus: Te peate viibima kataloogist, mille faile
 #   soovite arhiveerida!
 #
$ cd /shared
$ tar cjf ~/pildid.tar.bz2 pildid/
```

Nagu näete, kasutasime siin kolme võtit: `c` andis `tar`'ile teada, et soovime luua arhiivi, `j` andis korralduse tihendada see `bzip2`'iga ja `f ~/pildid.tar.bz2` korralduse luua arhiiv meie kodukataloogis nimega `pildid.tar.bz2`. Nüüd kontrollime, kas loodud arhiiv on ikka korrektne. Selleks saab lasta näiteks loetleda selle failid:

```
 #
 # Suunduge oma kodukataloogi
 #
$ cd
$ tar tjvf pildid.tar.bz2
```

Siin andsime `tar`'ile korralduse loetleda (`t`) failid arhiivis `pildid.tar.bz2` (`f pildid.tar.bz2`), hoiatasime, et arhiiv on tihendatud `bzip2`'iga (`j`) ja teatasime, et soovime pikka nimekirja (`v`). Oletame nüüd, et kustutasite kogemata või mingil põhjusel oma pildikataloogi. Õnneks on Teil aga arhiiv alles ja soovite nüüd selle ekstraktida, et saada failid tagasi sinna, kus nad algselt olid (`/shared` alamkataloogis). Aga kuna Te soovite ka edaspidi kasutada `find`'i uute piltide leidmise käsku, siis on vaja säilitada ka kõik failide atribuudid:

```
 #
 # cd kataloogi, kuhu soovite arhiivi ekstraktida
 #
$ cd /shared
$ tar jxpf ~/pildid.tar.bz2
```

Ja asi ants!

Oletame nüüd, et soovite arhiivist ekstraktida ainult ja ainult kataloogi `/pildid/autod`. Selleks tuleb anda käsk:

```
$ tar jxf ~/pildid.tar.bz2 pildid/autod
```

Kui üritate varudanda spetsiaalseid faile, võtabki `tar` neid spetsiaalsete failidena ega ürida nende sisust tõmmist teha, Sestab võite julgelt arhiveerida näiteks `/dev/mem`. Programm tuleb väga edukalt toi-

me ka viitadega, nii et needki ei valmista muret. Nimeviitade kohta tasuks tutvuda võtmega `h` (vaadake `manuaalilehekülg`e).

9.4.2. bzip2 and gzip: andmete tihendamise programmid

Me puutusime nende utiliitidega kokku juba `tar`'i juures. Erinevalt näiteks Windows® keskkonnas kasutatavast rakendusest WinZip® käib arhiveerimine ja tihendamine kahe erineva utiliidiga: `tar` on mõeldud arhiveerimiseks ja kaks programmi, mida kohe tutvustame, tihendamiseks: `bzip2` ja `gzip`. Te võite muidugi kasutada ka muid tihendamistööriistu: GNU/Linuxile on olemas ka `zip`, `arj` ja `rar`, kuigi neid kasutatakse harva.

Algselt loodi `bzip2` utiliidi `gzip` asendajaks. Selle tihendustase on üldiselt parem, kuid samas nõuab see töötamise ajal märksa rohkem ressursse. Vanemate süsteemidega ühilduvuse tagamiseks on `gzip` endiselt kasutusel.

Mõlemal käsul on sarnane süntaks:

```
gzip [võtmed] [failid]
```

Kui failinimi andmata jätta, pöörduvad nii `gzip` kui `bzip2` andmeid otsides standardsisendi poole ja saadavad oma tulemuse standardväljundisse. Seepärast saab mõlemat programmi kasutada ka torudes. Mõlemal on ka ühised võtmed:

- `-1` ... `-9`: tihendustaseme määramine. Mida suurem arv, seda parem tihendus, aga parem tähendab ka aeglasemat.
- `-d`: failide lahtipakkimine. See võrdub programmide `gunzip` või `bunzip2` kasutamisega.
- `-c`: parameetrina antud failide tihendamise/lahtipakkimise tulemus saadetakse standardväljundisse.

Vaikimisi kustutavad nii `gzip` kui ka `bzip2` faili(d), mida nad tihendavad või lahti pakivad, kui Te ei kasuta just võtit `-c`. `bzip2` korral saab seda vältida võtmega `-k`, `gzip`'il sellele vastavat võtit pole.

Võtame nüüd ette näited. Oletame, et soovite tihendada aktiivses kataloogis kõik laiendiga `.txt` failid ja kasutada selleks programmi `bzip2` maksimaalse tihendusega. Selleks tuleb anda käsk:

```
$ bzip2 -9 *.txt
```

Nüüd oletame, et soovite jagada kellegagi oma pildiarhiivi, aga tal ei ole programmi `bzip2`, vaid ainult `gzip`. Selleks ei ole vaja arhiivi lahti ja uuesti kokku pakkida, vaid selle võib lahti pakkida standardväljundisse, suunata see torusse, tihendada standardväljund ja suunata väljund uude arhiivi. See käib nii:

```
bzip2 -dc pildid.tar.bz2 | gzip -9 >pildid.tar.gz
```

Te võite ka `bzip2 -dc` asemel kirjutada `bzcat`. Programmile `gzip` on samuti samasugune asendaja olemas, aga selle nimi on `zcat`, mitte `gzcat`. Kui soovite tihendatud faile mitte lahti pakkida, vaid neid otse vaadata, võite kasutada programme `bzless` `bzip2` ja `zless` `gzip` puhul. Proovige nüüd näiteks leida iseseisvalt käsk, milega vaadata tihendatud faile ilma neid lahti pakkimata ja ilma programme `bzless` või `zless` kasutamata.

9.5. Veel palju-palju muud...

Käske on sedavõrd palju, et neid vähegi põhjalikumalt käsitleda üritav raamat annaks välja vähemalt väga paksu, kui mitte mitmeköitelise entsüklopeedia mõõdu. Käesolevas peatükis ei jõudnud me isegi kümnendikuni temaatilisest materjalist, aga juba sellega, mida me põgusalt puudutasime, võite Te väga palju ära teha. Soovi korral võite tutvuda mõningate `manuaal`lehekülgedega: sort(1), sed(1) ja zip(1L) (jah, just nii: Te võite GNU/Linuxis lahti pakkida või luua ka `.zip`-faile), convert(1) ja nii edasi. Kõige paremini saab neid käske mõistagi tundma õppida praktikas ning eksperimenteerimisel võite avastada palju kasulikku ja huvitavat, võib-olla isegi ootamatu. Proovige ja Te ei kahetse!

Peatükk 10. Protsesside juhtimine

Me rääkisime juba, mida protsessid endast kujutavad (Sektsioon 1.3). Nüüd vaatame, kuidas protsessidest ja nende iseloomust ülevaade saada ning kuidas nendega midagi ette võtta.

10.1. Täpsemalt protsessidest

Protsesse on võimalik jälgida ning neid saab surmata, peatada, lasta edasi töötada jne. Et paremini mõista edaspidi esinevaid näiteid, tuleb meil esmalt protsessidega veidi lähemalt tutvuda.

10.1.1. Protsessipuu

Nagu failid, on ka kõik GNU/Linuxis töötavad protsessid korraldatud puustruktuuri. Selle juureks on süsteemne protsess `init`, mis käivitatakse algkäivituse ajal. Süsteem omistab igale protsessile numbri (PID, *Process ID* ehk 'protsess ID'), et seda oleks võimalik unikaalselt tuvastada. Protsessid pärivad ka oma eellase PID (PPID, *Parent Process ID* ehk 'eellasprotsessi ID'). `init` on selles mõttes omaenda eellane: `init` PID ja PPID on 1.

10.1.2. Signaalid

Iga UNIX® protsess reageerib talle saadetavatele signaalidele. Kokku on 64 signaali, mida tuvastatakse kas nende numbri (alustades `1`) või sümbolnime järgi (`SIGx`, kus `x` on signaali nimi). 32 "kõrgemat" signaali (`33` kuni `64`) on reaalajasignaalid ning nende käsitlemine väljub käesoleva peatüki raamidest. Kõigi signaalide korral määrab protsess ise oma reaktsiooni, välja arvatud kahe signaali korral: number 9 (`KILL`) ja number 19 (`STOP`).

Signaal `9` lõpetab protsessi pöördumatult ega anna sellele aega isegi ennast kõiki reegleid täites sulgeda. See on signaal, mis saadetakse kinnijooksnud või mingeid tõsiseid probleeme tekitavale protsessile. Signaalide täielikku nimekirja saab näha käsuga `kill -l`.

10.2. Protsesside info: ps ja pstree

Need kaks käsku näitavad süsteemis parajasti töötavate protsesside nimekirja vastavalt Teie määratud kriteeriumile. `pstree` väljund on võrreldes käsuga `ps -f` mõnevõrra selgem.

10.2.1. ps

`ps` käivitamine ilma argumentideta näitab ainult protsesse, mille Te ise olete algatanud ja mis on seotud Teie kasutatava terminaliga:

```
$ ps
   PID TTY          TIME CMD
 18614 pts/3    00:00:00 bash
 20173 pts/3    00:00:00 ps
```

Nagu enamikul UNIX® utiliitidel, on ka `ps`'il rida võtmeid, millest levinumad on järgmised:

- `a`: kõigi kasutajate käivitatud protsesside näitamine.
- `x`: protsesside näitamine, mida ei kontrolli ükski terminal või kontrollib mõni muu terminal kui see, mida Te kasutate.
- `u`: kõigi protsesside juures näidatakse selle käivitanud kasutaja nime ja käivitamise aega.

Võtmeid on veel palju, neist annab täieliku ülevaate manuaalilehekülg ps(1).

`ps` väljund jaguneb mitmeks väljaks, millest huvipakkuvaim on arvatavasti `PID`, mis sisaldab protsessi identifikaatorit. Väli `CMD` sisaldab käivitatud käsu nime. Väga tihti kutsutakse `ps` välja järgmisel moel:

```
$ ps ax | less
```

See näitab kõigi parajasti töötavate protsesside nimekirja, kus saate üles leida protsessid, mis paistavad probleeme tekitavat, ja need siis lõpetada.

10.2.2. pstree

Käsk `pstree` näitab protsesse puukujulisena. Selle üks eeliseid on see, et näete otsekohe protsesside eellasi: kui soovite surmata terve rea protsesse ja need on kõik mingi käsu järglased, võite lihtsalt surmata eellase. Kõigi protsesside PID nägemiseks tuleb anda võti `-p`, protsessi käivitanud kasutaja nägemiseks võti `-u`. Kuna puustruktuur on tavaliselt üsna pikk, oleks mõttekas `pstree` välja kutsuda järgmisel moel:

```
$ pstree -up | less
```

See annab ülevaate kogu protsessipuust.

10.3. Signaalide saatmine protsessidele: kill, killall ja top

10.3.1. kill, killall

Nende kahe käsuga saab protsessidele signaale saata. Käsk `kill` nõuab argumendina protsessi numbrit, `killall` aga selle nime.

Mõlemale käsule võib lisaks anda argumendina signaali numbri, mida soovite protsessile saata. Vaikimisi saadavad mõlemad protsessile signaali 15 (`TERM`). Kui näiteks soovite surmata protsessi, mille PID on 785, andke järgmine käsk:

```
$ kill 785
```

Kui soovite aga saata signaali 19 (`STOP`, andke selline käsk:

```
$ kill -19 785
```

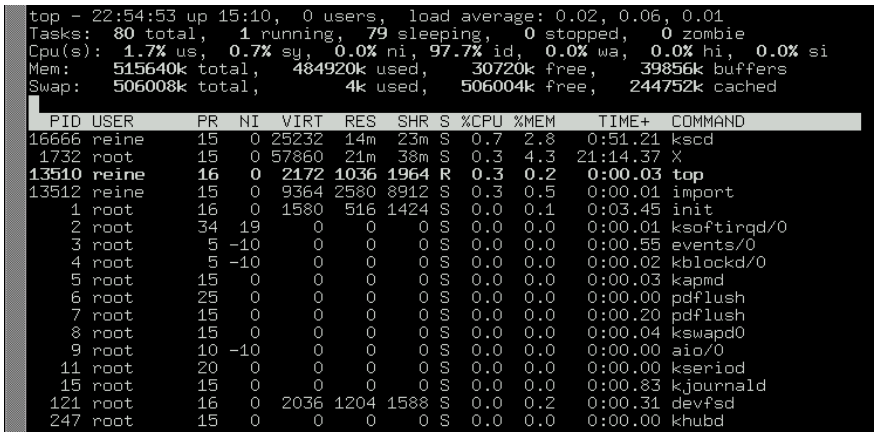
Oletame nüüd, et soovite surmata protsessi, mille kohta Te teate käsu nime. Te ei pea hakkama `ps`'i abil protsessi järjekorranumbrit otsima, vaid võite surmata protsessi nime järgi. Kui tegemist on näiteks protsessiga "mozilla", võite anda käsu:

```
$ killall -9 mozilla
```

Igal juhul saate surmata ainult iseenda protsesse, kui te ei ole just administraator (`root`), nii et Teil pole vaja mitmekasutajasüsteemis töötades tunda muret võimaliku sekkumise pärast teiste kasutajate asjadesse, sest Teie signaalid ei puuduta kuidagi nende protsesse.

10.3.2. ps ja kill üheskoos: top

`top` on programm, mis täidab korraga nii `ps`'i kui `kill`'i funktsioone ja millega saab ka jälgida protsesse reaalajas, kusjuures see annab Teile infot CPU ja mälu kasutamise, töötamisaja jms. kohta, nagu näitab Joonis 10-1.



Joonis 10-1. Protsesside jälgimine top'iga

Utiliiti `top` juhitakse täielikult klaviatuurilt. Abi saab vaadata klahvile **h** vajutades. Levinumad käsud on järgmised:

- **k**: sellega saadetakse protsessidele signaale. `top` tahab teada protsessi PID-d ning saadetava signaali numbrit või nime (vaikimisi `TERM` või `15`).
- **M**: see käsk sordib protsessid mälukasutuse järgi (väli `%MEM`).
- **P**: see sordib protsessid CPU aja järgi (väli `%CPU`), mis on ka vaikimisi sorteerimismeetod.
- **u**: sellega saab lasta näidata konkreetse kasutaja protsesse. `top` tahab ka teada, millist kasutajat soovite jälgida. Teil tuleb anda kasutaja **nimi**, mitte UID. Kui Te nime ei anna, näidatakse kõiki protsesse.
- **i**: vaikimisi näidatakse kõiki, ka magavaid protsesse. Selle käsuga saab lasta näidata ainult parajasti töötavaid protsesse ehk protsesse, mille väljal `STAT` seisab `R` (*Running* ehk 'töötab'). Sama käsu uuestiandmisel näidatakse taas kõiki protsesse.
- **r**: sellega saab muuta valitud protsessi prioriteeti.

10.4. Protsesside prioriteedi määramine: nice, renice

Iga protsess töötab süsteemis teatud kindla prioriteediga, mida nimetatakse ka "viisakuseks" ja mis jääb vahemikku -20 (suurim prioriteet) kuni 19 (väikseim prioriteet). Kui see ei ole kindlaks määratud, töötavad protsessid vaikeprioriteediga 0 ("baasprioriteet"). Suurema prioriteediga ehk väiksema viisakusega (kuni -20) protsessid ajastatakse töötama sagedamini kui väiksema prioriteedega (kuni 19) protsessid, mis tagab neile rohkem protsessori tööaega. Tavalised kasutajad saavad ainult omaenda protsesside prioriteeti alandada vahemikus 0 kuni 19. Administraator (`root`) võib määrata igale protsessile mis tahes väärtuse.

10.4.1. renice

Kui mõni protsess nõuab liiga palju süsteemi ressursse, võib surmamise asemel hoopis muuta protsessi prioriteeti. Selleks saab kasutada käsku `renice`, mille süntaks on järgmine:

```
renice prioriteet [[-p] pid ...] [[-g] pgrp ...] [[-u] kasutaja ...]
```

kus `prioriteet` on prioriteedi arvväärtus, `pid` (kasutage mitme protsessi korral võtit `-p`) protsessi ID, `pgrp` (kasutage mitme korral võtit `-g`) protsessi grupi ID ja `kasutaja` (kasutage mitme korral võtit `-u`) protsessi omaniku kasutajanimi.

Oletame, et Teil töötab protsessi, mille PID on 785 ja mis sooritab pikka ning keerulist teadusarvutust. Selle ajal eelistaksite mängida mõnda mängu, mis vajab aga süsteemi ressursside vabastamist. Selleks andke käsk:

```
$ renice +15 785
```

Sel juhul kulub protsessile arvatavasti rohkem aega, enne kui ülesanne saab täidetud, kuid see-eest ei haara see teiste protsesside eest endale CPU aega.

Kui olete süsteemiadministraator ja näete, et mõnel kasutajal töötab liiga palju protsesse, mis kasutavad liiga palju süsteemi ressursse, võite muuta antud kasutaja protsesside prioriteet üheainsa käsuga:

```
# renice +20 -u peter
```

Pärast seda on kõigil peter protsessidel väikseim prioriteet ning need ei sega enam kuidagi teiste kasutajate käivitatud protsesse.

10.4.2. nice

Nüüd, kus olete saanud teada, kuidas muuta protsesside prioriteeti, võib Teil tekkida huvi, kuidas käivitada käsud juba etteantud prioriteediga. Selleks on mõeldud käsk `nice`.

Selle kasutamisel tuleb tegelikult käivitatav käsk anda `nice`'i võtmena. Võtmega `-n` saab määrata prioriteedi. Vaikimisi määrab `nice` prioriteediks 10.

Oletame, et soovite luua Mandriva Linuxi paigaldus-CD-ROM-i ISO-tõmmist:

```
$ dd if=/dev/cdrom of=~/mandriva1.iso
```

Mõnes süsteemis, kus on standardne IDE CD-ROM, võib suure infohulga kopeerimine võtta enda alla väga palju süsteemi ressursse. Kui Te ei soovi, et kopeerimine blokeeriks muud protsessid, võite käivitada protsessi väiksema prioriteediga näiteks sellise käsuga:

```
$ nice -n 19 dd if=/dev/cdrom of=~/mandriva1.iso
```

Peatükk 11. Käivitusfailid: init sysv

System V käivitusskeem on pärit juba AT&T UNIX® aegadest ning see on üks traditsioonilisi UNIX® viise süsteem käivitada. See vastutab teenuste käivitamise või peatamise eest, et sel moel viia süsteem ühte teatud vaikimisi süsteemiolekust. Teenused ulatuvad alates kasutaja autentimisest kuni kohaliku graafikaserveri ja Internetiteenusteni.

11.1. Alguses oli init

Kui süteem käivitub ning kernel on kõik ära konfigureerinud ja juurfailisüsteemi haakinud, käivitab ta käsu `/sbin/init`[1]. `init` on kõigi süsteemi protsesside algpunkt ning just temal lasub vastutus viia süsteem vajalikule ***käivitustasemele***. Käivitustasemeid vaatleme veidi hiljem (Sektsioon 11.2).

`init`'i konfiguratsioonifailiks on `/etc/inittab` ja sel on lausa omaette manuaalilehekülg (inittab(5)), mistõttu me tutvustame siin ainult kõige olulisemaid konfiguratsioonivääртusi.

Esimene rida, millele tuleks tähelepanu osutada, on selline:

```
si::sysinit:/etc/rc.d/rc.sysinit
```

See rida annab `init`'ile teada, et `/etc/rc.d/rc.sysinit` tuleb käivitada pärast süsteemi initsialiseerimist (`si` tähendabki *System Init*). Vaikimisi käivitustaseme määramiseks otsib `init` seejärel rida, kus seisab võtmesõna `initdefault`:

```
id:5:initdefault
```

Antud juhul saab `init` teada, et vaikimisi käivitustase on 5. See tähendab ka seda, et tasemele 5 suundumiseks tuleb tal anda korraldus:

```
l5:5:wait:/etc/rc.d/rc 5
```

Nagu edaspidi näete, on teiste käivitustasemete süntaks samasugune.

`init` vastutab ka teatud programmide taaskäivitamise (`respawn`) eest, mida ei saa käivitada ükski muu protsess. Näiteks käivitab `init` kõiki sisselogimisprogramme, mis töötavad Teie käsutuses olevas kuues virtuaalses konsoolis[2]. Teist virtuaalset konsooli tuvastatakse näiteks nii:

```
2:2345:respawn:/sbin/mingetty tty2
```

11.2. Käivitustasemed

Kõik süsteemi käivitamisega seotdu failid asuvad kataloogis `/etc/rc.d`. Need failid on järgmised:

```
$ ls /etc/rc.d
init.d/  rc0.d/  rc2.d/  rc4.d/  rc6.d/            rc.local*    rc.sysinit*
rc*      rc1.d/  rc3.d/  rc5.d/  rc.alsa_default*  rc.modules*
```

Nagu mainitud, on `rc.sysinit` esimene fail, mille süsteem käivitab. See kannab vastutust masina põhilise seadistuse määramise eest: klaviatuuritüüp, teatud seadmete seadistamine, failisüsteemi kontroll jne.

Seejärel käivitatakse skript `rc`, mille argumendiks on soovitud käivitustase. Nagu nägime, on käivitustase lihtsalt üks täisarv. Iga käivitustaseme `<x>` jaoks peab olema vastav kataloog `rc<x>.d`. Tüüpilise Mandriva Linuxi paigalduse korral on käivitustasemeid kuus:

- 0: masina täielik peatamine.
- 1: ***ainukasutaja***režiim. Väga tõsiste probleemide või süsteemi taastamise jaoks.
- 2: ***mitmekasutaja***režiim ilma võrguta.

1. Seepärast ei ole `/sbin` paigutamine mõnda teise failisüsteemi kui juurfailisüsteem kohe üldse hea mõte. Kernel ei ole sel hetkel haakinud veel ühtki muud partitsiooni ega suuda seepärast leida käsku `/sbin/init`.
2. Kui Te ei soovi just täpselt kuut virtuaalset konsooli, võite neid lisada või eemaldada seda faili muutes. Kui soovite konsoolide arvu suurendada, siis teadke, et neid võib olla kuni 64. Kuid ärge unustage, et ka X töötab virtuaalses konsoolis, nii et X'i jaoks tuleks vähemalt üks konsool vabaks jätta.

- 3: mitmekasutajarežiim võrguga.
- 4: pole kasutusel.
- 5: nagu käivitustase 3, aga pannakse tööle graafiline sisselogimisliides.
- 6: taaskäivitus.

Vaatame nüüd kataloogi `rc3.d` sisu:

```
$ ls /etc/rc.d/rc3.d/
K09dm@          S12syslog@    S24messagebus@      S40atd@          S91dictd-server@
S01udev@        S13partmon@   S25haldaemon@       S55sshd@         S92lisa@
S03iptables@    S15cups@      S25netfs@           S56ntpd@         S95kheader@
S05harddrake@   S17alsa@      S29numlock@         S56rawdevices@   S99local@
S10network@     S18sound@     S33nifd@            S75keytable@
S11shorewall@   S20xfs@       S34mDNSResponder@   S90crond@
$
```

Nagu näete, on kõik selle kataloogi failid nimeviidad, kusjuures neil on väga spetsiifiline vorm. Üldiselt on see selline:

```
<S|K><järjekord><teenuse_nimi>
```

`S` tähendab teenuse käivitamist (*Start*), `K` selle surmamist (*Kill*) ehk peatamist. Skriptid käivitatakse alanevas numbrite järjekorras, kui kahel skriptil peaks olema sama number, arvestatakse alanevalt tähestikulist järjekorda. Samuti võime näha, et kõik nimeviidad osutavad vastavale skriptile kataloogis `/etc/init.d` (välja arvatud skript `local`, mis vastutab spetsiifilise teenuse juhtimise eest).

Kui süsteem siseneb teatud käivitustasemele, pannakse etteantud järjekorras tööle `K`-lingid: käsk `rc` tuvastab asukoha, millele link viitab, ning käivitab seejärel vastava skripti argumendiga `stop`. Seejärel pannakse samamoodi tööle `S`-skriptid, ainult et nende korral on argumendiks `start`.

Süvenemata skriptide olemusse võime öelda, et kui süsteem suundub käivitustasemele 3, pannakse kõigepealt tööle käsk `K09dm` (s.t. `/etc/init.d/dm stop`). Siis võetakse ette `S`-skriptid: kõigepealt `S01udev`, mis kutsub välja `/etc/init.d/udev start`, seejärel `S03iptables` ja nii edasi.

Eelöeldu põhjal võite mõne minutiga luua täiesti omaenda käivitustaseme (kasutades selleks näiteks käivitustaset 4) või vältida mõne teenuse käivitamist või peatamist, kustutades vastava nimeviida.

11.2.1. Käivitustaseme teenuste seadistamine

Teenuste lisamiseks, eemaldamiseks, aktiveerimiseks või deaktiveerimiseks mingil käivitustasemel saab kasutada ka käsku `chkconfig`. Käsuga `chkconfig --add teenuse_nimi` saab teenuse `teenuse_nimi` lisada (aktiveerida) kõigile toetatud[3]käivutustasemetele ning käsuga `chkconfig --del teenuse_nimi` määratud teenuse kõigilt käivitustasemetelt eemaldada (deaktiveerida).

Käsuga `chkconfig --list` saate teada, millised teenused on saadaval, millised on nende nimed ja milline on nende olek kõigil defineeritud käivitustasemetel.

Käsuga `chkconfig --levels 35 sshd on` aktiveeritakse SSH-server (`sshd`) käivitustasemetel 3 ja 5, käsuga `chkconfig --levels 3 sound off` aga eemaldatakse käivitustasemelt 3 heli toetus. Kui jätta ära parameeter `--levels tasemete_nimekiri`, aktiveeritakse või deaktiveeritakse määratud teenus käivitustasemetel 2, 3, 4, ja 5. Pange tähele, et see võib kaasa tuua teenuste lubamise sellistel käivitustasemetel, mis ei suuda neid korralikult toetada. Seepärast on mõttekam määrata konkreetne käivitustase.

3. "Toetatud" käivitustase tähendab näiteks seda, et võrguga seotud teenuseid ei saa lisada käivutustasemele 2, sest see ei toeta võrguühendust.

11.2.2. Töötava süsteemi teenuste juhtimine

Töötavas süsteemis saab teenuseid juhtida käsuga `service` sõltumata sellest, kas teenused on seadistatud antud käivitustasemel töötama või mitte. Selle käsu süntaks on järgmine:

```
service teenuse_nimi toiming
```

kus `teenuse_nimi` on vajaliku teenuse nimi kujul, nagu selle annab `chkconfig --list`, ning `toiming` üks järgmistest:

start

: Käivitab määratud teenuse. Pange tähele, et enamik teenuseid hoiatab, kui nad on juba käivitatud. Sellisel juhul tuleks kasutada toimingut `restart`.

stop

: Peatab määratud teenuse. Pange tähele, et teenuse peatamisel lahutatakse automaatselt kõik kasutajad, kes olid antud teenusega ühendatud.

restart

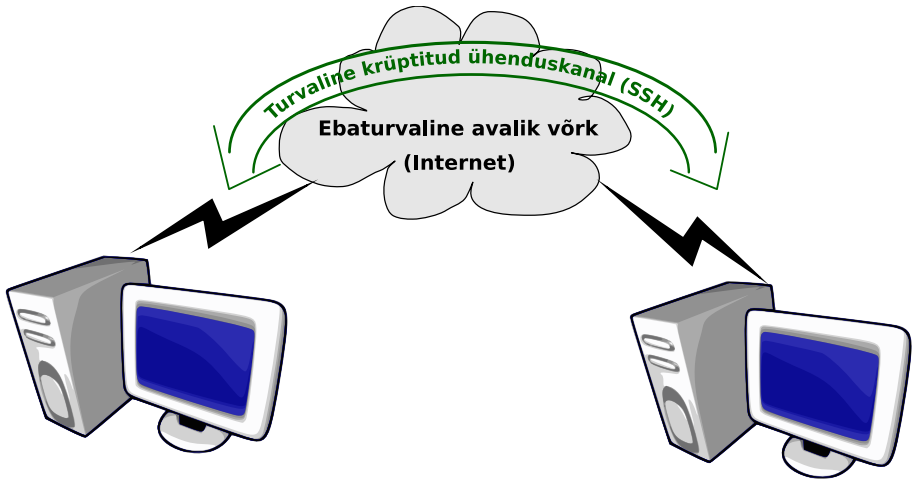
: Peatab ja seejärel käivitab määratud teenuse. Sama tulemuse annab `service teenuse_nimi stop && service teenuse_nimi start`. Pange tähele, et teenuse taaskäivitamisel lahutatakse automaatselt kõik kasutajad, kes olid antud teenusega ühendatud.

muud, teenusest sõltuvad toimingud

: Erinevad teenused toetavad erinevaid toiminguid (eelnevalt mainitud on ühised kõigile teenustele). Näiteks `reload` laadib uuesti teenuse konfiguratsioonifaili ilma teenust taaskäivitamata, `force-stop` sunnib peale teenuse seiskamise, `status` näitab infot teenuse oleku kohta jne. Käsku `service teenuse_nimi` andes näete kõiki toiminguid, mida antud teenus toetab.

Peatükk 12. Turvaline kaugligipääs

Süsteemiadministraatoritel võib mõnikord olla vaja saada ühendus füüsiliselt kuskil mujal asuvate masinatega, et redigeerida konfiguratsioonifaile, juhtida teenuseid, käivitada programme jne. Kaugligipääsuks kasutati varem sageli programmi `telnet`, kuid see on üsna ebaturvaline. Et kogu suhtlus toimub avaliku (ja sestap loomu poolest ebaturvalise) Interneti vahendusel, vajavad süsteemiadministraatorid turvalise kaugligipääsu võimalust. `ssh` (mis tähendabki turvalist shelli, inglise keeles **S**ecure **SH**ell) võimaldab ligipääsu kaugmasinatele väga turvaliselt, krüptides kogu suhtluse.



Joonis 12-1. SSH ühendusskeem

12.1. SSH-serveri seadistamine

"Server" tähendab siinkohal masinat, millega Te ühenduse loote. Kontrollige lihtsalt, et pakett `openssh-server` on paigaldatud ning `sshd` teenus töötab[1].

SSH-serveri elementaarne seadistus võimaldab kasutajatel pääseda (ehk "ssh-ida") masinasse, kui neil on seal konto. Kui soovite piirata SSH-ligipääsu konkreetsete kasutajatega, redigeerige faili `/etc/ssh/sshd_config` ning lisage sinna umbes selline rida (või muutke seda):

```
AllowUsers queen peter@192.168.0.*
```

Toodud näite korral on ainult kasutajatel queen ja peter õigus ühenduda SSH vahendusel masinaga, sealjuures kasutajal peter on ligipääsuõigus ainult masinast, mis asub `192.168.0.` (koht-)võrgus.

Kasutajad loovad ühenduse oma tavalise kontoga ning saavad seejärel käsu `su` abil võtta endale administraatori (`root`) õigused. Kui soovite, et kasutajad saaks SSH vahendusel sisse logida otse kasutaja `root` õigustes, andke reale `PermitRootLogin no` kuju `PermitRootLogin yes`. Palun arvestage, et kuigi see võib olla mugav, vähendab see põhimõtteliselt turvalisust.

SSH-serveri võtmetest ja seadistamisest räägivad lähemalt manuaalileheküljed sshd(8) ja sshd_config(5).

12.2. SSH kliendi seadistamine

"Klient" tähendab siinkohal masinat, millest ühendus luuakse. Kontrollige lihtsalt, et pakett `openssh-clients` on paigaldatud.

Andke käsk `ssh kasutajanimi@kaugmasin`, kui soovite luua ühenduse masinaga `kaugmasin`, kasutades kontot `kasutajanimi`. Teie käest päritakse seejärel parooli. Andke parool ning Teile tagataksegi samasugune ligipääs, nagu asuksite ka füüsiliselt kaugmasina konsooli taga.

Sõltumata sellest, kas loote ühenduse ühe või mitme masinaga (viimane on süsteemiadministraatori puhul isegi usutavam), saate parooli küsimise etapi vahele jätta SSH võtmeid kasutades. Käsuga `ssh-keygen` saab genereerida oma SSH võtme ning seejärel käsuga `ssh-copy-id kasutajanimi@kaugmasin` kopeerida

1. Selle käivitamiseks andke käsk `service sshd start`. SSH teenus seadistatakse käivituma juba algkäivituse ajal.

selle kaugmasinatele. Kui seejärel annate käsu `ssh-copy-id`, küsitakse Teie käest küll parooli, aga ainult üks kord kogu süsteemi kohta. Nüüd võite rahulikult luua SSH ühenduse kaugmasinatega, ilma Teie käest enam parooli päritaks.

Selle mehhanismi toimimiseks tuleb anda käsks `ssh-add` ning sisestada oma paroolifraas — selle lõite oma SSH võtit genereerides — iga kord, kui alustate seanssi klientmasinas.

Kui saate teate, et ühendust Teie autentimisprogrammiga ei õnnestunud luua, andke enne käsku `ssh-add` käsk `eval 'ssh-agent'` (pange tähele jutumärke).

12.3. Failide kopeerimine võrgusüsteemist või võrgusüsteemi

Failide liigutamiseks kaugmasinasse, kus töötab SSH-server, tarvitage käsku `scp` (see tähendab turvalist kopeerimist, inglise keeles **S**ecure **C**o**P**y). Selle süntaks on järgmine:

```
scp [võtmed] kohalik_asukoht [kasutaja@]kaugmasin:[täielik_asukoht_kaugmasinas]
```

Kui Te ei määra komponenti `kasutaja@`, kasutatakse Teie kasutajanime klientmasinas. Kui jätate ära asukoha kaugmasinas, kopeeritakse fail kaugmasinas `kasutaja` kodukataloogi. Pange tähele, et koolon (`:`) eristab kasutajanime ja masina määratluse asukohast kaugmasinas.

Failide liigutamiseks kaugmasinast kohalikku masinasse on süntaks järgmine:

```
scp [võtmed] [kasutaja@]kaugmasin:täielik_asukoht_kaugmasinas kohalik_asukoht
```

Kui lähtekohaks määrata kataloog, on kohustuslik kasutada võtit `-r` (rekursiivne). Täpsemalt räägib `scp` võtmetest scp(1).

Peatükk 13. Tarkvarahaldus käsureal

Rpmdrake oma erinevate liidestega kujutab endast kõigest võimsa käsureatööriistade komplekti urpmi kasutajaliidest. Nende huvides, kes soovivad oma tarkvara hallata käsureal (see on kasulik näiteks kaughaldamise korral), toome siinkohal ära mõningad kasulikumad käsud.

13.1. Tarkvara paigaldamine ja eemaldamine

Seda saab teha kahe lihtsa käsuga:

`urpmi <paketi_nimi>`

See käsk paigaldab paketi `paketi_nimi`, kui see on olemas, või paketi, mille nimes leidub string `paketi_nimi`. Kui sobima peaks enam kui üks pakett, näidatakse Teile sobivuste nimekirja. Sel juhul tuleb Teil lihtsalt kirjutada vajaliku valiku number ning vajutada klahvi **Enter**.

Kui paketil, mida soovite paigaldada, on sõltuvusi (teisi tarkvarapakette, mida on vaja Teie soovitud paketi korralikuks töötamiseks), näidatakse Teile nende nimekirja. Vaadake see üle ja vajutage kõigi pakettide paigaldamiseks klahvi **J**.

`urpme <paketi_nimi>`

See käsk eemaldab tarkvarapaketi `paketi_nimi`. Kui sellest sõltuvad mingid muud paigaldatud tarkvarapaketid, näidatakse nende nimekirja koos põhjusega, miks antud pakett tuleb samuti eemaldada. Vaadake nimekiri üle ja vajutage kõigi pakettide eemaldamiseks klahvi **J**.

Nii `urpmi` kui `urpme` toetavad võtme `--auto` kasutamist, mille korral tarkvara ning võimalikud sõltuvused paigaldatakse või eemaldatakse automaatselt.

Täpsemalt räägivad nende käskude võtmetest urpmi(8) ja urpme(8) manuaalilehekülјed.

13.2. Tarkvaraallikate haldamine

Tarkvaraallikad kujutavad endast mitmesuguseid "andmekandjaid", kust saab tarkvara paigaldada. Et käsku `urpmi` üldse kasutada, peab olema määratud vähemalt üks tarkvaraallikas. Eelnevalt määratud allikateks on näiteks andmekandjad, kust Te oma süsteemi paigaldasite (võrk, CD, DVD jne.). Te võite aga allikaid ka ise lisada, eriti just selliseid, kust hankida veaparandusi ja turvauuendusi. Tarkvaraallika lisamine ja eemaldamine käsureal on lihtne, kuid kindlasti tuleb väga täpselt jälgida käsu süntaksit.

13.2.1. Uute allikate lisamine

`urpmi.addmedia <nimi> <URL>`

See käsk võimaldab lisada uue andmekandja kas kohalikult kõvakettalt, eemaldatavalt seadmelt (CD/DVD) või võrgust (protokollid HTTP, FTP, NFS, `ssh` või `rsync`). URL erinevate allikate lisamise süntaksis on mõnevõrra varieeruv, mistõttu oleks eelnevalt kasulik teha tutvust urpmi.addmedia(8) manuaalilehekülјega.

Kui soovite lisada uue uuendusteallika, kasutage käsu `urpmi.addmedia` juures võtit `--update`.

Kas Te ei tea, kus leida uusi tarkvaraallikaid, mis pakuks asjalikku ning just Teie Mandriva Linuxi süsteemi tarbeks pakendatud tarkvara? Neid aitab leida Easy Urpmi veebilehekülg (`http://easyurpmi.zarb.org/`). Ka Mandriva Club (`http://club.mandriva.com/`) pakub urpmi andmekandjaid (`http://club.mandriva.com/modules.php?name=Mirrors-list`) test- ja kontributsioonipakettidega.

Mandriva Clubi allikad on saadaval ainult klubi liikmetele.

13.2.2. Allikate eemaldamine

```
urpmi.removemedia <nimi>
```

Selle käsuga eemaldatakse andmekandja `nimi`. Kui Teil ei ole oma tarkvaraallikate nimed meeles, andke lihtsalt käsk `urpmi.removemedia` ja näete kõiki olemasolevaid tarkvaraallikaid.

13.2.3. Allikate uuendamine

```
urpmi.update <nimi>
```

See käsk uurib läbi vastava andmekandja ja uuendab selle pakettide nimekirja. Ennekõike on see kasuks uuendusteandmekandjate puhul, mille sisu võib üsna sageli muutuda. Kui soovite uuendada kõiki oma tarkvaraallikaid korraga, kasutage võtit `-a`.

13.2.4. Allikate järjekord

Järjekord, milles tarkvaraallikad on määratud failis `/etc/urpmi/urpmi.cfg`, on päris oluline, sest see määrab, milliselt andmekandjalt pakett paigaldatakse, kui antud paketti peaks pakkuma mitu allikat: paketid paigaldatakse järjekorras esimesest allikast.

Võrguandmekandja lisamisel tuleks see lisada eemaldatava ja kohaliku allika ette. Selle põhjuseks on asjaolu, et võrguandmekandjatel leidub enamasti rohkem tarkvarapakette kui eemaldatavatel või kohalikel andmekandjatel.

13.3. Nipid ja trikid

13.3.1. Süntees või täielik nimekiri?

Andmekandja lisamisel on pakettide nimekiri võimalik esitada kahel kujul: süntees või täielik. Pakettide sünteesnimekirja leidmiseks ja kasutamiseks pruukige võtit `--probe-synthesis`, täieliku nimekirja leidmiseks ja kasutamiseks võtit `--probe-hdlist`. Sünteesnimekirjad on väiksemad, mistõttu sobivad paremini aeglasema võrguühenduse korral, kuid samas sisaldavad nad ka märksa vähem infot pakettide kohta.

13.3.2. Konkreetset faili sisaldava tarkvarapaketi leidmine

Te teate, et Teie süsteemis on üks kindel fail, aga Te ei tea, mis tarkvarapaketti see kuulub... Andke lihtsalt käsk `urpmf <failinimi>` ning näetegi kõiki, nii paigaldatud kui paigaldamata pakette, mis sisaldavad faili `failinimi`.

Sünteesnimekirja kasutamise korral otsib `urpmf` faili ainult juba paigaldatud pakettides.

You can even provide only a partial name. For example `urpmf salsa` will return a list of all packages which contain a file whose name contains the word `salsa`.

```
[root@test queen]# urpmf salsa
```

```
kaffe:/usr/lib/kaffe/lib/i386/libtritonusalsa-1.1.2.so
kaffe:/usr/lib/kaffe/lib/i386/libtritonusalsa.la
kaffe:/usr/lib/kaffe/lib/i386/libtritonusalsa.so
```

13.3.3. Pakettide uuendamine

Järgmise käsuga saab uuendada konkreetset tarkvarapaketti:

```
urpmi.update -a && urpmi --update <paketi_nimi>
```

Järgmise käsuga saab automaatselt uuendada kõik tarkvarapaketid täpselt nii, nagu teeb seda Mandriva Update:

```
urpmi.update -a && urpmi --update --auto-select --auto
```

Kui Teil ei ole spetsiaalselt uuendusandmekandjaks määratud tarkvaraallikat, tuleb eeltoodud urpmi käskude korral võti `--update` ära jätta.

Lisa A. Sõnastik

konto
UNIX® süsteemides nime, isikliku kataloogi, parooli ja shelli kombinatsioon, mida pruukides kasutajal on võimalik süsteemiga ühineda.

alias
Mehhanism, mida shell kasutab mingi stringi asendamiseks teisega enne käsu täitmist. Kõiki antud seansis defineeritud aliaseid näeb käsuga `alias`.

ACPI
Advanced Configuration and Power Interface ehk täiustatud konfiguratsiooni- ja toiteliides. Mõeldud riistvara tuvastamiseks ja seadistamiseks ning voolutarbe juhtimiseks. Erinevalt APM-ist, mis tugineb ainult BIOS-ele, tugineb ACPI ka operatsioonisüsteemile, mistõttu on kasutaja jaoks lihtsam. ACPI pakub ka serveritele ja tööjaamadele voolutarbe haldamise võimalusi.

APM
Advanced Power Management eh täiustatud toitehaldus. Mõne BIOS-e korral saab seda kasutada masina saatmiseks ooteseisundisse (standby) pärast teatud aega mitteaktiivsust. Sülearvutitel annab APM ka teada aku olekust ja (kui see on toetatud) aku oletatavast järelejäänud tööajast. Uuemates sülearvutites on siiski kasutusel pigem juba ACPI kui APM.
Vt. ka: ACPI.

ARP
Address Resolution Protocol ehk aadressiteisenduse protokoll. Internetiprotokoll, mida kasutatakse Internetiaadressi dünaamiliseks sidumiseks füüsilise (riistvaralise) aadressiga kohtvõrgus. Selle kasutamine on piiratud võrkudega, mis toetavad riistvara-aadresside levitamist.

ASCII
American Standard Code for Information Interchange ehk Ameerika infovahetuse standardkood. Seda kasutatakse sümbolite, sealhulgas juhtsümbolite salvestamiseks arvutis. Paljud 8-bitised koodid (näiteks ISO-8859-1, mis on Linuxis vaikimisi kooditabel, kui Te ei ole just valinud midagi muud, näiteks UTF-8) sisaldavad alumises pooles ASCII-d.
Vt. ka: ISO-8859, UTF-8.

assemblerkeel
Programmeerimiskeel, mis on kõige lähemal masinakeelele ehk niinimetatud "madalkeelest". Assemblerkeele eeliseks on kiirus, sest programmid pannakse kirja protsessori instruktsioonidena, mistõttu käivitatavate failide genereerimisel ei ole transleerimist peaaegu vajagi. Suuremaks miinuseks on asjaolu, et see sõltub protsessorist (või arhitektuurist). Ka on keerukamate programmide kirjutamine üsna aeganõudev. Niisiis on assemblerkeel küll kiireim programmeerimiskeel, kuid see tuleb iga arhitektuuri jaoks eraldi kirjutada.

ATAPI
AT Attachment Packet Interface ehk AT-siini manuse pakettliides. ATA (*Advanced Technology Attachment* ehk AT-siini manuse, mida rohkem tuntakse nimetusega IDE ehk *Integrated Drive Electronics* ehk integreeritud ajamielektroonika) spetsifikatsiooni laiendus, mis pakub lisakäske CD- ja lindiseadmete juhtimiseks. Selle laiendusega varustatud IDE kontrollereid nimetatakse ka EIDE (*Enhanced IDE* ehk täiustatud IDE) kontrolleriteks.
Vt. ka: IDE.

ATM
Asynchronous Transfer Mode ehk asünkroon-andmeedastus. ATM võrgutehnoloogia pakib andmed standardse suurusega plokkidesse (53 baiti: 48 andmetele ja 5 päisele), mida on mugav ühest punktist teise toimetada. ATM on ennekõike väga kiiretele (multimegabitistele) võrkudele mõeldud andmete edastamise tehnoloogia.

atomaarne
Operatsioonide kogumit nimetatakse atomaarseks, kui need täidetakse korraga ja neid ei saa enne lõpetamist katkestada. See on omamoodi " kõik või ei midagi": kõik operatsioonid täidetakse edukalt või ei arvestata ühegi operatsiooni tulemust. Sellised võivad olla ka väga tähtsad või väga lihtsad operatsoonid (näiteks kahe täisarvu liitmine).

taust
See tähendab shelli kontekstis protsessi, mis töötab taustal ega sega muid protsesse. Selle vastandiks on esiplaaniprotsess.
Vt. ka: töö, esiplaan.

varundamine
Oluliste andmete salvestamine turvalisele andmekandjale või asukohta. Varukoopiaid tuleks teha regulaarselt, eriti kui kõne all on eluliselt vajalik info ja konfiguratsioonifailid (tähtsaimad kataloogid, mida varundada, on `/etc`, `/home` ja `/usr/local`). Tavaliselt kasutavad paljud kataloogide ja failide varundamiseks programmi `tar`, tihendades failid programmiga `gzip` või `bzip2`. Lisaks võib muidugi kasutada programme `dump` ja `restore` või ka paljusid muid vabasid või kommertsvarundamisprogramme.

pakktöötlus
Töötlemisrežiim, mille korral protsessorile edastatavad tööd või instruktsioonid täidetakse üksteise järel, kuni kõik on töödeldud.

piiks
Heli, millega Teie arvuti kõlar annab teada näiteks mitme võimalusega olukorrast käsu lõpetamisel. Paljud programmid annavad piiksuga teada mingitest tähelepanu vajavatest situatsioonidest.

beetatestimine
Tähendab programmi testimise teatud järku. Programmid läbivad enne lõpliku väljalaste valmimist tavaliselt "alfa", "beeta" ja "release candidate'i" staadiumi.

binaarfail
Programmeerimises tähendavad binaarfailid kompileeritud käivitatavat koodi.

bitt
BInary digiT ehk kahendarv. Arv, mis võib olla kas 0 või 1, sest aluseks on 2 (kahendsüsteem). See on digitaalse info kõige elementaarsem ühik.

plokkseade
Failid, mille sisu puhverdatakse. Kõik selliste failide lugemis-kirjutamisoperatsioonid käivad läbi puhvri, mis võimaldab riistvaral asünkroonset lugemist ja kirjutamist, see aga omakorda väldib seda, et süsteem pöörduks ketta poole, kui andmed on juba puhvris.
Vt. ka: puhver, puhvermälu, sümbolseade.

alglaadimine
Protseduur, mis toimub pärast arvuti sisselülitamist ja mille käigus tuvastatakse üksteise järel välisseadmed ning laaditakse mällu operatsioonisüsteem.

alglaadimisketas
Alglaadimisketas (diskett, CD, DVD või mõni muu seade) sisaldab koodi, mida on vaja operatsioonisüsteemi laadimiseks kõvakettalt (mõnikord piisab ka seadmest endast).

alglaadur
Programm, mis käivitab operatsioonisüsteemi. Paljud alglaadurid annavad Teile võimaluse valida menüüst, millist operatsioonisüsteemi soovite laadida. Just sellepärast on populaarsed sellised alglaadurid nagu GRUB ja LiLo, mille korral on väga lihtne üleval pidada kahe või enama operatsioonisüsteemiga süsteeme.

BSD
Berkeley Software Distribution ehk Berkeley tarkvaradistributsioon on UNIX® variant, mille töötas välja Berkeley ülikooli arvutiteaduse teaduskond. Seda on alati peetud teistest versioonidest tehniliselt täielikumaks ning see on andnud palju uuendusi arvutimaailmale üldiselt, eriti aga UNIX® maailmale.

puhver
Fikseeritud suurusega väike mäluosa, mille saab seostada plokkseadme, süsteemitabeli, protsessi või muu sellisega. Puhvermälu tagab kõigi puhvrite korrektsuse.
Vt. ka: puhvermälu.

puhvermälu
Operatsioonisüsteemi kerneli eluliselt oluline osa, mis vastutab kõigi puhvrite aktuaalsuse eest, kahandab vajaduse korral puhvermälu suurust, puhastab ebavajalikud puhvrid ja teeb veel palju muud.
Vt. ka: puhver.

viga
Programmi ebaloogiline või ebaühtlane käitumine teatud juhtudel või käitumine, mis ei järgi dokumentatsiooni või programmile ettenähtud standardeid. Sageli võivad vigu programmidesse kaasa tuua uued omadused ja võimalused, eriti kui need ei ole põhjalikult testitud. Ajalooliselt tuleneb inglise keeles kasutatav sõna 'bug' ('putukas', vahel eesti keeles ka toorlaenuna 'puuk') sellest, et päris ehtsad putukad ronisid kunagiste arvutite juures kasutatud perfokaartide aukudesse, mis tõi kaasa vigu programmide töös. Väidetavalt olevat admiral Grace Hopper selle peale kuulutanud: "It's a bug!" Tasapisi levis see laiemalt, kuid arvestage, et siinöeldu on ainult üks seletusi viga tähistava termini *bug* tekkele.

bait
Kaheksa järjestikuse biti jada, mis kümnendsüsteemi teisendatuna annab tulemuseks täisarvu vahemikus 0 kuni 255. Bait on süsteemis alati "atomaarne", s.t. see on väikseim adresseeritav üksus.
Vt. ka: bitt.

tõst
Sõnede korral tähendab tõst erinevust väike- ja suurtähtede vahel.

CHAP
Challenge-Handshake Authentication Protocol ehk väljakutse ja kätlusega autentimisprotokoll. Seda kasutavad ISP-d oma klientide autentimiseks. Selle skeemi kohaselt saadetakse kliendile (ühendust loovale masinale) väärtus, mida see kasutab väärtuse põhjal räsi (hash) arvutamiseks. Klient saadab räsi tagasi serverile, kus seda võrreldakse serveri enda arvutatud räsiga. See meetod erineb PAP-ist selle poolest, et pärast esialgset autentimist sooritatakse autentimine perioodiliselt uuesti.
Vt. ka: PAP.

sümbolseade
Failid, mille sisu ei puhverdata. Füüsiliste seadmetega seostamise korral toimib kogu seadme sisend ja väljund otsekohe ja vahetult. Mõningad spetsiaalsed sümbolseadmed loob ka operatsioonisüsteem (`/dev/zero`, `/dev/null` jms.). Need vastavad andmevoogudele.
Vt. ka: plokkseade.

CIFS
Common Internet File System ehk üldine interneti-failisüstem. SMB failisüsteemi järglane, kasutatakse DOS-süsteemides.
Vt. ka: SMB.

klient
Programm või arvuti, mis aeg-ajalt võtab teatud ajaks ühendust teise programmi või arvutiga, et anda sellele ülesandeid või hankida infot. Partnersüsteemide korral (**peer-to-peer**), näiteks SLIP või PPP, peetakse kliendiks seda poolt, mis algatab ühenduse, serveriks aga seda, kellele väljakutse saadetakse. See on üks **klient-serversüsteemi** osa.
Vt. ka: server.

klient-serversüsteem
Süsteem või protokoll, mis koosneb **serverist** ja vähemalt ühest **kliendist**.

käsurida
Seda pakub shell ning see võimaldab kasutajal käske vahetult anda. See on ühtlasi lakkamatute "sõimusõdade" allikas selle pooldjate ja põlastajate vahel.

käsure iim
Vi ja selle kloonide korral programmi olek, mille korral klahvi vajutamine ei sisesta redigeeritavasse faili sümbolit, vaid hoopis sooritab teatud toimingu (vähemalt vaikimisi, kui Te ei ole just seadistusi muutnud). Sellest saab väljuda mõnda "lisamisrežiimi naasvat" käsku kasutades: **i**, **I**, **a**, **A**, **s**, **S**, **o**, **O**, **c**, **C**...

kompileerimine
See on lähtekoodi, mis on inimesele arusaadav (vähemalt teatud ettevalmistusega inimestele) ning kirja pandud mõnes programmeerimiskeeles (näiteks C), transleerimine masinale arusaadavaks binaarfailiks.

lõpetamine
Nii nimetatakse shelli oskust asendada osaline sõne täieliku failinime, kasutajanime või mõne muu elemendiga, kui vastav element on olemas.

tihendamine
Viis vähendada failide suurust või edastavate märkide arvu. Failide tihendamise programmideks on näiteks compress, `zip`, `gzip` ja `bzip2`.

konsool
Nii nimetatakse tänapäeval terminale. Omal ajal olid viimased masinad (kuvar pluss klaviatuur), mis olid ühendatud ühe suure keskarvutiga. PC puhul on füüsiliseks terminaliks klaviatuur ja ekraan.
Vt. ka: virtuaalne konsool.

küpsised
Ajutised failid, mille kirjutab kohalikule kõvakettale võrgu-veebiserver. Need annavad serverile teada kasutaja eelistused, kui sama kasutaja serveriga uuesti ühendust võtab.

datagramm
Datagramm on diskreetne pakett andmete ja päistega, mis sisaldavad aadresse. See on IP võrgu iseseisev edastusühik. Vahel nimetatakse seda ka lihtsalt "paketiks".

sõltuvused
Kompileerimise sammud, mis tuleb täita enne järgmisi samme programmi edukaks kompileerimiseks. Sama mõistet kasutatakse ka juhul, kui mõni programm, mida soovite paigaldada, sõltub teistest programmidest, mis ei pruugi olla Teie süsteemi paigaldatud. Sel juhul näete teadet, mis ütleb, et süsteem peab paigaldamiseks "rahuldama sõltuvused".

töölaud
Kui kasutate X Window Systemit, on töölaud see koht ekraanil, kus Te töötate ja kus näidatakse aknaid ning ikoone. Seda nimetatakse ka taustaks ning tavaliselt täidab seda mingi värv, värviüleminek või isegi pilt.
Vt. ka: virtuaalsed töölauad.

DHCP
Dynamic Host Configuration Protocol ehk dünaamiline masinakonfiguratsiooni protokoll. See protokoll on mõeldud kohtvõrgu masinatele, et need saaksid dünaamiliselt serverist IP-aadressi ja muud võrguseadistused.

kataloog
Failisüsteemi struktuuri osa. Kataloogis võivad paikneda failid või teised kataloogid (viimaseid nimetatakse sel juhul alamkataloogiks). Sellist failisüsteemi ülesehitust nimetatakse sageli kataloogipuuks. Kui soovite näha, mis mõnes kataloogis asub, tuleb anda vastav käsk. Mõnikord nimetatakse puuanaloogiast lähtudes ka kataloogis asuvaid faile lehtedeks ning alamkatalooge harudeks. Kataloogide kohta kehtivad samasugused õigused nagu failidegi kohta, kuigi nende konkreetne tähendus võib mõnevõrra erineda. Spetsiaalsed kataloogid `.` ja `..` tähistavad vastavalt antud kataloogi ennast ja selle ülemkataloogi. Eriti graafilises töökeskkonnas nimetatakse katalooge vahel ka kaustadeks.

diskreetsed väärtused
Mittepidevad väärtused, see tähendab, et üksteisele järgnevate väärtuste vahel on teatud "ruum".

distributsioon
Selle mõistega eristatakse ühe GNU/Linuxi tootja toodet teistest. Distributsioon koosneb Linuxi kernelist ja utiliitidest, samuti paigaldamisprogrammist, kolmandate osapoolte programmidest ja vahel ka firmaomasest tarkvarast.

DLCI
Data Link Connection Identifier ehk andmesideühenduse identifikaator. Seda kasutatakse unikaalse virtuaalse punktidevahelise ühenduse tuvastamiseks kaadriretranslaatorvõrgus (Frame Relay network). DLCI-d omistab tavaliselt kaadriretranslaatorvõrgu pakkuja.

DMA
Direct Memory Access ehk otsemällupöördus. PC arhitektuuri omadus, mis võimaldab välisseadmel põhimälust lugeda või sellesse kirjutada ilma protsessorit (CPU) kaasamata. PCI välisseadmed kasutavad siinihaldurit ega vaja DMA-d. Siinihaldur võimaldab kontrolleril suhelda muude seadmetega ilma CPU vahenduseta.

DNS
Domain Name System ehk domeeninimede süsteem. See on Interneti nimede ja aadresside jagamise süsteem. See võimaldab siduda domeeninime IP-aadressiga, mis lubab otsida saite domeeninime järgi, ilma

et oleks vaja teada selle IP-aadressi. DNS võimaldab ka pöördotsingut ehk siis masina IP-aadressi tuvastamist selle nime põhjal.

DPMS
Display Power Management System ehk kuva energiahalduse süsteem. Protokoll, mida kõik moodsad monitorid kasutavad energiasäästmise võimaluste pakkumiseks. Selliseid monitore kutsutakse tihtipeale "rohelisteks" monitorideks.

echo
Otsetõlkes tähendab kaja ning sellega tähistatakse olukorda, kus sümbolid, mida Te kirjutate, ilmuvad ekraanil nähtavale (näiteks kasutajanime kirjutamise korral). Mõned programmid maskeerivad turvakaalutlustel vähemalt osa sisestatavast infost. Selle näiteks on parooli kirjutamine, mille korral iga sümboli asemel näidatakse tärni (⋆) või ei näidata üldse midagi.

redaktor
Seda mõistet kasutatakse kõige enam seoses programmidega, mis on mõeldud teksti redigeerimiseks (siit ka nimetus tekstiredaktor). GNU/Linuxi tuntuimad redaktorid on GNU redaktor Emacs (Emacs) ja UNIX® redaktor Vi.

ELF
Executable and Linking Format ehk käivitatav ja lingitav failivorming. See on enamikus GNU/Linuxi distributsioonides kasutatav binaarfailivorming.

e-post
Tähendab elektron- ehk arvutiposti või viisi, kuidas saadetakse elektrooniliselt sõnumeid. Sarnaselt tavalistele kirjadele (niinimetatud tigupost) vajab e-post kirjade kohaletoimetamiseks sihtkohta ja saatja aadressi. Viimane esineb kujul "saatja@saatja.domeen", saajal peab aga olema aadress kujul "saajad@saaja.domeen". E-post on väga kiire suhtlemiskanal ning enamasti kulub kirja jõudmiseks adressaadini vaid mõni hetk või minut sõltumata sellest, millises maailma punktis ta füüsiliselt asub. E-kirja kirjutamiseks on vajalik e-posti klient, olgu siis tekstirežiimis (näiteks pine või `mutt`) või graafiline (näiteks KMail).

keskkond
Protsessi täitmise kontekst. See hõlmab kogu infot, mida operatsioonisüsteem vajab protsessi haldamiseks ning protsessor protsessi korrektseks täitmiseks.
Vt. ka: protsess.

keskkonnamuutujad
Protsessi keskkonna osa. Keskkonnamuutujaid saab vahetult vaadata shellis.
Vt. ka: protsess.

varjestamine
Shelli kontekstis tähendab sõne ümbritsemist jutumärkidega, et shell ei tõlgendaks antud sõnet käsuna. Kui Teil on näiteks vaja kasutada käsureal tühikuid ja seejärel suunata käsu tulemus toru kaudu mõnele muule käsule, tuleb esimene käsk panna jutumärkidesse või lisada tühikute ette `\` (varjestamise ehk "escape"-käsk), sest muidu tõlgendab shell seda valesti ja käsk ei anna oodatud tulemust.

ext2
Lühend väljendist "Extended 2 file system" ehk Teine laiendatud failisüsteem. See on GNU/Linuxile iseloomulik failisüsteem ja sellele on mased kõigi UNIX® failisüsteemide tunnused: spetsiaalsete failide (sümbolseadmed, nimeviidad jne.) toetus, failiõigused ja nii edasi.

FAQ
Frequently Asked Questions ehk korduma kippuvad küsimused (eesti keeles kasutatakse ka lühendit KKK). Dokument, mis sisaldab rea küsimusi ja vastuseid mingi konkreetse teema kohta. Ajalooliselt levisid FAQ-id uudistegruppides, kuid tänapäeval võib sedalaadi dokumente kohata paljudel veebisaitidel ning oma FAQ on sageli ka kommertstarkvaral. Üldiselt võib neid pidada päris heaks infoallikaks.

FAT
File Allocation Table ehk failipaigutustabel. Failisüsteem, mida kasutavad DOS ja Windows®.

FDDI
Fiber Distributed Digital Interface ehk kiudlevi-andmeliides. Ülikiirete võrkude füüsiline kiht, mis kasutab sidepidamiseks telefonijuhtme asemel optilist kaablit. Enamasti kasutusel suurtes võrkudes, sest see maksab üsna palju. Väga harva kasutatakse PC ja võrgukommutaatori ühendusvahendina.

FHS
File system Hierarchy Standard ehk failisüsteemi hierarhia standard. Dokument, mis sisaldab UNIX® süsteemide korrektse failipuu korraldamise juhiseid. Mandriva Linux vastab sellele standardile sisuliselt igas punktis.

FIFO
First In, First Out ehk 'esimesena sisse, esimesena välja'. Andmestruktuur või riistvaraline puhver, millest elemendid võetakse välja samas järjekorras, nagu need sisestati. FIFO üheks levinumaks näiteks on UNIX® torud.

failisüsteem
Skeem, mida kasutatakse failide salvestamiseks füüsilisele andmekandjale (kõvaketas, diskett jne.) ühetaolisel ja ühtlasel viisil. Failisüsteemide näideteks on FAT, GNU/Linuxi ext2fs, ISO9660 (seda kasutavad CD-ROM-id) jne. Virtuaalse failisüsteemi näiteks on `/proc` failisüsteem.

tulemüür
Masin või spetsiaalne riistvaraline komponent, mis kohtvõrgu topoloogias kujutab endast ainsat ühenduspunkti välise võrguga ja mis filtreerib ning kontrollib tegevust teatud portides või tagab, et ainult teatud liidestel on ligipääs välisele võrgule või et väljastpoolt pääseb ligi ainult teatud liidestele.

lipp
Indikaator (tavaliselt bitt), mida kasutatakse märguandmiseks programmi teatud oleku kohta. Nii on näiteks failisüsteemil muu hulgas lipp, mis annab teada, kas seda tuleb varundada või mitte: kui lipp on aktiveeritud, siis tehakse failisüsteemist varukoopia, kui aktiveerimata, siis mitte.

fookus
Akna olek, mille korral see saab vastu võtta klaviatuurisündmusi (klahvile vajutamisi, klahvi vabastamisi ja hiireklõpse), kui aknahalduri seadistustes ei ole teisiti määratud.

esiplaan
Shelli kontekstis on esiplaani protsess selline, mis parajasti töötab ja mis kontrollib klaviatuuri ja ekraani. Uue käsu sisestamiseks peab ootama, kuni selline protsess töö lõpetab.
Vt. ka: töö, taust.

kaadriretranslaator
Kaadriretranslaator ehk Frame Relay on võrgutehnoloogia, mis sobib ideaalselt ainult aeg-ajalt või teatud 'pursetena' toimuva võrguliikluse korral. Võrgukulusid saab sel juhul kokku hoida, sest kaadriretranslaatori kasutajad jagavad üht ja sama võrgumahtu.

kaadripuhver
Videokaardi mälu (RAM) projektsioon masina aadressiruumis. See võimaldab rakendustel kasutada videomälu ilma vajaduseta otseselt kaardiga suhelda. Kaadripuhvrit kasutavad kõik moodsad graafilised tööjaamad.

FTP
File Transfer Protocol ehk failiedastusprotokoll on Interneti standardprotokoll failide toimetamiseks ühest masinast teise.

täisekraan
Seda mõistet kasutatakse rakenduste korral, kui need hõlmavad kogu Teie ekraani nähtava osa.

lüüs
Lüüs (inglise keeles gateway) on masin või seade, mis tagab kohtvõrgule ligipääsu välisele võrgule.

GFDL
GNU Vaba Dokumentatsiooni Litsents. See litsents kehtib kogu Mandriva Linuxi dokumentatsioonile.

GIF
Graphics Interchange Format ehk graafikavahetuse vorming on pildifailide vorming, mida kasutatakse laialdaselt veebis. GIF-pilte saab tihendada või animeerida. Autoriõiguse probleemide tõttu ei ole nende kasutamine väga mõttekas ja märksa parem oleks need võimaluse korral asendada PNG-vormingus piltidega.
Vt. ka: PNG.

metamärgid
See tähendab shelli kontekstis oskust grupeerida teatud failinimed vastavalt etteantud mustrile.
Vt. ka: metamärkidega muster.

metamärkidega muster
Sõne, mis koosneb tavalistest ja spetsiaalsetest sümbolitest. Viimaseid tõlgendab ja laiendab shell.

GNU
GNU's Not Unix ehk 'GNU ei ole Unix'. GNU projekti algatas Richard Stallman 1980. aastate algul, üritades luua vaba operatsioonisüsteemi ("vaba" esineb siin sellises tähenduses nagu väljendis "kõnevabadus"). Praeguseks on olemas kõik operatsioonisüsteemi tööriistad, välja arvatud... kernel. GNU projekti kernel nimetusega Hurd ei ole lihtsalt veel igas mõttes töökindel. Linux on muu hulgas võtnud GNU käest üle kaks asja: C-keele kompilaatori `gcc` ja litsentsi GPL.
Vt. ka: GPL.

GPL
General Public License ehk Üldine Avalik Litsents. See on GNU/Linuxi kerneli litsents, mille eesmärk on sootuks vastupidine kommertstarkvara litsentsidele selles mõttes, et see ei piira tarkvara kopeerimist, muutmist ja levitamist tingimusel, et kättesaadav peab olema lähtekood. Ainus tegelik piirang on see, et isik, kellele Te tarkvara levitate, peab saama seda kasutada samade õigustega nagu Teie ise.

GUI
Graphical User Interface ehk graafiline kasutajaliides. Liides arvutis, mis koosneb menüüde, nuppude, ikoonide ja muu sellisega akendest. Enamiku kasutajate arvates on GUI etem kui CLI ehk käsurida (*Command Line Interface*), kuigi viimane on peaaegu igal juhul märksa võimsam.

guru
Ekspert, asjatundja. Enamasti kasutatakse inimeste kohta, kellel on suurepärased oskused või kes osutab teistele väärtuslikku abi.

riistvaraline aadress
See on arv, mis tuvastab unikaalselt masina füüsilises võrgus andmekandja ligipääsu kihis. Selle näideteks on **Etherneti aadressid** ja **>AX.25 aadressid**.

peidetud fail
Fail, mida ei saa "näha" käsku `ls` ilma spetsiaalsete võtmeteta andes. Peidetud failide nimede alguses seisab punkt (`.`) ning neid tarvitatakse kasutaja isiklike eelistuste ja programmide seadistuste salvestamiseks. Nii on näiteks bash'i käskude ajalugu salvestatud peidetud faili `.bash_history`.

kodukataloog
Sageli ka lihtsalt "kodu". See tähistab konkreetse kasutaja isiklikku kataloogi.
Vt. ka: konto.

masin
Tähendab arvutit, kasutatakse eriti siis, kui jutuks on võrku ühendatud arvutid.

HTML
HyperText Markup Language ehk hüperteksti märkekeel. Seda kasutatakse veebidokumentide loomiseks.

HTTP
HyperText Transfer Protocol ehk hüperteksti edastusprotokoll. Seda kasutatakse ühenduse loomiseks veebisaitidega ja HTML-dokumentide või failide hankimiseks.

ikoon
Väike pilt (tavaliselt mõõdus 16×16, 32×32, 48×48 või vahel ka 64×64 pikslit), millega graafilises töökeskkonnas tähistatakse dokumenti, faili või programmi.

IDE
Integrated Drive Electronics ehk integreeritud ajamielektroonika. Tänapäeva PC-de juures kõige levinum kõvaketaste siin. IDE siin võib hõlmata kuni kaht seadet ning siini kiirust piirab aeglasema käsujärjekorraga (mitte aga aeglasema edastamiskiirusega!) seade.
Vt. ka: ATAPI, SATA, S-ATA.

IMAP
Internet Message Access Protocol ehk Interneti sõnumipöördusprotokoll. See võimaldab pääseda ligi võrguserveris asuvatele e-kirjadele ilma vajaduseta need kõigepealt kohalikku masinasse laadida. Selle poolest erineb see e-kirjade hankimise POP protokollist.
Vt. ka: POP.

infosõlm
Infosõlm ehk inode on sisenemispunkt, mis juhatab UNIX®-laadsetes failisüsteemides faili sisu juurde. Infosõlm on määratud unikaalse numbriga ja see sisaldab viidatava faili metainfot, näiteks selle kasutamise aegu, tüüpi ja suurust, **aga mitte selle nime!**

lisamisre iim
Vi ja selle kloonide korral tähendab programmi olekut, mille puhul klahvile vajutades lisatakse vajutatud klahvi märk redigeeritavasse faili (välja arvatud väga erandlikel juhtudel, nagu näiteks lühendi lõpetamine, paremjoondus rea lõpuga vms.). Sellest režiimis saab väljuda, kui vajutada klahvile **Esc** (või **Ctrl-[**).

Internet
Hiiglaslik võrk, mis ühendab kogu maailma arvuteid.

IP-aadress
Numbriline aadress, mis koosneb (versioonis 4 ehk IPv4) neljast osast ja mis tuvastab Teie arvuti võrgus. IP-aadressid on struktureeritud hierarhiliselt: tipptase ja riiklikud domeenid, domeenid, alamdomeenid ja iga masina isiklik aadress. IP-aadress näeb välja umbes selline: `192.168.0.1`. Masina aadress võib olla kas staatiline või dünaamiline. Staatilised IP-aadressid on muutumatud, need omistatakse püsivalt. Dünaamilised IP-aadresside korral muutub IP-aadress igal võrkuühendumisel. Enamikul kodukasutajatest on tavaliselt dünaamiline IP-aadress, enamikul ettevõtetest kasutajatel aga staatiline IP-aadress.

IP maskeerimine
Selle tehnoloogiaga varjab tulemüür Teie arvuti tegeliku IP-aadressi välismaailma pilgu eest. Tüüpiliselt saavad kõik läbi tulemüüri tehtud välisühendused tulemüüri IP-aadressi. See on kasulik näiteks siis, kui Teil on kiire Internetiühendus ja ainult üks IP-aadress, aga Te tahate oma sisevõrgus kasutada rohkem arvuteid.

IRC
Internet Relay Chat ehk Interneti retranslatsioonivestlus või rühmadiskussioon Internetis. Üks väheseid Interneti reaalajas vestlemise standardeid. See võimaldab luua kanaleid, pidada eravestlusi ja vahetada faile. Samuti lubab see serveritel üksteisega ühenduda, mistõttu tänapäeval ongi mitu IRC võrku, millest tuntumad on vahest **Undernet**, **DALnet** ja **EFnet**.

IRC kanalid
"Kohad" IRC kanalites, kus saab teiste inimestega vestelda. Kanalid luuakse IRC serverites ning kasutajad liituvad nendega, et teistega suhelda. Mingil kanalil kirjutatud sõnumid on näha ainult antud kanaliga liitunud kasutajatele. Kaks või rohkem kasutajat võivad luua "privaatse" kanali, kus teised kasutajad neid ei sega. Kanalinimede alguses seisab `#`.

ISA
Industry Standard Architecture ehk tööstuslik standardarhitektuur. Esimene PC-del kasutatud siin, millest tasapisi loobutakse PCI siini kasuks. ISA on endiselt levinud skännerite, CD-kirjutite ja muu vanema riistvara jaoks kasutatavatel SCSI-kaartidel.

ISDN
Integrated Services Digital Network ehk integreeritud teenuste digitaalvõrk. Heli, digitaalsete võrguteenuste ja video kommunikatsioonistandardite kogum. See loodi sihiga asendada senine telefonisüsteem, niinimetatud PSTN (*Public Switched Telephone Network* ehk fikstelefonivõrk) või POTS ()*Plain Old Telephone Service* ehk vana hea telefoniteenus, analoogtelefoniteenus). ISDN-i nimetatakse ka kanalikommutatsiooniga andmesidevõrguks.

ISO
International Standards Organization ehk Rahvusvaheline Standardiseerimisorgranisatsioon. Ettevõtete, konsultantide, ülikoolide ja muude organisatsioonide ühendus, mis tegelevad paljudes valdkondades, sealhulgas arvutiasjanduses, standardite väljatöötamisega. Standardeid kirjeldavad dokumendid kannavad numbreid. Standard ISO-9660 kirjeldab näiteks CD-ROM-idel kasutatavat failisüsteemi.

ISO-8859
Standard ISO-8859 hõlmab mitmeid ASCII märgistiku 8-bitiseid laiendusi. Eriti oluline on ISO-8859-1 ehk "ladina tähestik nr. 1", mis on väga laialdaselt levinud ja mida võib õigupoolest pidada standardse ASCII *de facto* asendajaks.
ISO-8859-1 toetab järgmisi keeli: afrikaani, baski, katalaani, taani, hollandi, inglise, fääri, soome, prantsuse, galeegi, saksa, islandi, iiri, itaalia, norra, portugali, šoti, hispaania ja rootsi.
Pange tähele, et ISO-8859-1 sümbolid on ühtlasi ISO-10646 (Unicode) esimesed 256 sümbolit. Kuid selles puudub eurosümbol ja see ei hõlma täielikult soome ja prantsuse keelt. Nende puuduste likvideerimiseks loodi ISO-8859-1 modifikatsioon ISO-8859-15.
Vt. ka: ASCII, UTF-8.

ISP
Internet Service Provider ehk Internetiteenuste pakkuja. Firma, mis müüb tarbijatele Interneti kasutamise võimalust kas telefoniliini või lairibakanali vahendusel (näiteks T1, DSL või kaabliühendus).

JPEG
Joint Photographic Experts Group ehk Ühendatud Fotograafiaekspertide Grupp. Organisatsioon, mis töötas välja samanimelise, praeguseks väga levinud pildifailivormingu. JPEG sobib kõige paremini fotograafiliste piltide, mitte aga ebareaalsete piltide jaoks.

töö
Kasutatuna shelli kontekstis tähendab töö protsessi, mis töötab taustal. Ühes ja samas shellis võib töötada mitu tööd ning neid saab teineteisest sõltumatult juhtida.
Vt. ka: esiplaan, taust.

kirjendamine
Kirjendamine suurendab failisüsteemi töökindlust, muutes selle tehinguliseks. See tähendab, et andmete füüsilise kirjutamise asemel hetkel, kui selleks soovi avaldatakse, pannakse kirja nii-öelda päevikukirje ning andmed ise kirjutatakse "plokina" millalgi hilje. See mõjutab oluliselt ka jõudlust ning aega, mis kulub failisüsteemi analüüsimisele ning vajaduse korral parandamisele.

kernel
Kernel on operatsioonisüsteemi tuum. See vastutab ressursside eraldamise ning protsesside eristamise eest. See tegeleb kõigi süvatasemeoperatsioonidega, mis võimaldavad programmidele otseselt suhelda Teie arvuti riistvaraga, haldab puhvermälu jne.

surmaring
Emacsis tähendab surmaring (kill ring) pärast redaktori käivitamist lõigatud või kopeeritud tekstialade kogumit. Tekstialasid saab taas välja kutsuda, et neid redigeeritavasse faili lisada, ning sisuliselt ongi selle kogumi struktuur ringikujuline.

LAN
Local Area Network ehk kohtvõrk. Tavaline väljend masinate võrgu kohta, mis on ühendatud ühtse füüsilise juhtmestikuga (või muude vahenditega) piiratud geograafilisel alal, enamasti ühes asutuses või hoones.
Vt. ka: WAN.

käivitamine
Programmi väljakutsumine või töölepanemine.

teek
Protseduuride ja funktsioonide kogum binaarkujul, mida programmeerijad kasutavad oma programmides (niivõrd, kuivõrd seda lubab teegi litsents). Programmi, mis vastutab jagatud teekide laadimise eest käitusajal, nimetatakse dünaamiliseks linkuriks.

link
Viit infosõlmele kataloogis, mis ühtlasi annab infosõlmele (faili)nime. Infosõlmed, millel ei ole linki (ja sestap ka nime), on näiteks anonüümsed torud (neid kasutab shell), soklid ehk võrguühendused, võrguseadmed jne.

linkimine
Kompileerimise viimane aste, mis koosneb kõigi objektfailide linkimisest käivitatava faili loomiseks ning lahendamata sümbolite sidumisest dünaamiliste teekidega (välja arvatud siis, kui nõutav on staatiline linkimine, mille korral nende sümbolite kood lisatakse käivitatavasse faili).

Linux
UNIX®-laadne operatsioonisüsteem, mis võib töötada üsna erinevates arvutites ja mida kõik võivad vabalt kasutada ning muuta. Linuxi (kerneli) kirjutas Linus Torvalds.

kasutajatunnus
Nimi, millega kasutaja saab ennast UNIX® süsteemis tuvastada ja süsteemi kasutada.

Päringutabel
Tabel, milles on salvestatud koodid (või sildid) ja nende tähendused. Sageli on see andmefail, mida mingi programm kasutab konkreetse elemendi kohta lisainfo hankimiseks.
Näiteks HardDrake kasutab sellist tabelid tootjate tootekoodide ja nendega seotud konfiguratsiooniinfo salvestamiseks. Üks selle tabeli ridu, mis annab teavet `CTL0001` kohta, näeb välja selline

```
"CTL0001"       "sb"    "Creative Labs|SB16"    "sound" "HAS_OPL3|HAS_MPU401|HAS_DMA16|HAS_JOYSTICK"
```

loopback
Virtuaalne võrguliides, mis tähistab masinat ennast. See lubab töötavatel programmidel jätta arvestamata spetsiaalsed juhud, mis kaks võrguolemit on tegelikult üks ja sama masin.

major
Seadmeklassi määrav arv.

manuaalilehekülg
Väikesed dokumendid, mis kirjeldavad käsku ja selle kasutamist. Neid saab uurida käsuga `man`. Üks esimesi kohti, kuhu vaadata, kui soovite mõne vähetuntud või tundmatu käsu kohta põhjalikumat infot näha.

MBR
Master Boot Record ehk alglaadimissektor. Sellist nime kannab esimene sektor kõvakettal, millelt sooritatakse alglaadimine. MBR sisaldab koodi, mida kasutatakse operatsioonisüsteemi laadimiseks mällu või alglaadurit (näiteks LiLo) ning kõvaketta partitsioonitabelit.

MIME
Multipurpose Internet Mail Extensions ehk universaalsed Internetiposti laiendused. Sõne kujul `tüüp/alamtüüp`, mis kirjeldab e-kirjale lisatud faili sisu. See lubab MIME-teadlikel e-posti klientidel määrata võimaliku toimingu vastavalt failitüübile.

minor
Arv, mis määrab konkreetse seadme.

MPEG
Moving Picture Experts Group ehk Filmiekspertide grupp. ISO komisjon, mis töötab välja video- ja audiotihenduse standardeid. MPEG on ühtlasi ka nende loodud algoritmide nimi. Paraku on selle vormingu litsents üpris piirav, mistõttu on ka suhteliselt vähe avatud lähtekoodiga MPEG-mängijaid...

haakepunkt
Koht või kataloog, kuhu GNU/Linuxi failisüsteemis on ühendatud partitsioon või seade. Näiteks Teie CD-ROM on haagitud kataloogis `/mnt/cdrom`, kus võite uurida kõigi ühendatud CD-de sisu.

haakimine
Seade on haagitud, kui see on ühendatud GNU/Linuxi failisüsteemi. Seadme haakimise järel saate uurida selle sisu. Mõnevõrra on selle mõiste välja tõrjunud uus võimalus, niinimetatud "supermount", mille korral kasutajad ei pea enam eemaldatavaid andmekandjaid ise käsitsi ühendama.
Vt. ka: haakepunkt.

MSS
Maximum Segment Size ehk segmendi maksimaalne suurus on suurim andmehulk, mida on võimalik ühekorraga liidese kaudu edastada. Kui soovite vältida kohalikku fragmenteerumist, peaks MSS võrduma IP päise MTU väärtusega.

MTU
Maximum Transmission Unit ehk maksimaalne edastusühik on parameeter, mis määrab kindlaks suurima datagrammi, mida saab IP-liidese kaudu edastada ilma vajaduseta jagada see väiksemateks ühikuteks. MTU peab olema suurem kui suurim datagramm, mida Te soovite edastada ilma jagamata. Pange tähele, et see väldib ainult kohalikku fragmenteerumist: mõnes punktis andmete teekonnal võib olla määratud

väiksem MTU ja seal jagatakse datagramm ikkagi. Tüüpväärtused on 1500 baiti Ethernet-liidese või 576 baiti PPP-liidese korral.

multitegumtöö
Operatsioonisüsteemi oskus jagada protsessori tööaega mitme protsessi vahel. Madaltasemel kasutatakse selle kohta ka mõistet multiprogrammtöö. Ühelt protsessilt teisele lülitumine nõuab aktiivse protsessi konteksti salvestamist ja selle taastamist, kui protsess uuesti tööle asub. Seda operatsiooni nimetatakse kontekstkommutatsiooniks ning see võetakse ette mitu korda sekundis, mis jätab kasutajale mulje, nagu hoiaks operatsioonisüsteem korraga töös mitut rakendust. Multitegumtööd on kaht liiki: tõrjuva multitegumtöö korral vastutab operatsioonisüsteem protsessori eraldamise eest protsessidele, mittetõrjuva ehk võrdõigusliku multitegumtöö korral juhivad protsessid ise protsessorit. Esimene variant, mida kasutab GNU/Linux, on mõistagi eelistatum, sest nii ei saa ükski programm haarata endale kogu protsessori aega ja teisi protsesse blokeerida. Viisi, kuidas mitme parameetri põhjal valitakse protsess, mida parajasti tööle panna, nimetatakse planeerimiseks.

mitmekasutajasüsteem
Seda mõistet kasutatakse operatsioonisüsteemi kohta, mis võimaldab paljudel kasutajatel süsteemi sisse logida ja seda kasutada ühel ja samal ajal, kusjuures iga kasutaja saab tegutseda sõltumatult teistest. Mitmekasutajasüsteemi peab tagama operatsioonisüsteem. GNU/Linux on ühtaegu nii multitegumtööd kui ka multikasutajasüsteemi pakkuv operatsioonisüsteem (nagu tegelikult kõik UNIX® süsteemid).

nimega toru
UNIX® korral lingitud toru (erinevalt torudes, mida kasutab shell).
Vt. ka: toru, link.

nimetamine
Tähistab tavaliselt objektide tuvastamise meetodit. Sageli räägitakse failide, programmi funktsioonide ja muu sellise "nimetamiskonventsioonidest".

NCP
NetWare Core Protocol ehk NetWare'i tuumprotokoll. Selle protokolli töötas **Novell** välja Novell Netware-i faili- ja printeriteenuste kasutamiseks.

NFS
Network File System ehk võrgufailisüsteem. Selle töötas välja **Sun Microsystems** failide läbipaistvaks jagamiseks võrgus.

uudistegrupid
Diskussiooni- ja uudistealad, millele pääseb ligi uudiste- või USENET-i klientidega, mis võimaldavad temaatiliste uudistegruppide uudiseid lugeda ja neid ise postitada. Nii on näiteks uudistegrupp `alt.os.linux.mandrake` alternatiivne (alt) uudistegrupp, mis tegeleb operatsioonisüsteemiga (os) GNU/Linux (linux), konkreetsemalt aga Mandriva Linuxiga (mandrake). Uudistegrupid on rühmitatud nii, et oleks võimalik maksimaalselt hõlpsalt leida huvipakkuvale teemale vastav grupp.

NIC
Network Interface Controller ehk võrgukontroller või võrguliides. Arvutisse paigaldatud adapter, mis tagab füüsilise ühenduse võrguga (näiteks Ethernet-kaart).

NIS
Network Information System. NIS kandis varem ka nime "Yellow Pages", kuid selle õigused kuulusid firmale **British Telecom**. NIS on protokoll, mille **Sun Microsystems** töötas välja üldise info jagamiseks NIS-**domeenis**, mis võib koosneda nii tervest kohtvõrgust (LAN) või ka selle mingist osast. Selles saab eksportida paroolide andmebaase, teenuste andmebaase, gruppide infot jms.

null, sümbol
Sümbol või bait 0. Seda kasutatakse sõne lõpu tähistamiseks.

objektkood
Kompileerimise ajal genereeritud kood, mis lingitakse käivitatava faili loomiseks muude objektkoodide ja teekidega. Objektkood on masinloetav.
Vt. ka: kompileerimine, linkimine.

lennult
"Lennult" tehakse midagi siis, kui tehtav toimub muude toimingute kõrval, ilma et Te seda märkakski (või vähemalt ei nõuta Teie käest sellesse sekkumist).

avatud lähtekood
Nii nimetatakse programmi vaba lähtekoodi, mis on tehtud avalikuks nii arendajatele kui üldse kogu maailmale. Selle põhjenduseks on veendumus, et kui lubada lähtekoodi kasutada ja muuta suuremal programmeerijate seltskonnal, on lõpptulemusena loodud toode märksa töökindlam ja kasulikum. Populaarsete avatud lähtekoodiga programmide hulka kuuluvad näiteks Apache, sendmail ja GNU/Linux.

operatsioonisüsteem
Liides rakenduste ja riistvara vahel. Kõigi operatsioonisüsteemide peamiseks ülesandeks on hallata masinale omaseid ressursse. GNU/Linuxi süsteemi korral tegelevad sellega kernel ja laaditavad moodulid. Muud tuntumad operatsioonisüsteemid on Amiga®OS, Mac OS®, FreeBSD®, OS/2®, UNIX® ja Windows® kõigis oma variantides.

omanik
Kasutajate ja nende failide kontekstis tähendab faili omanik kasutajat, kes selle faili lõi.

omanikugrupp
Gruppide ja nende failide kontekstis on faili omanikugrupp grupp, millesse kuulub antud faili loonud kasutaja.

PAP
Password Authentication Protocol ehk paroolautentimise protokoll. Seda protokolli kasutavad paljud ISP-d oma klientide autentimiseks. Selle kohaselt saadab klient (Teie) serverile oma identifikaatori-paroolipaari, kuid krüptimata kujul. CHAP on märksa turvalisem ja sestap eelistatum autentimisprotokoll.
Vt. ka: CHAP.

lehitseja
Programm, mis näitab tekstifaili ekraanikaupa, lubades hõlpsasti failis edasi-tagasi liikuda ning sõnesid otsida. Erinevatest lehitsejatest soovitame kasutada programmi `less`.

parool
Salasõna või sõnade või tähtede kombinatsioon, mida kasutatakse millegi turvamiseks. Paroole kasutatakse koos kasutajanimedega mitmekasutajasüsteemides, veebilehekülgedel, FTP-saitidel ja mujal. Parool peaks olema raskesti äraarvatav fraas või tähtede ja numbrite kogum ning üldjuhul ei tohiks selleks olla mõni üldlevinud sõna. Paroolide ülesanne on tagada, et teised inimesed ei saaks arvutisse või saidile sisse logida Teie kontoga.

paik, paikamine
Fail, mis sisaldab lähtekoodi tehtavaid parandusi, millega lisatakse uusi omadusi, kõrvaldatakse vigu või muudetakse seda mingil muul viisil vastavalt paiga valmistaja soovidele ja vajadustele. Vastav toiming ehk "paikamine" tähendabki nende muudatuste lisamist lähtekoodile.

otsingutee
Otsingutee ehk rada tähistab failide ja kataloogide asukohta failisüsteemis. Otsingutee erinevaid lõike eraldab "kaldkriips" ('/'). GNU/Linuxi süsteemides kasutatakse kaht tüüpi otsinguteid. **Suhteline** otsingutee on faili või kataloogi asukoht parajasti aktiivse kataloogi suhtes. **Absoluutne** või **täielik** otsingutee on faili või kataloogi asukoht juurkataloogi suhtes.

PCI
Peripheral Component Interconnect ehk välisseadmeühendus on siin, mille lõi **Intel** ja millest tänaseks on saanud PC ja muude arhitektuuride standardsiin. See on ISA järelkäija ning pakub arvukalt teenuseid: seadme tuvastamine, seadistusinfo, IRQ jagamine, siinihaldus jne.

PCMCIA
Personal Computer Memory Card International Association. Väga sageli nimetatakse seda ka "PC-kaardiks" ning see on muutunud sülearvutiga ühendatud väliste kaartide standardiks: modemin, kõvakettad, mälukaardid, Ethernet-kaardid jms. Lühendit tõlgendatakse vahel küll ka humoorikamalt: *People Cannot Memorize Computer Industry Acronyms* ehk 'Inimesed ei suuda arvutilühendeid ära õppida'...

toru
Spetsiaalne UNIX® failitüüp. Üks programm kirjutab andmed torusse, teine programm aga loeb selle teisest otsast väljuvad andmed. UNIX® torud on FIFO-d, nii et andmed loetakse just selles järjekorras, nagu need saadeti. Väga laialdaselt kasutusel shellis. Vaata ka **nimega torud**.

pixmap
Lühend sõnadest "pixel map" ehk pikslikaart. See on lihtsalt teine väljend bittrasterpiltide kohta.

plugin
Lisaprogramm, mida kasutatakse veebidokumendi multimeediasisu esitamiseks. Tavaliselt saab selle vähese vaevaga Internetist tõmmata, kui Teie veebilehitseja ei peaks veel suutma antud laadi informatsiooni esitada.

PNG
Portable Network Graphics ehk porditav võrgugraafika. Peamiselt veebis kasutamiseks mõeldud pildifailide vorming. See loodi patendivaba asendusena GIF-vormingule, kuid pakub ka mõningaid lisavõimalusi.

PnP
Plug'N'Play ehk 'Ühenda ja mängi'. Esimene ISA lisandus, mille eesmärk oli anda seadmete kohta seadistusinfot. Hiljem muutus laiemaks mõisteks, mis tähistab kõiki seadmeid, mis oskavad oma seadistusparameetrid teada anda. Kõik PCI seadmed on Plug'N'Play.

POP
Post Office Protocol ehk postkontoriprotokoll. Üks levinumaid protokolle e-kirjade tõmbamiseks ISP-lt. Veel ühe kaugligipääsu pakkuva e-posti protokolli näiteks on IMAP.
Vt. ka: IMAP.

portimine
Üks kahest võimalusest panna programm tööle süsteemis, millele see ei olnud algselt mõeldud. Näiteks selleks, et Windows® jaoks mõeldud programm töötaks ilma abivahenditeta GNU/Linuxis, tuleb see kõigepealt GNU/Linuxisse portida.

PPP
Point to Point Protocol ehk kakspunktprotokoll. Seda protokolli kasutatakse andmete saatmiseks jadaliinide kaudu. Tavaliselt kasutatakse seda IP-pakettide saatmiseks Internetti, kuid seda võib pruukida ka koos selliste protokollidega, nagu näiteks Novelli IPX.

tehtejärjestus
Määrab avaldise liikmete käsitlemise järjekorra. Kui tehteks on näiteks `4 + 3 * 2`, on tulemuseks 10, sest korrutamine seisab tehtejärjekorras kõrgemal positsioonil kui liitmine. Kui soovite, et kõigepealt arvud liidetaks, tuleb kasutada sulge: `(4 + 3) * 2`. Nüüd on tulemuseks 14, sest sulud omakorda seisavad kõrgemal positsioonil kui liitmine ja korrutamine, mistõttu kõigepealt sooritatakse sulgudes olev tehe.

eelprotsessorid
Kompileerimisdirektiivid, mis annavad kompilaatorile korralduse asendada direktiivid lähtefailis kasutatavas programmeerimiskeeles kooduga. Näiteks C-keele eelprotsessorid on `#include`, `#define` jne.

protsess
Operatsioonisüsteemi kontekstis on protsess käivitatud programm koos keskkonnaga.

viip
See tähistab shellis kursori ees asuvat stringi. Seda nähes võite kirjutada käsu.

protokoll
Protokollid korraldavad erinevate masinate suhtlemist võrgus riistvara või tarkvara kasutades. Nad määravad kindlaks edastatavate andmete vormingu, selle, milline masin kontrollib millist jms. Tuntumateks protokollideks on HTTP, FTP, TCP ja UDP.

puhverserver
Masin, mis paikneb võrgu ja Interneti vahel ning mille ülesandeks on suurendada enamlevinud protokollid (näiteks HTTP ja FTP) andmeside kiirust. Puhverserver säilitab varasemad päringud oma puhvris, nii et kui mõni masin esitab päringu, mis on juba puhvris, on see võimalik väga kiiresti täita, sest masin saab vastuse kohalikust puhvrist, mitte ei pea seda ootama Internetist. Puhverserverid tulevad eriti

kasuks väikese läbilaskevõimega võrkudes (näiteks modemiühendused). Mõnikord on puhverserver ka ainus masin, millel on õigus välisvõrke kasutada.

rippmenüü
Menüü, mida saab "lahti kerida" selle ühes servas asuva nupuga. Nupule klõpsates "kerib" menüü ennast lahti ja Te näete kõiki selle kirjeid.

kvoot
Meetod kettakasutuse piiramiseks ning kasutajatele limiidi kehtestamiseks. Administraatorid võivad piirata näiteks kasutajate kodukataloogide suurust, määrates teatud failisüsteemile selleks piirkvoodi.

RAID
Redundant Array of Independent Disks ehk sõltumatute ketaste liiasmassiiv. Selle Berkeley ülikooli arvutiteaduskonna algatatud projekti kohaselt salvestatakse andmed erinevaid skeeme kasutades tervele reale ketastele ehk kettamassiivile. Algselt kasutati selleks odavaid vanemaid kettaid, mistõttu esialgu mõtestati lühendit lahti kui *Redundant Array of Inexpensive Disks* ehk odavate ketaste liiasmassiiv.

RAM
Random Access Memory ehk suvapöördusmälu (enamasti kasutatakse küll mõistet muutmälu). See tähistab arvuti põhimälu. "Random" ehk suvaline tähendab seda, et iga mälu osa poole võib otse pöörduda.

kirjutuskaitse
Faili korral tähendab see seda, et faili ei saa kirjutada ehk üle salvestada: Te võite selle sisu lugeda, kuid ei saa seda muuta.
Vt. ka: lugemis-kirjutamisõigus.

lugemis-kirjutamisõigus
Faili tähenduses õigus faili kirjutada ehk salvestada: Te võite lugeda faili sisu ja seda muuta.
Vt. ka: kirjutuskaitse.

regulaaravaldis
Äärmiselt võimas tööriist, mis võimaldab otsida just vajalikke tekstistringe. Regulaaravaldisega saab määrata kindlaks mustri, mida tuleb otsida. Seda kasutavad paljud UNIX® utiliidi: `sed`, awk, grep, perl jne.

RFC
Request For Comments ehk kommentaarisoov. RFC-d on ametlikud Internetistandardi dokumendid, mida avaldab IETF (*Internet Engineering Task Force* ehk Internetiehituse töörühm). Need kirjeldavad kõiki protokolle, nende kasutamist, nõudeid jne. Kui soovite teada, kuidas mingi protokoll töötab, otsige üles vastav RFC.

administraator
Igas UNIX® süsteemis kõige suuremate õigustega kasutaja. Tavaliselt on administraatoriks isik, kes vastutab UNIX® süsteemi hooldamise ja järelevalve eest. Tal on ka süsteemis ligipääs absoluutselt kõigele.

juurkataloog
Failisüsteemi kõrgeima taseme kataloog. Sellel kataloogil ei ole enam ülemkataloogi, mistõttu juurkataloogi korral tähendab '..' sedasama kataloogi. Juurkataloogi tähiseks on '/'.

juurfailisüsteem
See on tipptaseme failisüsteem, kus GNU/Linux haagib juurkataloogipuu. Juurfailisüsteem peab asuma omaette partitsioonil, sest see kujutab endast kogu süsteemi alust. See sisaldab juurkataloogi.

marsruut
Teekond, mille datagrammid läbivad võrgus lähtekohast sihtkohani ehk teekond ühest masinast teise.

RPM
RPM Package Manager ehk- RPM-pakettide haldur. RPM on pakkimisvorming, mille **Red Hat** lõi tarkvara pakendamiseks. Tänapäeval kasutavad seda paljud GNU/Linuxi distributsioonid, kaasa arvatud Mandriva Linux.

käivitustase
Süsteemi tarkvara seadistus, mis lubab ainult teatud konkreetseid valitud protsesse. Iga käivitustaseme lubatud protsessid on kirjas failis `/etc/inittab`. Tavaliselt kasutatakse seitset käivitustaset (0, 1, 2, 3, 4, 5,

ja 6) ning nende vahel lülituda on lubatud ainult privilegeeritud kasutajale, kel on õigus käivitada käsud `init` ja `telinit`.

SATA, S-ATA
Serial ATA ehk jada-ATA. ATA järglane. SATA esimese põlvkonna andmekiiruseks on 1,5 Gbps, kuid jadaühendus ja kasutatav tehnoloogia lubab palju enamati, samal ajal kui paralleel-ATA sisuliseks piiriks on UDMA133.
Vt. ka: ATAPI, IDE.

skript
Sisuliselt on shelliskriptid käskude jada, mis täidetakse samamoodi, nagu oleks nad ükshaaval konsoolis antud. UNIX®i shelliskripte võib mingil määral võrrelda DOS-i pakkfailidega (batch).

SCSI
Small Computers System Interface ehk väikearvutite süsteemiliides. Suure läbilaskevõimega siin, mille eesmärk om võimaldada mitut laadi välisseadmete ühendamist. Erinevalt IDE-st ei ole SCSI siin piiratud kiirusega, millega välisseadmed käske vastu võtavad. Tavaliselt on SCSI siin otse emaplaadile integreeritud ainult väga võimsate masinate korral, enamik PC-sid kasutab aga vastavat lisakaarti.

turvatasemed
Mandriva Linuxi unikaalne omadus, mis võimaldab Teil määrata erinevaid piirangute tasemeid vastavalt sellele, kui turvaliseks Te soovite süsteemi muuta. Eelnevalt on kindlaks määratud kuus taset vahemikus 0 kuni 5, kus 5. tase on kõige rangema turvalisusega. Te võite turvataseme omadused aga ka ise määrata.

segmentimisviga
Segmentimisviga tekib siis, kui programm üritab pöörduda mälu poole, mida talle ei ole eraldatud. Üldiselt põhjustab see programmi töö kohese peatumise.

server
Programm või arvuti, mis pakub teatud võimalust või teenust ning ootab ühendusi **klientidelt**, et täita nende käske või anda neile nõutavat infot. Partnersüsteemi (**peer-to-peer**), näiteks SLIP- või PPP-serveri korral, peetakse serveriks ühenduse seda poolt, millele esitatakse väljakutse, väljakutse esitajat aga kliendiks. See on üks **klient-serversüsteemi** komponent.
Vt. ka: klient, klient-serversüsteem.

variparoolid
Paroolihalduskomplekt UNIX® süsteemides, mille korral krüptitud paroole sisaldav fail ei ole erinevalt tavalisest paroolisüsteemist globaalselt loetav. See pakub ka muid võimalusi, näiteks paroolide aegumine.

shell
shell on operatsioonisüsteemi kerneli peamine liides, mis pakub välja käsurea, kus kasutajad saavad anda käske programmide ja süsteemsete käskude töölepanemiseks. Kõik shellid pakuvad ka mingit skriptikeelt, millega saab ülesandeid automatiseerida või sagedamini kasutatavaid keerukamaid ülesandeid lihtsustada. Mainitud shelliskripte võib võrrelda DOS-i pakkfailidega (batch), kuid need on märksa võimsamad. Shellidest võib nimetada näiteks järgmisi: bash, sh, tcsh.

ainukasutajare iim
See tähistab operatsioonisüsteemi olukorda või isegi operatsioonisüsteemi ennast, kus on on lubatud sisse logida ja süsteemi kasutada korraga ainult ühel kasutajal.

saidipõhine
SEe tähendab, et info, mida sellised programmid nagu imake ja `make` kasutavad lähtefailide kompileerimiseks, sõltub saidist, arvutiarhitektuurist, arvutisse paigaldatud teekidest jne.

SMB
Server Message Block ehk serverisõnumiplokk. Protokoll, mida kasutatakse Windows® masinates failide ja printerite jagamiseks võrgus.
Vt. ka: CIFS.

SMTP
Simple Mail Transfer Protocol ehl lihtne e-posti edastamisprotokoll. See on üldlevinud e-kirjade edstamise protokoll. Paljud e-posti edastusagendid, näiteks sendmail ja postfix, kasutavad SMTP-d. Vahel kutsutakse neid SMTP-serveriteks.

sokkel
Võrguühendusele vastav failitüüp.

pehmed lingid
Vt.: nimeviidad

standardveaväljund
Failideskriptor number 2, mille avab iga protsess ja mida tavapäraselt kasutatakse failideskriptorina, kuhu protsess kirjutab esinenud vead. Tavaliselt on selleks arvuti ekraan.
Vt. ka: standardsisend, standardväljund.

standardsisend
Failideskriptor number 0, mille avab iga protsess ja mida tavapäraselt kasutatakse failideskriptorina, kust protsess hangib andmed. Tavaliselt on selleks arvuti klaviatuur.
Vt. ka: standardveaväljund, standardväljund.

standardväljund
Failideskriptor number 1, mille avab iga protsess ja mida tavapäraselt kasutatakse failideskriptorina, kuhu protsess kirjutab oma väljundi. Tavaliselt on selleks arvuti ekraan.
Vt. ka: standardveaväljund, standardsisend.

striimer
Seade, mille sisendiks on märkide (katkestamata ega väiksemateks tükkideks jagamata)vood ehk "striimid". Tüüpiline striimer on lindiseade.

SVGA
Super Video Graphics Array. Kuvaristandard, mille lõi VESA PC-arhitektuurile. Ekraanilahutus oli algselt 800x600x16 värvi, kuid peagi juba kuni 1024x768x16 värvi ja hiljem rohkemgi.

lüliti
Lülititega saab muuta programmide käitumist. Neid nimetatakse ka käsureavõtmeteks või argumentideks. Kui soovite teada, kas programmil on lüliteid, lugege vastava programmi `man`-lehekülge või üritage anda programmile lüliti `--help` (s.t. käsk `programm --help`).

nimeviidad
Spetsiaalsed failid, mis sisaldavad ainult teisele failile viitavat stringi. Nende failide kasutamine võrdub stringiga viidatava faili kasutamisele (sõltumata sellest, kas viidatav fail on üldse olemas või mitte). Viidatava faili asukoht võib olla antud nii suhtelise kui absoluutse otsinguteena.

sihtmärk
Kompileerimise eesmärk, s.t. kompilaatori loodav binaarfail.

TCP
Transmission Control Protocol ehk edastusohje protokoll. See on üldlevinud ja väga usaldusväärne protokoll, mis kasutab võrgupakettide edstamiseks IP-d. TCP lisab IP-le teatud kontrollmehhanismi, mis tagab, et paketid kohale toimetatakse. Erinevalt UDP-st töötab TCP ühendusega režiimis, mis tähendab, et kaks masinat peavad olema loonud ühenduse, enne kui saavad hakata andmeid vahetama.

telnet
Loob ühenduse võrgumasinaga ja võimaldab Teil sinna sisse logida, kui teil muidugi on seal konto. Telnet on kõige levinum kaugsisselogimise viis, kuigi sellele on ka paremaid ja turvalisemaid alternatiive, näiteks `ssh`.

teemavalmidus
Graafiline rakendus on teemavalmidusega, kui selle välimust on võimalik muuta reaalajas. Enamik aknahaldureid on teemavalmidusega.

TLDP
The Linux Documentation Project. Mittetulundusühing, mis hooldab GNU/Linuxi dokumentatsiooni. Selle tuntuimad dokumendid on HOWTO-d, kuid see hooldab ka FAQ-e ja isegi mõningaid raamatuid.
Vt. ka: FAQ.

läbimine
UNIX® süsteemi kataloogi korral tähendab see kasutaja õigust siseneda antud kataloogi ja võib-olla ka selle alamkataloogidesse. See tähendab, et kasutajal peab olema antud kataloogis käivitamisõigus.

URL
Uniform Resource Locator ehk üldine ressursiaadress, lihtsamalt öeldes internetiaadress. Spetsiaalse vorminguga string, mis identifitseerib Internetis asuva ressursi unikaalsena. Ressursiks võib olla fail, server või mõni muu element. URL-i süntaks on järgmine:
`protokoll://kasutaja:parool@server.nimi[:port]/ressursi/asukoht`.
Kui antud on ainult masina nimi ning protokolliks on `http://`, siis tõmmatakse vaikimisi fail, mida server on määratud vaikimisi näitama. Tavaliselt on selleks fail `index.html`.

kasutajanimi
See on nimi (õigemini küll sõna), mis identifitseerib süsteemis kasutaja. Igale kasutajanimele omistatakse unikaalne ja ainukordne UID (user ID ehk kasutaja ID).
Vt. ka: kasutajatunnus.

UTF-8
Unicode Transformation Format 8 ehk universaalne teisendusvorming 8. See on Unicode'i märkide 8-bitiste baitide kadudeta kodeering. UTF-8 kodeerib iga Unicode'i märgi üheks kuni neljaks baidiks, baitide arv sõltub sealjuures Unicode'i märgile omistatud täisarvulisest väärtusest. See on väga tõhus kodeering Unicode'i dokumentidele, mis enamasti kasutavad US-ASCII märke, sest tähistab kõiki märke vahemikus U+0000 kuni U+007F üheainsa baidina. UTF-8 on XML-i vaikekodeering.
Vt. ka: ISO-8859, ASCII.

muutujad
Stringid, mida kasutavad `Makefile`-failid ja mis asendatakse alati nende tegeliku väärtusega. Tavaliselt määratakse need kindlaks `Makefile`-faili alguses. Nende eesmärk on lihtsutada `Makefile`-failide ja lähtefailide puu haldust.
Ülidsemalt on aga muutujad programmeerimiskeeltes sõnad, mis viitavad teistele olemitele (arvudele, stringidele, tabelitele jne.), mis võivad programmi käivitamisel erinevatel kordadel olla erinevad.

jutukas
Käskude korral tähendab jutukas režiim seda, et käsk annab standard- või ka standardveaväljundisse teada kõik oma tegemised ja nende tulemused. Vahel võimaldavad käsud määrata kindlaks "jutukuse taseme" ehk siis valida, millise koguse või millist infot käsk teada annab.

VESA
Video Electronics Standards Association. Tööstusstandardite organisatsioon, mis tegeleb PC-arhitektuuriga. See on loonud näiteks SVGA standardi.

virtuaalne konsool
Varem nimetati neid enamasti terminalideks. GNU/Linuxi süsteemides võimaldavad virtuaalsed konsoolid kasutada ühel ekraanil või monitoril mitut sõltumatult töötavat seanssi. Vaikimisi on Teie käsutuses kuus virtuaalset konsooli, millele suundumiseks tuleb vajutada klahvikombinatsioone **Alt-F1** kuni **Alt-F6**. On ka seitsmes virtuaalne konsool **Alt-F7**, millel töötab X Window System. X'is saate tekstikonsooli suunduda klahvikombinatsioonidega **Ctrl-Alt-F1** kuni **Ctrl-Alt-F6**.
Vt. ka: konsool.

virtuaalsed töölauad
X Window Systemi korral võib aknahaldur pakkuda Teile mitme töölaua kasutamise võimalust. See on väga mugav ja lubab korraldada oma aknaid nii, et neid ei satuks kümnete kaupa üksteise peale. Seda võib võrrelda peaaegu mitme monitori olemasoluga. Vastavalt aknahaldurile võib mõnevõrra erineda viis, kuidas ühelt virtuaalselt töölaualt teisele liikuda.
Vt. ka: aknahaldur, töölaud.

WAN
Wide Area Network ehk laivõrk. See võrk sarnaneb LAN-iga, kuid ühendab arvuteid, mis ei ole omavahel ühendatud ühtede ja samade kaablitega ning võivad üksteisest isegi väga kaugel asuda.
Vt. ka: LAN.

metamärk
Märke '*' ja '?' kasutatakse metamärkidena, mis võivad tähistada ükstaspuha mida. '*' tähistab suvalist arvu märke, sealhulgas ka 0 märki. '?' tähistab täpselt üht märki. Metamärke kasutatakse sageli regulaaravaldistes.

aken
Võrkudes tähendab **aken** suurimat andmekogust, mida saaja on mis tahes ajahetkel võimeline vastu võtma.

Graafilises töökeskkonnas tähendab aken ristkülikut, mille sees töötab mingi rakendus ja mis tavaliselt koosneb tiitliribast, menüüst, olekuribast ning rakenduse tööalast.

aknahaldur
Programm, mis vastutab graafilise töökeskkonna "välimuse ja tunnetuse" eest, tegeldes akende ribade, raamide, nuppude, juurmenüüde ja mõningate kiirklahvidega. Ilma aknahaldurita ei oleks ka virtuaalseid töölaudu, võimalust akende suurust lennult muuta, neid liigutada jne.

töötsoonide vahetaja
Väike aplett, mis võimaldab liikuda virtuaalsete töölaudade vahel. Kannab ka nimetust pager (vahel tõlgitud ekslikult ka kui peiler).
Vt. ka: virtuaalsed töölauad.

Aineregister