Beowulf-toteutus kaleeri.uku.fi

Jukka Pitkänen

Report B/2000/1

UNIVERSITY OF KUOPIO

Department of Computer Science
and Applied Mathematics

P.O.Box 1627, FIN-70211 Kuopio, FINLAND

Sisältö

Luku 1

Esittely

Tässä dokumentissa käsitellään Beowulf-klusteria, joka sijaitsee Kuopion yliopiston Tietojenkäsittelytieteen ja sovelletun matematiikan laitoksella ja löytyy Internetin nimipalvelusta osoitteella *kaleeri.uku.fi*. Beowulf käsitteenä tarkoittaa joukkoa koneita, jotka yhdistävät voimansa ja toimivat siten kuin käytössä olisi vain yksi tehokas tietokone. Voimien yhdistäminen tapahtuu käytännössä rinnakkaisuuden avulla. Beowulf ei ole kuitenkaan oikea rinnakkaistietokone, siinä mielessä että klusterin koneilla olisi yhteinen muisti, vaan kommunikointi koneiden välillä tapahtuu verkon kautta. Käsitteellisesti onkin parempi puhua hajautetusta rinnakkaisuudesta.

Kaleerin kokoonpano lyhyesti:

5 kpl PC-koneita:

- 533 MHz Intel Celeron -prosessori
- 256 MT muistia

- 3Com 905c 10/100 Fast Etherlink -verkkokortti
- Ei kiintolevyä
- Noin 4000 mk / kone

Kytkin 100 Mbit/s:

- 3Com 3300 XM (noin 7000 mk)

Maailman 500 nopeinta supertietokonetta -listalla olevan hitaimman koneen nopeus on noin 44 gigaflopsia, kun taas Kaleerista saa tehoa irti parhaimmillaan noin 0,4 gigaflopsia. Tulos on supertietokoneiden luokassa vielä melko mitätön mutta kuitenkin huomattavasti suurempi kuin mitä yhdellä normaalilla koti-PC:llä saisi. Kaleeriin kytketyissä koneissa ei ole kiintolevyjä palvelinta lukuunottamatta, joten lisäkoneiden ja siten myös lisätehon asentaminen klusteriin käy todella helposti.

Kaleerin käyttöjärjestelmänä on Linux ja lisäksi siihen on tällä hetkellä asennettuna MPI-kirjasto (*Message Passing Interface*), joka tarjoaa ympäristön ja funktioita Beowulfissa toimivien rinnakkaisohjelmien tekemiseksi [HM97]. Ohjelmointikielenä voi olla C, C++ tai Fortran.

Tässä dokumentissa käydään läpi Beowulfin asentaminen koneiden, verkon ja ohjelmistojen osalta. Normaalia Beowulf-asennusta kiinnostavammaksi Kaleerin tekevät kiintolevyttömät koneet, sillä käyttöjärjestelmän lataaminen kiintolevyttömiin koneisiin verkon yli palvelimelta ei ole ihan normaali toimenpide. Dokumentin loppupuolella selvitellään Beowulfia yleisesti: mitä tarvitaan laitteistojen sekä ohjelmistojen osalta. Dokumentin lopussa on lisäksi Kaleeria koskeva ylläpito-ohje sekä liitteenä pieni esimerkki MPI-ohjelmasta.

Luku 2

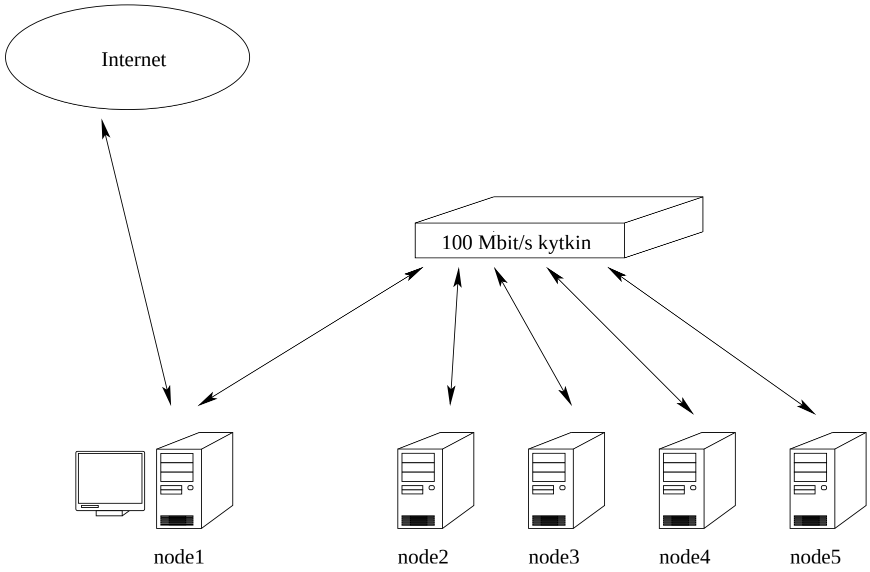
Asennusohje

2.1 Laitteisto

Beowulfin kokoaminen vastaa pienen lähiverkon rakentamista. Kaleerissa on viisi konetta ja yksi kytkin. Yksi koneista tulee olemaan palvelin, jonka kautta klusteria käytetään. Palvelinkone eroaa muista pelkästään laskentaan osallistuvista koneista siten, että siinä on kiintolevy, kaksi verkkokorttia, näyttö ja näppäimistö. Palvelinkone sijoitetaan sellaiseen kohtaan, mistä sitä on helppo käyttää, loput koneet voidaan sijoittaa esimerkiksi lattialle tai hyllyyn.

Koneiden BIOS-asetukset (*Basic Input/Output System*) vaativat hiukan säätöä. Kaikista koneista tulee asettaa "PnP-OS Installed"-asetus pois päältä. Kiintolevyttömät koneet tulee lisäksi asettaa käynnistymään levykeasemasta.

Klusteriin kytketyt koneet muodostavat oman pienen, Internetistä eristetyn, lähiverkon. Verkko rakennetaan siten, että kaikki koneet kytketään normaalilla RJ45-parikaapelilla kiinni kytkimeen. Palvelinkoneessa oleva toinen verkkokortti kytketään normaalisti Internetiin, joten palvelinkone toimii yhdyskäytävänä Internetin ja Beowulfin välissä. Kuvassa 2.1 on kaavio verkon rakenteesta.



Kuva 2.1: Beowulf-verkon rakenne

Beowulf tarvitsee nopean verkon, jotta laitteistosta saadaan kaikki teho irti. Verkkokortit ja kytkin ovat 100 Mbit/s -nopeuteen kykeneviä, mutta useasti laitteet eivät heti asennuksen jälkeen toimi maksiminopeudella vaan oletuksena on käytössä 10 Mbit/s.

Paras tapa on asettaa kaikki verkkokortit sekä myös kytkimen portit Auto-Negotiate-tilaan, jolloin kortti ja kytkin neuvottelevat keskenään parhaimman mahdollisen yhteystyypin. Tavoitteena on saada muodostettua 100 Mbit/s Full-Duplex (FD) -tyyppinen yhteys. FD tarkoittaa, että verkossa voi liikkua yhtä aikaa kumpaankin suuntaan dataa maksiminopeudella.

Verkkokortit on hyvä konfiguroida valmistajan toimittamalla yleensä DOS-pohjaisella konfigurointiohjelmalla, joka tallettaa asetukset verkkokortin EP-

ROM-piirille. Myös käyttöjärjestelmän ajureista voi yleensä säätää asetuksia, mutta niistä käsin tehdyt säädöt eivät aina toimi yhtä luotettavasti. Kytkintä voidaan konfiguroida takana olevan RS232-liittimen kautta tai verkon kautta ottamalla joko Telnet- tai WWW-yhteys kytkimen IP-osoitteeseen (*Internet Protocol*). Lopuksi kannattaa varmistaa yhteyden nopeus verkkokortin takana olevasta LED-merkkivalosta, FD-asetuksen voi tarkistaa diagnostiikkaohjelmalla.

2.2 Ohjelmisto

Laitteistoasennusten jälkeen asennetaan klusterin koneisiin käyttöjärjestelmät sekä tarvittavat ohjelmat Beowulf-toimintaa varten. Tässä dokumentissa on selitetty Linuxin asentaminen ja konfigurointi Beowulfia varten, samat periaatteet soveltuvat kuitenkin hyvin myös muihin Unix-järjestelmiin.

Tilanne on yksinkertainen, mikäli kaikissa klusterin koneissa on kiintolevyt. Tällöin tarvitsee asentaa Linux vain yhteen koneeseen ja kloonata se kaikkiin muihin. Ainoastaan palvelimen konfiguraatio eroaa joiltakin osin pelkästään laskentaan osallistuvien koneiden konfiguraatioista. Kaleerin koneissa, palvelinta lukuunottamatta, ei ole kuitenkaan kiintolevyjä, joten asennus vaatii normaalia enemmän virittämistä. Kaleerissa pelkästään palvelimeen asennetaan Linux ja loput koneista lataavat käyttöjärjestelmänsä verkon yli palvelimelta.

Prosessi etenee seuraavasti:

- Tehdään ydin, jossa mukana *ROOT_NFS*-optio sekä ajurit verkkokortille

- Koneet lataavat tämän ytimen käynnistyslevykkeeltä
- Koneet lähettävät verkkoon DHCP-kyselyn saadakseen itselleen IP-osoitteen
- Palvelimessa oleva DHCP-palvelu vastaa kyselyyn
- Koneet asettavat itselleen juurihakemistoksi palvelimelta NFS:lla jaetun hakemiston: */tftpboot/<oma_ip>/*
- Koneet lataavat käyttöjärjestelmän normaalisti juurihakemistostaan

Prosessia varten palvelimessa tulee siis olla ytimen lähdekoodi, DHCP-palvelu, NFS-palvelu, Portmapper NFS:ää varten, sekä tftpboot-hakemistorakenne, jonka alla on jokaiselle koneelle oma juurihakemisto koneen IP-osoitetta vastaavassa alihakemistossa. Seuraavissa kappaleissa käydään läpi tarkemmin eri osiot.

2.2.1 Käynnistyslevyke

Palvelinta lukuunottamatta kaikki koneet käynnistetään käynnistyslevykkeellä. Käynnistyslevyke on mahdollisimman yksinkertainen ja ei sisällä mitään muuta kuin pelkän Linux-ytimen. Ytimeksi ei käy mikä tahansa vaan se täytyy itse konfiguroida ja kääntää. Ytimestä on samalla mahdollista tehdä mahdollisimman kevyt poistamalla siitä kaikki turhat ajurit ja ominaisuudet. Tärkeintä ytimeen on kuitenkin sisällyttää ajurit koneissa käytetyille verkkokorteille, tuki NFS-tiedostojärjestelmälle sekä määrittää käyttöön "Root file system on NFS"-optio. Ydin kannattaa kääntää samasta lähdekoodin versiosta, josta myös palvelimen valmis tai itse käännetty ydin on tehty.

Liitteessä A on Kaleerin käynnistyslevykkeen tekemisessä käytetyn ytimen konfigurointitiedosto. Ytimen kääntämisen jälkeen ydin kopioidaan semmoisenaan levykkeelle esimerkiksi komennolla:

```
dd if=/usr/src/linux/arch/i386/boot/zImage of=/dev/fd0
```

Linux käynnistyy siten, että ytimen lataamisen jälkeen ydin tarvitsee heti tiedon missä juurihakemisto sijaitsee. Tieto NFS-juuren käytöstä täytyy vielä määrittää levykkeelle:

```
mknod /dev/boot255 c 0 255
rdev /dev/fd0 /dev/boot255
```

Lisätietoa NFS-juuren käyttämisestä löytyy ytimen mukana tulevasta dokumentaatiosta (*Documentation/nfsroot.txt*).

2.2.2 Dynamic Host Configuration Protocol (DHCP)

Beowulf-verkko on aivan normaali TCP/IP-protokollaa (*Transmission Control Protocol / Internet Protocol*) käyttävä verkko ja jokainen verkkoon liitetty kone tarvitsee yksilöllisen IP-osoitteen, mutta kiintolevyttömiin koneisiin ei voi tallettaa tietoa niiden IP-osoitteesta. Koneet käynnistetään käynnistyslevykkeellä, jolle IP:n voisi tallettaa, mutta silloin jokaiselle koneelle pitäisi olla oma räätälöity käynnistyslevyke. Lisäksi levykkeestä tulisi normaalia, pelkän Linux-ytimen sisältävää, käynnistyslevykettä monimutkaisempi. Helpoin tapa on asentaa palvelinkoneeseen DHCP-palvelu, joka jakaa automaattisesti muille koneille IP-osoitteet niiden käynnistyessä.

DHCP-palveluja on kahdenlaisia: dynaamisia ja staattisia. Beowulf-verkkoon soveltuu paremmin staattinen järjestelmä, missä DHCP-palvelin antaa jokaiselle koneelle aina saman yksilöllisen osoitteen. Tätä varten DHCP-palvelulle täytyy kertoa jokaisen koneen verkkokortin yksilöllinen MAC-osoite (*Media Access Control*), johon jaettava IP-osoite kiinnitetään. Myös kytkimen MAC-osoite kannattaa lisätä DHCP-palveluun, jolloin sekin saa oman IP-osoitteen käynnistyessään ja kytkintä voidaan tämän jälkeen säätää ottamalla joko Telnet- tai WWW-yhteys suoraan kytkimen IP-osoitteeseen.

Beowulf-verkon koneet eivät tarvitse Internetin nimipalvelussa olevia IP-osoitteita, vaan koneille riittää hyvin lähiverkkokäyttöä varten varatut IP-avaruudet, esimerkiksi 10.x.x.x tai 192.168.x.x. Tietoturvan ja ylläpidon kannaltakin onkin parempi käyttää Beowulf-verkon sisällä lähiverkolle varattuja IP-osoitteita, joita ei reititetä Internetissä.

Asennettaessa DHCP-palvelua koneeseen, joka on kiinni toisella verkkokortilla myös julkisessa Internetissä, täytyy olla huolellinen, ettei DHCP-palvelu sotke toista verkkoa. DHCP-palvelun käynnistyskomennossa täytyy määrittää Beowulf-verkkoon yhteydessä oleva verkkokortti, johon tuleviin DHCP-kyselyihin vain vastataan. Kaleerissa Beowulfiin kytkeytyvän verkkokortin rajapinta on *eth1* ja sen määritys DHCP-palvelulle on tiedostossa */etc/rc.d/init.d/dhcpd*:

```
...
# See how we were called.
case "$1" in
  start)
        # Start daemons.
        echo -n "Starting dhcpd: "
        daemon /usr/sbin/dhcpd eth1
        RETVAL=$?
        echo
        [ $RETVAL -eq 0 ] && touch /var/lock/subsys/dhcpd
        ;;
...
```

Liitteessä B on esimerkki Kaleerissa käytetyn DHCP-palvelun konfigurointitiedostosta *dhcpd.conf*.

2.2.3 Network File System (NFS)

Palvelimeen täytyy asentaa NFS-palvelu, jotta muut koneet saavat ladattua käyttöjärjestelmänsä palvelimelta. NFS tarvitsee toimiakseen myös Portmapper-palvelun. Palvelimelta jaetaan NFS:llä joko pelkkä tftpboot-hakemistorakenne tai koko palvelimen juurihakemisto, kuten Kaleerissa on tehty. Jako määritellään tiedostoon */etc/exports*:

```
/           10.0.0.0/0(rw,no_root_squash)
/boot       10.0.0.0/0(rw,no_root_squash)
```

Ensimmäinen sarake on jaettu hakemisto ja toisessa sarakkeessa on jakoon liittyviä parametrejä. *10.0.0.0/0* tarkoittaa, että jaon saa ottaa käyttöön kuka tahansa 10.x.x.x-aliverkon osoitteesta. *Rw* tarkoittaa sekä luku- että kirjoitusoikeuksia ja *no_ root_ squash* tarkoittaa, että asiakaskoneen pääkäyttäjällä (*root*) on pääkäyttäjän oikeudet myös NFS:llä käyttöön otetuilla levyillä.

Kirjoitusoikeuksin varustettujen NFS-jakojen kanssa täytyy noudattaa erityistä varovaisuutta, ettei jakoa pääsisi ottamaan käyttöön kukaan Beowulf-verkon ulkopuolelta. Exports-tiedostossa olevien rajoitteiden lisäksi kannattaa */etc/hosts.deny*-tiedostossa kieltää kaikki palvelut kaikista osotteista ja määrittää */etc/hosts.allow*-tiedostoon oikeudet vain Beowulf-lähiverkolle sekä paikalliselle koneelle (*localhost*).

```
/etc/hosts.deny: ALL : ALL

/etc/hosts.allow: ALL : 127.0.0.1 10.0.0.
```

Portmapper-palvelu ei tarvitse mitään erityistä konfigurointia, riittää kunhan palvelu on käynnistetty normaalisti.

2.2.4 TFTPBOOT-hakemistorakenne

Palvelimen juurihakemistoon tehdään tftpboot-hakemisto kiintolevyttömiä koneita varten. Jokaista konetta kohden tehdään vielä tftpboot-hakemiston alle oma alihakemisto ja alihakemistolle annetaan nimeksi koneen IP-osoite, koska koneet käynnistyessään saatuaan IP-osoitteensa DHCP-palvelulta asettavat juurihakemistokseen NFS-osoitteen:

<palvelimen_ip>:/tftpboot/<oma_ip>/

Palvelimen IP-osoitteen koneet saavat DHCP-vastauksen mukana.

Koska jokaiselle koneelle tehdään oma juurihakemisto, laitetaan sen IP-osoitetta vastaavaan hakemistoon koko käyttöjärjestelmän hakemistorakenne. Jos klusterissa on koneita esimerkiksi 10 kappaletta ja jokaisessa hakemistossa oleva Linux veisi noin 500 megatavua, tarvittaisiin 10 konetta varten jo 5 gigaa kiintolevytilaa. Toisaalta samojen tiedostojen kopioiminen 10 eri hakemistoon ei tunnu järkevältä ja tarvittavien muutosten tekeminen jälkikäteenkin on hankalaa, kun pitäisi päivittää tiedot kaikkiin hakemistoihin.

Yksi vaihtoehto olisi tehdä kaikkiin hakemistoihin pelkästään linkit (*hard link*) osoittamaan palvelimen hakemistorakenteen vastaaviin tiedostoihin. Linkkejä tulisi todella paljon ja järjestelmä olisi sekava ylläpitää. Lisäksi /etc-hakemistossa on aina konekohtaisia tiedostoja, joten niitä varten täytyisi vielä tehdä erilainen järjestely. Tietojen kopioiminen tai linkitys kaikkiin tftpboot-alihakemistoihin ei ole tarpeellista, mikäli koneet konfiguroidaan siten, että ne käyttävät suoraan palvelimen hakemistorakennetta. Tämän takia Kaleerissa on NFS:llä jaettu koko hakemistorakenne juuresta lähtien (luku 2.2.3). Koneet siis käynnistyessään heti juurihakemiston asetuksen jälkeen, yhdistävät *mount* sen päälle palvelimelta yksitellen */bin*, */boot*, */home*, */lib*, */sbin*, ym. hakemistot.

Jokaista konetta kohden palvelimen */tftpboot/<oma_ip>/*-alihakemistoihin kuitenkin tarvitsee kopioida ainakin *init*- ja *mount*-käsky sekä niiden tarvitsemat kirjastot, joita ilman palvelimen hakemistojen yhdistäminen oman juurihakemiston päälle ei onnistu.

Koneet eivät yhdistä palvelimelta /etc-hakemistoa, koska siellä on aina muutamia konekohtaisia tiedostoja. Jokaisella koneella on sen sijaan oma oikea etc-hakemisto tftpboot-hakemistorakenteen alla. Näiden etc-hakemistojen kaikki muut, paitsi konekohtaiset tiedostot, ovat kuitenkin linkkejä palvelimella sijaitsevaan malli-etc-hakemistoon. Näin myös etc-hakemistossa olevien tiedostojen päivitys on helppoa, koska tarvitsee vain muuttaa malli-hakemistossa olevia tiedostoja ja linkkien ansiosta muutokset tulevat heti voimaan kaikissa klusterin koneissa.

Liitteissä C ja D on kaksi skriptiä (*addserver.sh* ja *addnode.sh*), joiden avulla voidaan luoda ja ylläpitää tftpboot-hakemistorakennetta.

2.2.5 Beowulf-ohjelmisto

Beowulf-toimintaa varten täytyy klusterin koneisiin asentaa jokin viestinvälitysprotokollan toteuttava kirjasto. Eräs toteutus MPI-kirjastosta Linuxille on MPICH [Tea99]. Se on ilmainen ja MPI-standardin [For95] mukainen. Asennus käy helposti asentamalla se vain palvelimeen, jonka jälkeen se löytyy kaikista koneista. MPI-ohjelmien käynnistämistä varten tarvitaan klusteriin vielä *rsh*-ohjelma viritettynä siten, ettei se kysy salasanaa avattaessa RSH-yhteys (*Remote Shell*) mihin tahansa klusterin koneeseen. Tätä varten luodaan */etc/ hosts.equiv*-tiedosto ja kirjoitetaan sinne jokaisen klusterin koneen IP-osoite omalle rivilleen. Mikäli klusterissa ei käytetä nimipalvelua, voidaan koneille antaa selväkieliset nimet */etc/hosts*-tiedostoon. MPI-kirjastoa varten täytyy vielä tehdä */usr/share/machines.LINUX*-tiedosto, johon myös tulee klusterin koneiden osoitteet, yksi rivilleen. Tässä tiedostossa luetellaan ne koneet, joita MPI-ohjelmat käyttävät ohjelman suorituksessa. Luvussa 2.2.4 esitellyt skriptit päivittävät automaattisesti näitä tiedostoja.

Liitteessä E on esimerkki */etc/hosts-*, */etc/hosts.equiv-* ja */usr/share/machines.LINUX*-tiedostosta.

Luku 3

Tekniset ja toiminnalliset vaatimukset

3.1 Mitä beowulf on?

Ideaalisessa rinnakkaistietokoneessa on monta prosessoria ja yksi yhteinen muistiavaruus, jota kaikki prosessorit käyttävät. Rinnakkaistietokoneet ovat kuitenkin hyvin kalliita ja tuleekin paljon halvemmaksi hankkia monta erillistä tavallista tietokonetta. Erillisissä tietokoneissa haittapuolena on kuitenkin, että jokaisella prosessorilla on oma muisti, kun taas rinnakkaisohjelmien ohjelmoinnin ja suorituksen kannalta olisi parempi olla yhteinen muisti. Yhteisen muistin sijasta Beowulfin koneet ovat yhteydessä toisiinsa nopealla verkolla. Verkon kautta koneet voivat lähettää nopeasti toisilleen viestejä, jolloin voidaan tehdä ohjelmia, joita suoritetaan rinnakkaisesti kaikissa verkon koneissa. Näin ollen Beowulfissa suoritettu laskenta ei ole oikeata rinnakkaislaskentaa vaan hajautettua rinnakkaislaskentaa. Pelkkä hajautettu laskenta ei tarvitse nopeaa verkkoyhteyttä toimiakseen, mutta yhteyden kuitenkin.

Kunnolla toimiakseen Beowulf tarvitsee paljon koneita, joissa on nopeat prosessorit. Tämän lisäksi yhtä tärkeä on nopea verkkoyhteys, vähintään 100 Mbit/s ja Full-Duplex-tyyppisenä. Koneiden verkkoon kytkemiseksi tarvitaan lisäksi samaan nopeuteen kykenevä kytkin. Keskitin ei sovellu Beowulf-käyttöön, koska se hukkaa liikaa kaistanleveyttä kaiuttaessaan kaiken verkkoliikenteen kaikkiin keskittimeen kytkettyihin laitteisiin. Viime aikoina myös Gbit/s-nopeuksiin kykenevät verkkolaitteet ovat alkaneet yleistyä. Yleensä juuri kaistanleveydellä on suorituskykyyn paljon enemmän merkitystä kuin verkon latenssilla.

Mikä tahansa ohjelma ei toimi rinnakkaisesti Beowulfissa, vaan ohjelma täytyy alunperin ohjelmoida Beowulf-ympäristössä toimivaksi. Käytännössä tämä tarkoittaa sitä, että ohjelmoinnissa käytetään jotakin viestinvälityskirjastoa. Viestinvälityskirjasto sisältää funktioita ja ympäristön, joiden avulla on mahdollista tehdä ohjelmia, joita suoritetaan rinnakkaisesti kaikissa Beowulfin koneissa yhtä aikaa. Rinnakkaisohjelmointi käyttäen MPI:tä tai jotakin vastaavaa kirjastoa, on yleensä hankalaa ja työlästä.

Oikea rinnakkaistietokone on kallis mutta helppo ohjelmoida, kun taas Beowulf on halpa mutta vaikea ohjelmoida. Liitteissä F ja G on kaksi yksinkertaista esimerkkiä MPI-ohjelmasta.

3.2 Mitä Linuxin osia tarvitaan?

Kaleerissa on käyttöjärjestelmänä Linux, mutta Beowulf ei ole kuitenkaan käyttöjärjestelmäriippuvainen. Linuxia koskevat ohjeet ja periaatteet ovat helposti sovellettavissa myös muihin Unix-järjestelmiin.

Koska Beowulf on pelkistettynä lähiverkko, tarvitaan jokaisen koneen perustaksi verkko-ominaisuuksin varustettu Linux-asennus. Jotta koneita voisi käyttää rinnakkaisesti, tarvitaan sitä varten jokin viestinvälityskirjasto, esimerkiksi edellä mainittu MPI. MPI-kirjaston tulee löytyä verkon kaikista koneista.

MPI itse vaatii toimiakseen paljon muita Linuxin osia [Tea99]. MPI:tä käyttävät rinnakkaisohjelmat toimivat siten, että kun palvelimessa käynnistetään jokin MPI-ohjelma, se sama ohjelma käynnistyy automaattisesti kaikissa Beowulfin koneissa. Ohjelman suoritettava versio täytyy siis löytyä kaikista koneista vieläpä samasta paikkaa hakemistorakennetta, yleensä tämä tarkoittaa käyttäjän kotihakemistoa. Yksinkertaisin vaihtoehto on, että käyttäjä ennen MPI-ohjelman käynnistämistä kopioi itse ohjelman kaikkien koneiden kotihakemistoihinsa. Helpointa on kuitenkin jakaa palvelimelta käyttäjien kotihakemistot Beowulf-verkkoon käyttäen jotakin hajautettua tiedostojärjestelmää, jolloin kaikissa koneissa näkyy samat käyttäjien kotihakemistot.

Beowulfissa käytettäväksi hajautetuksi tiedostojärjestelmäksi soveltuu hyvin NFS. Se on yksinkertainen ja NFS:stä puuttuva tietoturvakaan ei haittaa Beowulf-käytössä, koska Beowulf-verkkojen sisäpuolella ei tarvita tietoturvaa ja siitä on enemmänkin vain haittaa. Kaleerissa laskentaan osallistuvissa koneissa ei ole kiintolevyjä ollenkaan, joten NFS:ää tarvitaan Kaleerissa myös niiden koneiden käyttöjärjestelmän lataamiseen verkon yli palvelimelta.

MPI-ohjelma käynnistyessään käynnistyy automaattisesti kaikissa verkon koneissa. Tämä on toteutettu käyttäen Linuxin *rsh*-komentoa, jonka avulla pystyy etäkäynnistämään toisessa koneessa olevia ohjelmia. Beowulfissa ne ohjelmat siis löytyvät jokaisesta koneesta NFS:llä jaetusta käyttäjän kotihakemistosta. RSH tulee virittää Beowulfissa siten, ettei se kysy salasanaa missään vaiheessa yhteydenmuodostusta.

Beowulf on sisältäpäin hyvin tietoturvaton, mutta koko verkko on kuitenkin ulkomaailta suojassa eikä siihen saa yhteyttä muuten kuin sisäänkirjautumalla ensin palvelimeen. Palvelimen tietoturvaan tuleekin kiinnittää normaalia enemmän huomiota. Käytettävyyden takia palvelimeen täytyy avata muutamia palveluita. Tärkein palveluista on *sshd*, jonka avulla käyttäjät pystyvät avaamaan pääteyhteyden palvelimeen Internetistä käsin ja pääsevät siten käyttämään Beowulfia. Ohjelmien ja lähdekoodien siirtoa varten palvelimeen on hyvä asentaa myös FTP-palvelu (*File Transfer Protocol*). Sähköpostin vastaanotto on yleensä turhaa Beowulfissa, mutta lähettämisen tulisi kuitenkin onnistua. Esimerkiksi ajastuksessa käytettävä *crontab*-ohjelma lähettää käyttäjälle sähköpostina ajastettavien ohjelmien tulosteita sekä mahdollisia virheilmoituksia. Tulostusta varten palvelimeen voi vielä lisätä LPR-palvelun.

Beowulf-verkon palvelimessa olevat palvelut tulee suojata huolella. Varsinkin sisäverkkoon päin oleva kirjoitusoikeuksin varustettu NFS ja RSH:n ilman salasanaa toimivat yhteydet eivät missään nimessä saisi olla auki Internetiin. Palveluita suojataan rajoittamalla osoitteita, joista palveluihin saa ottaa yhteyttä. Periaate on, että Beowulfin tarvitsemiin palveluihin saisi ottaa yhteyden vain verkon sisäpuolelta Beowulfissa käytetyistä lähiverkolle varatuista IP-osoitteista. Ulkopuolelta tulevat SSH- (*Secure Shell*) ja FTP-yhteydenottopyynnötkin kannattaa rajoittaa vain niihin osoitteisiin, joista todelliset käyttäjät tulevat.

Suojauksesta voi tehdä monikerroksisen rajoittamalla ensin osoitteita ohjelmasta itsestään, lisäksi Linuxissa on helppo tehdä rajoituksia *inetd*-palvelusta käynnistettäviin toisiin palveluihin konfiguroimalla */etc/hosts.deny* ja */etc/hosts.allow* tiedostoja. Matalimman tason suojauksen saa vielä edellisten lisäksi käyttämällä osittain Linuxin ytimeen sisäänrakennettua palomuuria *ipchains*-ohjelman avulla. Palomuurilla voi helposti esimerkiksi sulkea kaikki TCP- (*Transmission Control Protocol*) ja UDP-portit (*User Datagram Protocol*) ja avata vain ne, joita tarvitaan. Liitteessä H on esimerkki palomuuriskriptistä.

3.3 MPI-ohjelmointi

MPI-ohjelma eroaa tavallisesta ohjelmasta siten, että se käynnistyy automaattisesti kaikissa klusterin koneissa ja sen ohjelmoinnissa on käytetty MPI-kirjaston palveluita. MPI-ohjelmoinnissa ohjelmointikielenä voi olla C, C++ tai Fortran.

C-kielinen MPI-ohjelma käännetään *mpicc*-komennolla, esimerkiksi:

```
mpicc -o hello hello.c
```

Komento esikäsittelee käännöskutsua ja lopuksi kutsuu oikeata *gcc*-kääntäjää, joten kaikki *gcc*:n tunnistamat parametrit ovat käytettävissä. C++ -kielinen ohjelma käännetään komennolla *mpiCC* ja Fortran-kielinen komennolla *mpif77*.

Valmis MPI-ohjelma suoritetaan Beowulfissa *mpirun*-komennolla, esimerkiksi:

```
mpirun -np 5 ./hello
```

Komennon suorittamishetkellä kannattaa olla hakemistossa, johon itsellä on kirjoitusoikeudet, sillä komento luo siihen hakemistoon väliaikaisen tiedoston. Komennossa olevalla *-np <luku>* -parametrillä valitaan kuinka monta prosessoria suorituksessa käytetään. Optimaalisinta on käyttää niin monta kuin klusterissa on prosessoreita. Mikäli luku on suurempi kuin prosessorien lukumäärä, suorittavat prosessorit useampia versioita ohjelmasta yhtä aikaa. Käytettävissa olevat prosessorit on listattu */usr/share/machines.LINUX*-

tiedostossa. Komennon viimeisenä parametrinä on suoritettavan MPI-ohjelman nimi. Lisää tietoa *mpirun*-komennon käytöstä saa komennolla *mpirun -h*.

MPI-kirjasto sisältää yli sata funktiota, mutta jo kuudella perusfunktiolla voi tehdä paljon [Cen98]:

Viestinvälityksen aloitukseen käytettävät funktiot:

- **MPI_INIT**: alustaa MPI-ympäristön
- **MPI_COMM_SIZE**: palauttaa prosessien lukumäärän
- **MPI_COMM_RANK**: palauttaa kyseisen prosessin numeron (*rank*)

Viestien lähettämiseen ja vastaanottamiseen käytettävät funktiot:

- **MPI_SEND**: lähettää viestin
- **MPI_RECV**: vastaanottaa viestin

Viestinvälityksen lopetus:

- **MPI_FINALIZE**: Lopettaa MPI-ympäristön

Hyvä MPI-dokumentaatio löytyy Kaleerista WWW-muodossa avamaalla selaimeen tiedoston:

file:/usr/doc/mpich-1.2.0/www/index.html

Kyseinen dokumentaatio on referenssiluettelo, jossa on kuvaukset kaikista MPI:n käyttämistä komennoista ja funktioista sekä niiden parametreistä ja palautusarvoista.

Liitteissä F ja G on kaksi esimerkkiä yksinkertaisesta C-kielisestä MPI-ohjelmasta. Ohjelmien lähdekoodit ja suoritettavat versiot löytyvät Kaleerin hakemistosta */home/public/examples/*. Lisäksi MPI-asennuksen mukana tulleet esimerkit löytyvät Kaleerin hakemistosta */usr/share/mpi/examples/*.

Luku 4

Ylläpito-ohje

Kaleerin ylläpitoon liittyvät tiedostot sijaitsevat pääkäyttäjän kotihakemiston alaisuudessa. */root/kernel*-hakemistossa on käynnistyslevykkeellä käytettävän ytimen konfigurointitiedosto sekä valmis käännetty ydin. */root/doc*-hakemistossa on Beowulfiin liittyvia dokumentteja ja */root/softat*-hakemistossa on sekalaisia Beowulfiin liittyviä ohjelmia.

Tärkeimmät jatkuvaan ylläpitoon liittyvät tiedostot sijaitsevat */root/bin*-hakemistossa. Siellä on luvussa 2.2.4 esitellyn tftpboot-hakemistorakenteen hallinnointiin tarvittavat tiedostot. *Addserver.sh*-skripti ajetaan vain kerran puhtaassa Linux-asennuksessa. Parametrinä sille annetaan IP-osoite, joka annetaan palvelimelle Beowulf-verkkoon. *Addnode.sh*-skripti ajetaan aina uuden koneen lisäyksen jälkeen. Parametrinä skriptille annetaan IP-osoite, joka halutaan uudelle koneelle antaa. *Addnode.sh*-skripti päivittää automaattisesti */etc/hosts*-, */etc/hosts.equiv*- ja */usr/share/machines.LINUX*-tiedostot, mutta sen lisäksi täytyy käsin konfiguroida uuden koneen verkkokortin MAC-osoite DHCP-palveluun muokkaamalla */etc/dhcpd.conf*-tiedostoa.

/root/bin-hakemistossa on vielä *updatelinks.sh*-skripti (liite I), joka ajetaan aina uuden käyttäjätunnuksen luonnin jälkeen. Skripti on hyvä ajaa myös mikäli käyttäjätunnus- tai ryhmätunnustietoja muutetaan. Skripti luo uudelleen tftpboot-hakemistorakenteen alla olevia linkkejä, sillä linkit tuhoutuvat aina esimerkiksi ajettaessa *passwd*-ohjelma, koska se päivittää tiedostoja siten, että tuhoaa ensin alkuperäisen tiedoston ja luo sitten kokonaan uuden. Tuhottaessa tiedosto tuhoutuu samalla myös siihen osoittaneet linkit.

Liite A

Ytimen konfigurointitiedosto

```
#
# Automatically generated make config: don't edit
#

#
# Code maturity level options
#
# CONFIG_EXPERIMENTAL is not set

#
# Processor type and features
#
# CONFIG_M386 is not set
# CONFIG_M486 is not set
# CONFIG_M586 is not set
# CONFIG_M586TSC is not set
CONFIG_M686=y
CONFIG_X86_WP_WORKS_OK=y
CONFIG_X86_INVLPG=y
CONFIG_X86_BSWAP=y
CONFIG_X86_POPAD_OK=y
CONFIG_X86_TSC=y
CONFIG_X86_GOOD_APIC=y
CONFIG_1GB=y
# CONFIG_2GB is not set
# CONFIG_MATH_EMULATION is not set
# CONFIG_MTRR is not set
# CONFIG_SMP is not set

#
# Loadable module support
#
# CONFIG_MODULES is not set

#
```

```
# General setup
#
# CONFIG_BIGMEM is not set
CONFIG_NET=y
CONFIG_PCI=y
# CONFIG_PCI_GOBIOS is not set
# CONFIG_PCI_GODIRECT is not set
CONFIG_PCI_GOANY=y
CONFIG_PCI_BIOS=y
CONFIG_PCI_DIRECT=y
CONFIG_PCI_QUIRKS=y
CONFIG_PCI_OLD_PROC=y
# CONFIG_MCA is not set
# CONFIG_VISWS is not set
CONFIG_SYSVIPC=y
CONFIG_BSD_PROCESS_ACCT=y
CONFIG_SYSCTL=y
# CONFIG_BINFMT_AOUT is not set
CONFIG_BINFMT_ELF=y
# CONFIG_BINFMT_MISC is not set
# CONFIG_PARPORT is not set
CONFIG_APM=y
# CONFIG_APM_IGNORE_USER_SUSPEND is not set
# CONFIG_APM_DO_ENABLE is not set
# CONFIG_APM_CPU_IDLE is not set
# CONFIG_APM_DISPLAY_BLANK is not set
# CONFIG_APM_IGNORE_SUSPEND_BOUNCE is not set
# CONFIG_APM_RTC_IS_GMT is not set
# CONFIG_APM_ALLOW_INTS is not set
# CONFIG_APM_REAL_MODE_POWER_OFF is not set

#
# Plug and Play support
#
# CONFIG_PNP is not set

#
# Block devices
#
CONFIG_BLK_DEV_FD=y
CONFIG_BLK_DEV_IDE=y

#
# Please see Documentation/ide.txt for help/info on IDE drives
#
# CONFIG_BLK_DEV_HD_IDE is not set
CONFIG_BLK_DEV_IDEDISK=y
# CONFIG_BLK_DEV_IDECD is not set
# CONFIG_BLK_DEV_IDETAPE is not set
# CONFIG_BLK_DEV_IDEFLOPPY is not set
# CONFIG_BLK_DEV_IDESCSI is not set
# CONFIG_BLK_DEV_CMD640 is not set
# CONFIG_BLK_DEV_RZ1000 is not set
# CONFIG_BLK_DEV_IDEPCI is not set
# CONFIG_IDE_CHIPSETS is not set

#
# Additional Block Devices
#
# CONFIG_BLK_DEV_LOOP is not set
# CONFIG_BLK_DEV_NBD is not set
# CONFIG_BLK_DEV_MD is not set
```

```
# CONFIG_BLK_DEV_RAM is not set
# CONFIG_BLK_DEV_XD is not set
# CONFIG_BLK_DEV_DAC960 is not set
CONFIG_PARIDE_PARPORT=y
# CONFIG_PARIDE is not set
# CONFIG_BLK_CPQ_DA is not set
# CONFIG_BLK_DEV_HD is not set

#
# Networking options
#
CONFIG_PACKET=y
CONFIG_NETLINK=y
CONFIG_RTNETLINK=y
CONFIG_NETLINK_DEV=y
CONFIG_FIREWALL=y
CONFIG_FILTER=y
CONFIG_UNIX=y
CONFIG_INET=y
# CONFIG_IP_MULTICAST is not set
# CONFIG_IP_ADVANCED_ROUTER is not set
CONFIG_IP_PNP=y
# CONFIG_IP_PNP_DHCP is not set
CONFIG_IP_PNP_BOOTP=y
# CONFIG_IP_PNP_RARP is not set
CONFIG_IP_FIREWALL=y
CONFIG_IP_FIREWALL_NETLINK=y
CONFIG_NETLINK_DEV=y
# CONFIG_IP_TRANSPARENT_PROXY is not set
# CONFIG_IP_MASQUERADE is not set
# CONFIG_IP_ROUTER is not set
# CONFIG_NET_IPIP is not set
# CONFIG_NET_IPGRE is not set
# CONFIG_IP_ALIAS is not set
CONFIG_SYN_COOKIES=y

#
# (it is safe to leave these untouched)
#
# CONFIG_INET_RARP is not set
CONFIG_SKB_LARGE=y

#
#
#
# CONFIG_IPX is not set
# CONFIG_ATALK is not set

#
# Telephony Support
#
# CONFIG_PHONE is not set

#
# SCSI support
#
# CONFIG_SCSI is not set

#
# I2O device support
#
# CONFIG_I2O is not set
```

```
#
# Network device support
#
CONFIG_NETDEVICES=y

#
# ARCnet devices
#
# CONFIG_ARCNET is not set
# CONFIG_DUMMY is not set
# CONFIG_BONDING is not set
# CONFIG_EQUALIZER is not set
# CONFIG_NET_SB1000 is not set

#
# Ethernet (10 or 100Mbit)
#
CONFIG_NET_ETHERNET=y
CONFIG_NET_VENDOR_3COM=y
# CONFIG_EL1 is not set
# CONFIG_EL2 is not set
# CONFIG_ELPLUS is not set
# CONFIG_EL3 is not set
# CONFIG_3C515 is not set
CONFIG_BC90X=y
CONFIG_VORTEX=y
# CONFIG_LANCE is not set
# CONFIG_NET_VENDOR_SMC is not set
# CONFIG_NET_VENDOR_RACAL is not set
# CONFIG_NET_ISA is not set
CONFIG_NET_EISA=y
# CONFIG_PCNET32 is not set
# CONFIG_APRICOT is not set
# CONFIG_CS89x0 is not set
CONFIG_DE4X5=y
CONFIG_DEC_ELCP=y
# CONFIG_DGRS is not set
CONFIG_EEXPRESS_PRO100=y
CONFIG_NE2K_PCI=y
# CONFIG_TLAN is not set
# CONFIG_VIA_RHINE is not set
# CONFIG_SIS900 is not set
# CONFIG_NET_POCKET is not set

#
# Ethernet (1000 Mbit)
#
# CONFIG_SK98LIN is not set
# CONFIG_FDDI is not set
# CONFIG_PPP is not set
# CONFIG_SLIP is not set
# CONFIG_NET_RADIO is not set

#
# Token ring devices
#
# CONFIG_TR is not set
# CONFIG_NET_FC is not set

#
# Wan interfaces
```

```
#
# CONFIG_HOSTESS_SV11 is not set
# CONFIG_COSA is not set
# CONFIG_SEALEVEL_4021 is not set
# CONFIG_SYNCLINK_SYNCPPP is not set
# CONFIG_LANMEDIA is not set
# CONFIG_COMX is not set
# CONFIG_DLCI is not set
# CONFIG_WAN_DRIVERS is not set
# CONFIG_SBNI is not set

#
# Amateur Radio support
#
# CONFIG_HAMRADIO is not set

#
# IrDA (infrared) support
#
# CONFIG_IRDA is not set

#
# ISDN subsystem
#
# CONFIG_ISDN is not set

#
# Old CD-ROM drivers (not SCSI, not IDE)
#
# CONFIG_CD_NO_IDESCSI is not set

#
# Character devices
#
CONFIG_VT=y
CONFIG_VT_CONSOLE=y
CONFIG_SERIAL=y
CONFIG_SERIAL_CONSOLE=y
# CONFIG_SERIAL_EXTENDED is not set
# CONFIG_SERIAL_NONSTANDARD is not set
CONFIG_UNIX98_PTYS=y
CONFIG_UNIX98_PTY_COUNT=256
# CONFIG_MOUSE is not set

#
# Joysticks
#
# CONFIG_JOYSTICK is not set
# CONFIG_QIC02_TAPE is not set
# CONFIG_WATCHDOG is not set
# CONFIG_NVRAM is not set
CONFIG_RTC=y

#
# Video For Linux
#
# CONFIG_VIDEO_DEV is not set
# CONFIG_DTLK is not set

#
# Ftape, the floppy tape device driver
#
```

```
# CONFIG_FTAPE is not set

#
# Filesystems
#
# CONFIG_QUOTA is not set
# CONFIG_AUTOFS_FS is not set
# CONFIG_AFFS_FS is not set
# CONFIG_HFS_FS is not set
# CONFIG_FAT_FS is not set
# CONFIG_ISO9660_FS is not set
# CONFIG_JOLIET is not set
# CONFIG_MINIX_FS is not set
# CONFIG_NTFS_FS is not set
# CONFIG_HPFS_FS is not set
CONFIG_PROC_FS=y
CONFIG_DEVPTS_FS=y
# CONFIG_ROMFS_FS is not set
# CONFIG_EXT2_FS is not set
# CONFIG_SYSV_FS is not set
# CONFIG_UFS_FS is not set

#
# Network File Systems
#
# CONFIG_CODA_FS is not set
CONFIG_NFS_FS=y
CONFIG_ROOT_NFS=y
CONFIG_SUNRPC=y
CONFIG_LOCKD=y
# CONFIG_SMB_FS is not set
# CONFIG_NCP_FS is not set

#
# Partition Types
#
# CONFIG_BSD_DISKLABEL is not set
# CONFIG_MAC_PARTITION is not set
# CONFIG_SMD_DISKLABEL is not set
# CONFIG_SOLARIS_X86_PARTITION is not set
# CONFIG_NLS is not set

#
# Console drivers
#
CONFIG_VGA_CONSOLE=y
CONFIG_VIDEO_SELECT=y

#
# Sound
#
# CONFIG_SOUND is not set

#
# Kernel hacking
#
# CONFIG_MAGIC_SYSRQ is not set
```

Liite B

dhcpd.conf

```
subnet 10.0.0.0 netmask 255.255.255.0 {
option routers 10.0.0.1;
option subnet-mask 255.255.255.0;

        host switch {
           hardware ethernet 00:d0:96:f7:ca:18;
           fixed-address 10.0.0.254;
        }

        host node2 {
           hardware ethernet 00:01:02:25:eb:38;
           fixed-address 10.0.0.2;
        }
        host node3 {
           hardware ethernet 00:01:02:29:ca:b6;
           fixed-address 10.0.0.3;
        }
        host node4 {
           hardware ethernet 00:01:02:25:eb:1b;
           fixed-address 10.0.0.4;
        }
        host node5 {
           hardware ethernet 00:01:02:29:c7:f9;
           fixed-address 10.0.0.5;
        }
}
```

Liite C

addserver.sh

```
#!/bin/sh

### skripti, joka ajetaan vain kerran puhtaassa Linux-installaatiossa
### ennen koneiden lisäämistä addnode.sh -skriptillä
###
### tekee mm. /tftpboot ja templaatti-etc-hakemiston


if [ "$1" = "" ]; then
   echo "  Usage: $0 <ip-numero>   (esim 10.0.0.1)"
   exit 0
fi


export SERVER=$1
export TEMPLATE=/tftpboot/template
NODENRO=`echo $SERVER | cut -d"." -f4`

##### tee dirrit
mkdir -p $TEMPLATE/bin
mkdir -p $TEMPLATE/lib
mkdir -p $TEMPLATE/sbin
cp -af /dev $TEMPLATE
cp -af /etc $TEMPLATE
cp -af /var $TEMPLATE

##### pakolliset filet buuttaukselle, ennen kuin saadaan nfsmountit päälle
### binaryt
cp -f /bin/bash   $TEMPLATE/bin/
cp -f /bin/mount  $TEMPLATE/bin/
cp -f /sbin/init $TEMPLATE/sbin/
cd $TEMPLATE/bin
ln -sf  bash  sh

### libit
```

```
cp -f /lib/ld-2.1.3.so $TEMPLATE/lib/ld-2.1.3.so
cp -f /lib/libc-2.1.3.so $TEMPLATE/lib/libc-2.1.3.so
cp -f /lib/libtermcap.so.2.0.8 $TEMPLATE/lib/libtermcap.so.2.0.8
cd $TEMPLATE/lib
ln -sf ld-2.1.3.so ld-linux.so.2
ln -sf  libc-2.1.3.so libc.so.6
ln -sf  libtermcap.so.2.0.8 libtermcap.so.2



###### cleanup

rm -f $TEMPLATE/etc/sysconfig/network
rm -f $TEMPLATE/etc/sysconfig/network-scripts/ifcfg-eth*
rm -f $TEMPLATE/etc/fstab
rm -f $TEMPLATE/etc/mtab
rm -f $TEMPLATE/etc/exports
rm -f $TEMPLATE/var/lib/rpm/*
rm -f $TEMPLATE/etc/ntp/drift

rm -rf  $TEMPLATE/var/log/*
touch  $TEMPLATE/var/log/wtmp
chown root.utmp $TEMPLATE/var/log/wtmp
chmod 664  $TEMPLATE/var/log/wtmp



############## SERVICES for nodes

rm -f  $TEMPLATE/etc/rc.d/rc6.d/K*  # nodet buutataan brutaalisti
rm -f  $TEMPLATE/etc/rc.d/rc0.d/K*  # nodet haltataan brutaalisti
rm -f  $TEMPLATE/etc/rc.d/rc3.d/S*
rm -f  $TEMPLATE/etc/rc.d/rc6.d/S*killall
rm -f  $TEMPLATE/etc/rc.d/rc0.d/S*killall
cd $TEMPLATE/etc/rc.d/rc3.d
ln -sf ../init.d/random   S20random
ln -sf ../init.d/syslog   S30syslog
ln -sf ../init.d/inet     S50inet
ln -sf ../init.d/xntpd    S55xntpd
ln -sf ../init.d/keytable S75keytable
ln -sf ../init.d/network  S95network
ln -sf ../rc.local        S99local



############# log everything on servers syslogd (on server start with -r)

### nodejen konffifilet


rm -f $TEMPLATE/etc/syslog.conf
echo "*.* @$SERVER" > $TEMPLATE/etc/syslog.conf


cat << EOF > $TEMPLATE/etc/rc.d/init.d/halt
case "\$0" in
  *halt)
        command="halt"
        ;;
  *reboot)
        command="reboot"
```

```
        ;;
esac
eval \$command -f
EOF
chmod 755 /etc/rc.d/init.d/halt



cat << EOF > $TEMPLATE/etc/sysconfig/network
NETWORKING=yes
FORWARD_IPV4=false
EOF



## ei toimi enää 2.2.16 kernelin kanssa, JP 22.6.2000
# cat << EOF > $TEMPLATE/etc/sysconfig/network-scripts/ifcfg-eth0
# DEVICE=eth0
# BOOTPROTO=bootp
# ONBOOT=yes
# EOF


cp $TEMPLATE/etc/rc.d/rc.sysinit $TEMPLATE/etc/rc.d/rc.sysinit.2



cat << EOF > $TEMPLATE/etc/rc.d/rc.sysinit
#!/bin/sh
mount -n -o remount,rw /
mount -t nfs 10.0.0.1:/bin      /bin    -o ro,rsize=8192,wsize=8192,intr,timeo=14 2>/dev/null
#mount -t nfs 10.0.0.1:/boot /boot   -o ro,rsize=8192,wsize=8192,intr,timeo=14 2>/dev/null
mount -t nfs 10.0.0.1:/home     /home   -o rw,rsize=8192,wsize=8192,intr,timeo=14 2>/dev/null
mount -t nfs 10.0.0.1:/lib      /lib    -o ro,rsize=8192,wsize=8192,intr,timeo=14 2>/dev/null
mount -t nfs 10.0.0.1:/opt      /opt    -o ro,rsize=8192,wsize=8192,intr,timeo=14 2>/dev/null
mount -t nfs 10.0.0.1:/root     /root   -o rw,rsize=8192,wsize=8192,intr,timeo=14 2>/dev/null
mount -t nfs 10.0.0.1:/sbin     /sbin   -o ro,rsize=8192,wsize=8192,intr,timeo=14 2>/dev/null
mount -t nfs 10.0.0.1:/usr      /usr    -o ro,rsize=8192,wsize=8192,intr,timeo=14 2>/dev/null
/etc/rc.d/rc.sysinit.2

### aseta buutissa nodejen hostname /etc/hosts filen perusteella
IP=\`uname -n\`
HOSTNAME=\`egrep "\$IP |\$IP " /etc/hosts | awk '{print \$2}'\`
hostname \$HOSTNAME

EOF
chmod 755 $TEMPLATE/etc/rc.d/rc.sysinit




## foo fsck.nfs, palauttaa aina 0 = ok
cat << EOF > /sbin/fsck.nfs
#!/bin/sh
exit 0
EOF
chmod 755 /sbin/fsck.nfs
```

```
### xntpd
cat << EOF > $TEMPLATE/etc/ntp.conf
server $SERVER
server 127.127.1.0
fudge  127.127.1.0
driftfile /etc/ntp/drift
authenticate no
EOF

echo "10.0.0.1" > $TEMPLATE/etc/ntp/step-tickers



### lisää server (node) hosts, hosts.equiv ja machines.LINUX tiedostoihin

echo "$SERVER node$NODENRO `hostname | cut -d"." -f1` `hostname`" >> /etc/hosts
echo "$SERVER" >> /etc/hosts.equiv
## ei lisätä serveriä, local laskenta automaattisesti
# echo "node$NODENRO" >> /usr/share/machines.LINUX



### tulosta ohjeita käyttäjälle

echo ""
echo "Server IP: $SERVER = node$NODENRO lisätty järjestelmään"
echo ""
echo "Tarkista tiedostot:"
echo " /etc/hosts"
echo " /etc/hosts.equiv"
echo " /usr/share/machines.LINUX"
```

Liite D

addnode.sh

```
#!/bin/sh

### skripti, joka ajetaan aina uuden koneen lisäyksen jälkeen

if [ "$1" = "" ]; then
   echo "  Usage: $0 <ip-numero>   (esim 10.0.0.2)"
   exit 0
fi


export SERVER=10.0.0.1
export TEMPLATE=/tftpboot/template
export DEST_DIR=/tftpboot/$1
NODENRO=`echo $1 | cut -d"." -f4`

######## tyhjät hakemistot nfsmountille

mkdir -p $DEST_DIR/bin
mkdir -p $DEST_DIR/boot
mkdir -p $DEST_DIR/etc
mkdir -p $DEST_DIR/home
mkdir -p $DEST_DIR/lib
mkdir -p $DEST_DIR/mnt
mkdir -p $DEST_DIR/opt
mkdir -p $DEST_DIR/proc
mkdir -p $DEST_DIR/root
mkdir -p $DEST_DIR/sbin
mkdir -p $DEST_DIR/tmp
mkdir -p $DEST_DIR/usr


cp -af $TEMPLATE/dev $DEST_DIR
cp -af $TEMPLATE/var $DEST_DIR

chmod 777 $DEST_DIR/tmp
```

```
########## etc dirrin filet hardlinkataan yksitellan TEMPLATEn etc dirriin

cd $TEMPLATE/etc
for directory in $(find . -not -name '.' -type d 2>/dev/null) ; do
    mkdir -p $DEST_DIR/etc/$directory
done

for file in $(find . -type f -follow 2>/dev/null) ; do
    ln -f $TEMPLATE/etc/$file $DEST_DIR/etc/$file 2>/dev/null
done

## symlinkit manuaalisesti
cd $DEST_DIR/etc/sysconfig/network-scripts
ln -sf ../../../sbin/ifup ifup
ln -sf ../../../sbin/ifdown ifdown



####### filet buuttia varten, jotta saadaan nfs mount toimimaan

ln -f $TEMPLATE/bin/bash   $DEST_DIR/bin/bash
ln -f $TEMPLATE/bin/mount  $DEST_DIR/bin/mount
ln -f $TEMPLATE/sbin/init $DEST_DIR/sbin/init
cd $DEST_DIR/bin
ln -sf  bash  sh

ln -f $TEMPLATE/lib/ld-2.1.3.so $DEST_DIR/lib/ld-2.1.3.so
ln -f $TEMPLATE/lib/libc-2.1.3.so $DEST_DIR/lib/libc-2.1.3.so
ln -f $TEMPLATE/lib/libtermcap.so.2.0.8 $DEST_DIR/lib/libtermcap.so.2.0.8
cd $DEST_DIR/lib
ln -sf ld-2.1.3.so ld-linux.so.2
ln -sf  libc-2.1.3.so libc.so.6
ln -sf  libtermcap.so.2.0.8 libtermcap.so.2



######### node-kohtaiset konffit

## ainut yksilöllinen nodekohtainen file
cat << EOF > $DEST_DIR/etc/fstab
$SERVER:/tftpboot/$1 /          nfs rsize=8192,wsize=8192,intr,timeo=14
none                         /proc        proc   defaults   0 0
none                           /dev/pts        devpts  gid=5,mode=620  0 0
EOF


### nodet ajavat ensin modifioidun rc.sysinitin (mountataan nfs-levyt)
### jonka jälkeen kutsuu rc.sysinit.2:sta (=normaali rc.sysinit)

ln -f $TEMPLATE/etc/rc.d/rc.sysinit $DEST_DIR/etc/rc.d/rc.sysinit
ln -f $TEMPLATE/etc/rc.d/rc.sysinit.2 $DEST_DIR/etc/rc.d/rc.sysinit.2



#### lisää uusi kone hosts, hosts.equiv ja /usr/share/machines.LINUX fileihin

echo "$1 node$NODENRO" >> /etc/hosts
echo "$1" >> /etc/hosts.equiv
echo "node$NODENRO" >> /usr/share/machines.LINUX
```

```
#### tee hardlinkit tärkeimpiin systemwide asetustiedostoihin.
#### tämä ajettava myös aina uuden käyttäjän lisäyksen/poiston jälkeen.
/root/bin/updatelinks


### tulosta ohjeita käyttäjälle

echo ""
echo "Node IP: $1 = node$NODENRO lisätty järjestelmään"
echo ""
echo "Tarkista tiedostot:"
echo "  /etc/hosts"
echo "  /etc/hosts.equiv"
echo "  /usr/share/machines.LINUX"

echo ""
echo "====="
echo "Lisää kone /etc/dhcpd.conf tiedostoon ja aja /etc/rc.d/init.d/dhcpd restart"
echo ""
echo "host node$NODENRO {"
echo "   hardware ethernet <mac-osoite>"
echo "   fixed-address $1"
echo "}"
```

Liite E

hosts, hosts.equiv ja machines.LINUX

```
=== /etc/hosts ===
127.0.0.1 localhost.localdomain localhost
10.0.0.254 switch
10.0.0.1 node1 kaleeri kaleeri.uku.fi
10.0.0.2 node2
10.0.0.3 node3
10.0.0.4 node4
10.0.0.5 node5


=== /etc/hosts.equiv ===
10.0.0.1
10.0.0.2
10.0.0.3
10.0.0.4
10.0.0.5


=== /usr/share/machines.LINUX ===
# Change this file to contain the machines that you want to use
# to run MPI jobs on.  The format is one host name per line, with either
#    hostname
# or
#    hostname:n
# where n is the number of processors in an SMP.  The hostname should
# be the same as the result from the command "hostname"

node2
node3
node4
node5
```

Liite F

MPI-esimerkki 1

```
/*Parallel Hello World Program*/

#include <stdio.h>
#include <mpi.h>

main(int argc, char **argv) {
        int rank;
        MPI_Init(&argc,&argv);
        // jokainen prosessi saa yksillöllisen id:n (=rank)
        MPI_Comm_rank(MPI_COMM_WORLD, &rank);
        // jokainen kone tulostaa tekstin ja oman id:nsä
        printf("Hello World from Node %d\n",rank);
        MPI_Finalize();
}
```

*** ohjelman suoritus:

```
[jpitkane@kaleeri example]$ mpirun -np 5 ./hello
Hello World from Node 0
Hello World from Node 1
Hello World from Node 2
Hello World from Node 3
Hello World from Node 4
[jpitkane@kaleeri example]$
```

Liite G

MPI-esimerkki 2

```
/*Not So Simple Parallel Hello World Program*/

#include <stdio.h>
#include "mpi.h"
main(int argc, char **argv) {
        int rank, size, tag, i;
        MPI_Status status;
        char message[80];
        MPI_Init(&argc, &argv);
        MPI_Comm_size(MPI_COMM_WORLD, &size);  // alustaa size muuttujaan prosessien lukumäärän
        MPI_Comm_rank(MPI_COMM_WORLD, &rank);  // jokaiselle yksilöllinen prosessi id (=rank)
        tag = 100;


        // jos master, lähetä muille viesti  (0 = master)
        if (rank == 0) {
                strcpy(message, "Hello");
                // i = slave prosessien rank
                for (i=1; i<size; i++)
                        MPI_Send(message, 25, MPI_CHAR, i, tag, MPI_COMM_WORLD);
        }


        // jokainen slave odottaa viestiä masterilta, muuttaa sitä, lähettää takaisin
        else {
                // vastaanottavat viestin, jonka lähettäjän rank = 0 (master)
                MPI_Recv(message, 25, MPI_CHAR, 0, tag, MPI_COMM_WORLD, &status);
                strcat(message, ", this is slave");
                // lähettävät viestin takaisin rank 0:lle (master)
                MPI_Send(message, 25, MPI_CHAR, 0, tag, MPI_COMM_WORLD);
        }


        // master ottaa palautukset vastaan ja tulostaa
        if (rank == 0) {
```

```
                printf("%s, this is master\n",message);
                for (i=1; i<size; i++) {
                        // MPI_ANY_SOURCE = vastaanottaa viestin, jonka lähettäjä kuka vaan
                        MPI_Recv(message, 25, MPI_CHAR, MPI_ANY_SOURCE, tag, MPI_COMM_WORLD, &status);
                        printf("%s %d\n", message, i);
                }
        }
        MPI_Finalize();
}
```

*** ohjelman suoritus:

```
[jpitkane@kaleeri example]$ mpirun -np 5 ./hello2
Hello, this is master
Hello, this is slave 1
Hello, this is slave 2
Hello, this is slave 3
Hello, this is slave 4
[jpitkane@kaleeri example]$
```

Liite H

rc.firewall

```
#!/bin/sh

PATH=/bin:/usr/bin:/sbin:/usr/sbin

### alle 1024 porteista ulospäin auki vain sshd, lpr ja ftpd

ipchains -F

# vain yli 1024 input portit auki
ipchains -A input -s 0.0.0.0/0 -d 0.0.0.0/0 1024: -p tcp -j ACCEPT -i eth0
ipchains -A input -s 0.0.0.0/0 -d 0.0.0.0/0 1024: -p udp -j ACCEPT -i eth0

# erikseen avatut palvelut

# sshd
ipchains -A input -s 0.0.0.0/0 -d 0.0.0.0/0 ssh -p tcp -j ACCEPT -i eth0
ipchains -A input -s 0.0.0.0/0 -d 0.0.0.0/0 ssh -p udp -j ACCEPT -i eth0

# printer
ipchains -A input -s 0.0.0.0/0 -d 0.0.0.0/0 1022:1023 -p tcp -j ACCEPT -i eth0
ipchains -A input -s 0.0.0.0/0 -d 0.0.0.0/0 1022:1023 -p udp -j ACCEPT -i eth0

# ftpd
ipchains -A input -s 0.0.0.0/0 -d 0.0.0.0/0 20:20 -p tcp -j ACCEPT -i eth0
ipchains -A input -s 0.0.0.0/0 -d 0.0.0.0/0 21:21 -p tcp -j ACCEPT -i eth0

# deny kaikki muu
ipchains -A input -s 0.0.0.0/0 -d 0.0.0.0/0 -p tcp -j DENY -i eth0
ipchains -A input -s 0.0.0.0/0 -d 0.0.0.0/0 -p udp -j DENY -i eth0
```

Liite I

updatelinks.sh

```
#!/bin/sh

### skripti, joka käyttäjien lisäyksen tai tietojen muuttamisen jälkeen
###
### korjaa ed.m. operaatioissa rikkoontuneet hardlinkit

cd /tftpboot

for directory in $(find . -not -name '.' -type d -maxdepth 1) ; do
   cd $directory/etc
   ln -f /etc/passwd  passwd
   ln -f /etc/shadow  shadow
   ln -f /etc/group   group
   ln -f /etc/hosts   hosts
   ln -f /etc/hosts.equiv hosts.equiv
   cd ../..
done
```

Kirjallisuutta

[Cen98] Cornell Theory Center. Basics of mpi Programming, 1998. Saatavilla www-muodossa osoitteesta <http://www.tc.cornell.edu/Edu/Talks/MPI/Basic/>. Viitattu 8.8.2000.

[For95] MPI Forum. *MPI: A Message-Passing Interface Standard.* University of Tennessee, 1995.

[HM97] Juha Haataja and Kaj Mustikkamäki. *Rinnakkaisohjelmointi MPI:llä.* CSC - Tieteellinen laskenta Oy, 1997.

[Tea99] MPICH Team. Mpich - a portable mpi implementation, 1999. Saatavilla www-muodossa osoitteesta <http://www-unix.mcs.anl.gov/mpi/mpich/>. Viitattu 8.7.2000.