



TAMPEREEN
AMMATTIKORKEAKOULU

TUTKINTOTYÖRAPORTTI

**VUOROVAIKUTTEISEN KÄYTTÄJÄTUKIJÄRJESTELMÄN
LINUX-POHJAISEN TUOTANTOPALVELIMEN
KÄYTTÖÖNOTTO JA DOKUMENTOINTI**

Jani Luostarinen

Tietojenkäsittelyn koulutusohjelma
toukokuu 2006
Työn ohjaaja: Rami Lehtinen

TAMPERE 2006



Tekijä(t)	Jani Luostarinen	
Koulutusohjelma(t)	Tietojenkäsittely	
Tutkintotyön nimi	Vuorovaikuttamisen käyttäjätukijärjestelmän Linux-pohjaisen tuotantopalvelimen käyttöönotto ja dokumentointi	
Työn valmistumis- kuukausi ja -vuosi	Toukokuu 2006	
Työn ohjaaja	Rami Lehtinen	Sivumäärä: 45

TIIVISTELMÄ

Työn tavoitteena oli ottaa käyttöön ja dokumentoida Linux-pohjainen palvelin toimeksiantajan, Toyme Lab Oy:n, tuottamien palveluiden pohjaksi. Palvelut ovat Java-ohjelmointikielellä asiakkaalle kirjoitettuja tiedon haku- ja tallennuspalveluita. Käyttö tapahtuu sekä www-selaimella että matkapuhelimella lähetettävillä tekstiviesteillä.

Palvelin koostuu tietokannasta, Java servlet-säilöstä ja ohjelmistosta, joka välittää tekstiviestejä palvelimen ja käyttäjän välillä. Palvelimen asennuksessa pyrittiin ottamaan huomioon myös tietoturvaan sekä vikasietoisuuteen liittyviä seikkoja.

Työssä edellytetään vankkaa perustietämystä Linux-järjestelmistä sekä kokemusta niiden ylläpidosta. Perustiedot ohjelmoinnista ja tietokannoista auttavat ymmärtämään hyvin esiteltyjä ratkaisuja.

Lähdeaineistona on käytetty enimmäkseen työssä käytettyjen ohjelmistojen verkkosivuilla julkaistua virallista dokumentaatiota. Tämän lisäksi on hyödynnetty alan kirjallisuutta ja lehdissä julkaistuja artikkeleita.

Lopputuloksena saavutettiin tietoturvaltaan ja vakaudeltaan luotettavalla tasolla oleva palvelin sekä dokumentaatio jonka avulla voidaan myöhemmin ottaa käyttöön toiminnaltaan samanlainen ympäristö.

Palvelun pohjana olevia ohjelmistoja käytetään monenlaisten muidenkin palveluiden tarjoamiseen käyttäjille, joten dokumentoinnista on hyötyä myös muita Java- ja MySQL-pohjaisia palveluita työstettäessä.



Author(s)	Jani Luostarinen	
Degree Programme(s)	Business Information Systems	
Title	Implementation and documentation of Linux based production server for the interactive user support system	
Month and year	May 2006	
Supervisor	Rami Lehtinen	Pages: 45

ABSTRACT

The purpose of this work was to implement and write documentation of a Linux-based server to be used as a base for services provided by Toyme Lab Oy. These services are written in Java and they are meant for searching and archiving information. Services can be used with a web browser and text messages sent from mobile phone.

The server consists of a database, a Java Servlet container and software that deliver text messages between the server and user. Security and fault tolerance was kept in mind during the installation of the server.

This work requires some knowledge and experience of maintaining Linux-based systems. Basics of programming and databases make it easier to understand the results and methods presented in this document.

Sources are mostly collected from the official documentation of the project web pages. Some specific literature and articles from magazines has also been used.

As a result Toyme Lab has a secure and stable server environment with sufficient documentation to implement the same kind of system in future.

The programs used on this server can also be used for providing different kinds of services to users. This documentation can be useful in installation of a server providing Java and MySQL based services.

Keywords Linux Java server database servlet

1 JOHDANTO	5
2 VIITEKEHYS	6
2.1 TOYME LAB.....	6
2.2 PALVELINLAITTEISTO	7
2.3 LINUX.....	9
2.4 JAKARTA-TOMCAT SERVLET CONTAINER	11
2.5 MYSQL.....	13
2.6 KANNEL SMS GATEWAY	14
3 VAATIMUSMÄÄRITTELY	15
4 KÄYTTÖÖNOTON TOTEUTTAMINEN	17
4.1 ALKUVALMISTELUT	17
4.1.1 <i>Palvelimessa käytettävä laitteisto</i>	17
4.1.2 <i>Käyttöjärjestelmän asentaminen</i>	18
4.1.3 <i>Palvelimen TCP/IP – asetukset</i>	19
4.2 KRIITTISET PALVELUT	19
4.2.1 <i>Java 2 Software Development Kit (J2SDK)</i>	19
4.2.2 <i>Jakarta-Tomcat Servlet Container</i>	20
4.2.3 <i>MySQL</i>	22
4.2.4 <i>MySQL Connector/J</i>	22
4.2.5 <i>Kannel SMS Gateway</i>	23
4.2.6 <i>Firestarter palomuuuri</i>	23
4.3 TEHOSTAVAT PALVELUT	23
4.3.1 <i>Automysqlbackup</i>	23
4.3.2 <i>Häiriötön virransaanti</i>	24
4.3.3 <i>Levytilan jako</i>	25
4.3.4 <i>OpenSSH</i>	25
5 POHDINTA	26
5.1 PALVELIN	27
5.2 JAVA.....	28
5.3 TOMCAT	28
5.4 MYSQL.....	30
5.5 KANNEL	31
5.6 TIETOTURVA	32
6 ARVIO TAVOITTEIDEN TÄYTTYMISESTÄ	34
7 JATKOKEHITYS	36
LÄHTEET	37
LIITEET	39

1 Johdanto

Työn tavoitteena oli ottaa käyttöön ja dokumentoida Linux-pohjainen palvelin toimeksiantajan, Toyme Lab Oy:n, tuottamien palveluiden pohjaksi. Palvelut ovat Java-ohjelmointikielellä asiakkaalle kirjoitettu- ja tiedon haku- ja tallennuspalveluita. Palveluiden käyttö tapahtuu sekä www-selaimella että matkapuhelimella lähetettävillä tekstiviesteillä. Palveluissa käsiteltävä data sijaitsee yrityksen tiloissa olevalla palvelimella.

Tavoitteena oli myös palvelinympäristön dokumentointi niin, että toimenpiteet voidaan toistaa jatkossa siten, että palvelinympäristö säilyy pääpiirteittäin identtisenä uusilla käyttöön otettavilla palvelimilla. Dokumentoinnin tarkoituksena oli tukea omaa työskentelyäni palvelinten ylläpidossa tarjoamalla yhtenäisen tietolähteen palvelun osasten tämän hetkisestä konfiguraatiosta sekä syistä joiden vuoksi olen kyseisiin ratkaisuihin päätenyt. Yritykseen tulevaisuudessa tulevan työntekijän on myös helppo päästä sinuiksi uuden työympäristön kanssa ja jatkaa ympäristön ylläpitoa sekä kehittämistä. Dokumentointia voidaan hyödyntää myös muiden vastaavassa tilanteessa olevien yritysten palvelinten kehittämiseksi tai kokonaan uuden palvelimen käyttöönoton helpottamiseksi.

Työ ei sisällä ohjeita Linuxin asentamisesta, koska niitä on saatavilla runsain mitoin. Lukijalta edellytetään vankkaa perustietämystä Linux-järjestelmistä sekä jonkinlaista kokemusta niiden ylläpidosta. Perustiedot ohjelmoinnista ja tietokannoista auttavat ymmärtämään paremmin esiteltyjä ratkaisuja. TCP/IP-verkkojen perusteet sekä perustiedot ethernet-verkoista ovat myös tarpeen.

2 Viitekehys

2.1 Toyme Lab

Toyme Lab Oy toimii Tampereella ja työllisti toimeksiannon aikana kuusi henkilöä. Yritys käyttää palveluiden kehittämisessä sekä niiden tarjonnassa avoimen lähdekoodin tuotteita. Toyme Lab Oy on keskittynyt hiljaisen tiedon ongelmaan. Tällä tarkoitetaan organisaation tarvetta tallentaa kokemustietoa sähköiseen muotoon tietopääomaksi. Yritys kehittää ja markkinoi vuorovaikutteisen käyttäjätuen (Interactive User Support, IUS) järjestelmää. Se on suunnattu pääosin kone- ja laitteollisuuden asennus- sekä huoltohenkilöstölle.

Yritys tarjoaa interaktiivisen käyttäjätuen järjestelmää IUS-palveluna, jonka kautta kentällä asiakkaiden luona työskentelevät työntekijät vaihtavat tietoja muun organisaation kanssa. Palvelua käytetään yleisillä päätelaitteilla ja yhteystavoilla, kuten www-selaimella, sähköpostilla sekä tekstiviestillä. Palveluiden tarkoituksena on kerätä ja muokata jo kerättyä tietoa yhteen paikkaan, josta myös muut käyttäjät voivat sitä hyödyntää. Tiedolla tarkoitetaan tekstisisällön lisäksi myös liitetiedostoja, kuten kuvia ja erilaisia dokumentteja.

Yrityksen palvelimilla pyörii usean asiakkaan sovelluksia, joista jokaisella on useita käyttäjiä. Käyttäjät voivat ottaa yhteyttä palveluun ympäri maailmaa. Palveluiden pohjana olevia sovelluksia ajetaan Tomcat Servlet Container web-palvelimella ja ne hyödyntävät MySQL-relaatiotietokantaa tiedon tallennukseen. Kannel SMS Gatewayn tehtävänä on välittää tekstiviestit käyttäjältä Tomcatille ja paluuviestit takaisin käyttäjälle. Palvelun pohjana olevia ohjelmistoja käytetään monenlaisten palveluiden tarjoamiseen käyttäjille. Käytetyt ohjelmistot eivät rajoita tarjottua palvelua tiedon keräämiseen ja tallentamiseen, vaan niiden varaan voidaan rakentaa melkein mitä tahansa verkkokaupoista julkaisujärjestelmiin.

Yrityksellä oli palveluita varten palvelin, mutta sen tietoturvan ei uskottu olevan riittävällä tasolla. Palvelimen toiminnassa oli myös jonkin verran parannettavaa, niin laitteiston kuin ohjelmistojen osalta. Vanhaa palvelinta ei myöskään ollut dokumentoitu kovin tarkasti, joten sen ylläpito oli työlästä.

Kriittisillä palveluilla tarkoitan tässä työssä osia, joita ilman asiakkaalle tuotettavaa palvelua ei voitaisi toteuttaa. Niiden toimintaa voidaan mitata tietyinä hetkenä palvelun tavoitettavuutena ja sovelluksen toimintana odotetulla tavalla.

Tehostavat palvelut ovat osia, joilla on eniten merkitystä palveluita tarjoavalle yritykselle. Ne eivät tuo juurikaan lisäarvoa asiakkaalle, vaan niiden tavoitteena on selviytyä mahdollisimman hyvin ongelmatilanteista. Tällaisten palveluiden puute heijastuu asiakkaalle päin pitemmällä aikavälillä siten, ettei palvelu ole jatkuvasti saatavilla tai joi-tain hänen syöttämästään datasta on kadonnut.

Mukaan lasketaan myös palvelimen ylläpitotoimien suorittaminen ja tilan aktiivinen seuranta. Tässä tapauksessa ylläpito ja seuranta suori-tetaan etänä palvelimen fyysisen sijainnin vuoksi. Tämä tarkoittaa myös mahdollisuutta asentaa uusia sovelluksia asiakkaita varten sekä päivittää jo olemassa olevia versioita uudempiin. Yhdessä kriittiset ja tehostavat palvelut nitoutuvat saumattomaksi kokonaisuudeksi, jonka tarkoituksena on tarjota asiakkaalle palveluita ilman käyttökatoja tai riskiä tiedon katoamisesta.

Toyme Lab Oy:n perustarpeet palvelinympäristön kehittämiseksi ovat sellaisia, että niitä mietitään varmasti muissakin yrityksissä. Häiriötön virransaanti ja luotettava varmuuskopiointi ovat mielestäni vähintä mitä palveluiden tarjoamisessa tulisi ottaa huomioon.

2.2 Palvelinlaitteisto

Tietoliikenteessä palvelimella voidaan tarkoittaa sekä tietokoneessa pyörivää palvelinohjelmistoa, että itse tietokonetta, joka ajaa yhtä tai useampaa palvelinohjelmaa. Palvelinohjelmistojen tehtävänä on tarjota palveluita muille ohjelmille ja käyttäjille joko verkon yli tai samassa tietokoneessa. Palvelimella voidaan käsittää myös laajemmin lait-teiston, käyttöjärjestelmän sekä palvelinohjelmistojen muodostama kokonaisuus, jonka tarkoituksena on tarjota käyttäjille jotain palvelua.

Palvelimissa käytetty laitteisto on yleensä enemmän testattua ja var-matoimisempaa kuin kotona käytettävissä tietokoneissa. Jotkin osista voivat olla kahdennettuja vikasietoisuuden parantamiseksi. Palvelimet on suunniteltu olemaan jatkuvasti päällä toisin kuin pöytäkoneet, jotka tavallisesti sammutetaan käytön päätyttyä.

Palvelimen toiminnassa on otettava huomioon myös häiriötilanteet, kuten häiriöt sähköjakelussa ja palvelimen kovalevyn tuhoutuminen. Uninterruptible Power Supply (UPS) on järjestelmä tai laite, jonka tehtävä on taata tasainen virransyöttö lyhyissä katkoksissa ja syöttö-jännitteen epätasaisuuksissa. Se liitetään virtalähteen ja virtaa käyttä-vän laitteen väliin. UPS pystyy syöttämään suojaamilleen laitteille sähköä lyhyiden virtakatkoksienkin ajan akustaan.

Palvelimen laitteistona voidaan käyttää myös normaalia työpöytäko-neissa käytettyä tekniikkaa. Tällöin kustannukset ovat pienemmät,

mutta laitteisto ei ole yhtä suorituskykyinen ja varmatoiminen kuin varsinaista palvelimiin tarkoitettua laitteistoa käytettäessä. Pienemmissä ympäristöissä ratkaisu yleensä riittää kattamaan käyttäjien tarpeet. Tällaisissa tapauksissa vikasietoisuutta parannetaan yleensä käyttämällä Redundant Array of Independent Disks (RAID) levyjärjestelmää. Tässä järjestelmässä vikasietoisuutta, ja joissakin tapauksissa myös nopeutta, kasvatetaan liittämällä useampia fyysisiä levyjä yhdeksi loogiseksi levyksi (Frisch 2002:661).

Tekniikan ideana on tuottaa edullisilla kiintolevyillä PC-pohjaisiin tietokoneisiin kaupallisia levyjärjestelmiä vastaavia ominaisuuksia. Tekniikkaa käytetään ympäristöissä, joissa tiedon nopea saaminen ja virheettömyys ovat tärkeitä. Tällaisia ympäristöjä ovat esimerkiksi tietokanta ja tiedostopalvelimet (Østergaard 2004).

Yleisimmin tunnettuja ovat RAID0 ja RAID1 konfiguraatiot:

- RAID0 on tekniikka, jossa yhden loogisen levyn data jaetaan kahdelle fyysiselle levyille. Levyille voidaan kirjoittaa sekä lukea samanaikaisesti, jolloin teoreettinen nopeus kaksinkertaistuu. Mikäli toinen levyistä hajoaa, kaikki tallennettu tieto menetetään.
- RAID1 tekniikassa sama data peilataan kahdelle tai useammalle fyysisesti erilliselle levyille. Mikäli yksi levy hajoaa, kaikki data on yhä tallessa ehjillä levyillä. Periaatteessa tekniikka myös kaksinkertaistaa lukunopeuden, koska dataa voidaan lukea yhtä aikaa kahdelta levyiltä.

RAID-järjestelmä voidaan tehdä joko erillisellä tarkoitusta varten tehdyllä kortilla (Hardware RAID) tai käyttöjärjestelmän levyajureilla (Software RAID) (Østergaard 2004). Käyttöjärjestelmän levyajureilla toteutettaessa palvelimen suorituskyky laskee, koska prosessointiin käytetään koneen prosessoria. Samoin halvemmat RAID-kortit käyttävät nimestään huolimatta koneen prosessoritehoa prosessointiin.

Toinen levynkäyttöä tehostava menetelmä on Logical Volume Management (LVM). Tässä teknologiassa usealla levyllä olevaa tilaa voidaan yhdistellä isommiksi loogisiksi kokonaisuuksiksi. LVM:n etuna on se, että uusia osioita voidaan luoda helpommin sekä niiden kokoa on mahdollista muuttaa tarpeen mukaan (Frisch 2002:657). Palvelinta asennettaessa ei välttämättä ole tiedossa optimaalinen levytilan osiointi. Jokin osio voi esimerkiksi täytyä ennalta odottamattomasti, tai jokin osio halutaan erottaa myöhemmin omaksi osiokseen varmuuskopiointin helpottamiseksi (Lewis 2005).

Tekniikka mahdollistaa myös levyjen vaihtamisen ja lisäämisen järjestelmään siten, että niiden kapasiteetti voidaan ottaa suoraan käyttöön

lisätilaa kaipaavalle osiolle ilman, että osiota tarvitsee lähteä jakamaan pienempiin osiin, kuten normaalia osiointia käytettäessä. LVM voidaan yhdistää RAID:n käyttöön, jolloin palvelimelle saadaan joustavuutta osiointiin sekä lisää nopeutta ja vikasietoisuutta.

Yleensä palvelimissa on useampia prosessoreita jakamassa kuormaa. Moniprosessoriset palvelimet ovat reilusti kalliimpia kuin normaali PC-laitteisto. Normaalissa kuluttajille suunnatussa laitteistossa on tarjolla prosessoreita, jotka näkyvät käyttöjärjestelmälle kahtena loogisena prosessorina. Tekniikasta käytetään nimitystä Hyper Threading (HT). Tekniikka ei sinänsä vaadi ohjelmistolta muuta tukea kuin käyttöjärjestelmätasolla. Siitä on hyötyä vasta kun suoritetaan samaan aikaan vähintään kahta sovellusta. Vaihtoehtoisesti voidaan suorittaa sovellusta, joka suorittaa kahta säiettä yhtä aikaa. Parannus suorituskyvyssä riippuu hyvin paljon käytetystä sovelluksesta, ja joissakin tapauksissa sovellus voi jopa hidastua HT:n ollessa käytössä.

Palvelimen tulee sijaita fyysisesti paikassa, johon ainoastaan siihen oikeutetuilla henkilöillä on pääsy. Tunkeilija voisi laitteiston läheisyyteen päästessään aiheuttaa vahinkoa viemällä esimerkiksi kovalevyn, joka sisältää tärkeää tietoa. Hän voisi halutessaan myös tarkoituksellisesti vahingoittaa laitteistoa aiheuttaen yritykselle menetyksiä ja vahinkoa. Verkkolaitteet tulee myös suojata tarpeeksi hyvin, ettei niihin voida asentaa ulkopuolisia laitteita liikenteen kuuntelua varten.

2.3 Linux

Linux tai GNU/Linux on maailman yleisin käyttöjärjestelmä heti Windowsin jälkeen. Se kehitettiin alun perin pieneksi ja edulliseksi Unix-tyyppisten käyttöjärjestelmien kopioksi, mutta vähitellen siitä on kehittynyt varteenotettava ja monipuolinen käyttöjärjestelmä. Linuxia käytetään nykyään yleisesti palvelimissa, sillä se on todettu vakaaksi ja erittäin suorituskykyiseksi. Linux on nopea ja edullinen sekä toimii hyvin myös vanhemmissa laitteistoissa. Linuxiin on lisäksi saatavana paljon ilmaisia ohjelmia, joiden pohjalta voi kehittää omia ratkaisuja.

GNU/Linux-käyttöjärjestelmä koostuu Linux-ytimeistä sekä yleensä GNU-projektin tuottamista sovellusohjelmista ja työkaluista. Ohjelmien lähdekoodi on vapaasti saatavilla eikä niiden käytöstä peritä korvauksia. Ytimeistä ja sovellusohjelmista koostettuja ohjelmakokoelmia on useita ja niitä kutsutaan distribuutioiksi tai jakelupaketeiksi.

Linux on ilmainen käyttöjärjestelmä ja kaikkien ladattavissa internetistä, mutta siitä on olemassa myös kaupallisia versioita. Tällä hetkellä palvelimissa yleisimmin käytettyjä distribuutioita ovat Debian GNU/Linux, Red Hat Linux sekä Suse Linux. Näistä Red Hat sekä Suse ovat maksullisia ja tarjoavat käyttäjille tukea ja koulutusta. Debi-

an GNU/Linuxin tuki perustuu pitkälti muiden käyttäjien ja kehittäjien tarjoamaan tukeen.

Red Hatistä ja Suse:sta on saatavilla Enterprise-versiot, jotka ovat pidempään tuettuja ja paremmin testattuja kuin normaalit julkaisut. Ne ovat kalliimpia, mutta toisaalta kolmannen osapuolen sovellukset ovat yleensä paremmin dokumentoituja ja helpommin saatavilla näitä alustoja varten. Joistakin Enterprise-julkaisuista on saatavilla yhteisön ylläpitämiä ilmaisia versioita, jotka ovat pääosin yhteensopivia alkuperäisen tuotteen kanssa.

Linux-palvelinten sekä niillä ajettavien ohjelmien toimintaa säädellään skripteillä ja konfiguraatiotiedoilla. Konfiguraatiotiedostot sisältävät ohjeet ohjelman toiminnalle. Se luetaan yleensä ohjelmaan käynnistettäessä. Skripti on yleensä Bourne Again Shell (bash) komentotulkinnalle kirjoitettu yksinkertainen ohjelma, jonka tarkoituksena on jonkin toistuvan vaiheen automatisointi tai monia vaiheita sisältävän tehtävän yksinkertaistaminen.

Käytettävä laitteisto asettaa omat vaatimuksensa distribuution valinnalle. Lähinnä vaatimukset koskevat muistin ja levytilan määrää sekä tuettuja laitteita. Debian GNU/Linux vaatii vähemmän resursseja laitteistolta kuin Red Hat ja Suse. Tällä ei tosin enää nykyään ole juurikaan merkitystä, koska uudemmissa koneissa muistia on käytännössä aina tarpeeksi.

Linux-palvelimen ylläpitoon käytetään yleensä Secure Shell (ssh) -ohjelmistoa, joka sallii käyttäjän kirjautumisen palvelimelle omalta koneeltaan suojatussa yhteydessä (Puska 2001:267). Open Secure Shell (OpenSSH) on ilmainen versio ssh-protokollaa käyttävästä ohjelmistosta, joka mahdollistaa etäyhteyksien muodostamisen koneiden välille. OpenSSH salaa kaiken liikenteen, salasana mukaan lukien, turvallisen yhteyden takaamiseksi. Lisäksi se tukee lukuisia tunnelointi- ja tunnustautumistapoja (Frisch 2002:376). Toinen jonkin verran käytetty hallintatyökalu on Webmin (Puska 2001:272), mutta sillä ei voida aina suorittaa kaikkia tarvittavia komentoja ohjelmien asentamiseksi tai niiden asetusten muokkaamiseksi.

Linux-järjestelmissä käytetään palomuurina lähes poikkeuksetta iptablesia. Se on Linux-ytimen sisäänrakennetun palomuurin, netfilter:in, hallintaan tehty työkalu. Iptables luo netfilterille sääntöjä, jotka koostaan ketjuiksi, jotka taas muodostavat taulukoita. Iptables on mukana kaikissa uusissa Linux-jakeluissa ja se on korvannut vanhemman ipchains-nimisen työkalun (Netfilter.org project 2005). Yleensä iptables-pohjaista palomuuria hallitaan komentoriviltä tai skripteillä, mutta siihen on saatavilla myös graafisia käyttöliittymiä.

Toinen vaihtoehto Linuxin oman palomuurin sijasta tai rinnalle olisi erillisen rautapalomuurin käyttäminen. Tällaisia palomuuureja on saatavilla useilta valmistajilta halutuilla ominaisuuksilla.

2.4 Jakarta-Tomcat Servlet Container

Jakarta-Tomcat on servlet-säilö. Se on avoimen lähdekoodin Java-pohjainen web-palvelin, joka tukee servlettejä sekä JavaServer Pages (JSP) sivuja. Tomcatin kehityksestä vastaan pääosin Apache Software Foundationin Jakarta Project (Harrison & McFarland 2003:2).

Servlet mahdollistaa Java-ohjelmointikielellä kirjoitettujen sovellusten lisäävän dynaamista sisältöä web-palvelimelle. Luotu sisältö on usein HTML:ää (HyperText Markup Language), mutta voi olla muunkin muotoista dataa, kuten esimerkiksi XML:ää (Extensible Markup Language). Servletit ovat Javalla kirjoitettu vastine Advanced Server Pages (ASP) ja Hypertext Preprocessor (PHP) -teknologioille.

JSP tunnetaan Java-skriptien edeltäjänä. Teknologian avulla kehittäjiin on mahdollista luoda dynaamisesti HTML:ää, XML:ää tai muun tyyppisiä web-sivuja (Harrison & McFarland 2003:6). Se sallii Java-ohjelmointikielen ja joidenkin valmiiksi määriteltujen toimintojen sisällyttämisen staattiseen sisältöön. JSP-sivut käännetään servleteiksi käyttäen JSP-kääntäjää. Tomcat käyttää tähän työhön Jasper-kääntäjää (Harrison & McFarland 2003:3). Ohjelmisto on kirjoitettu Java-ohjelmointikielellä, joten se toimii kaikissa käyttöjärjestelmissä, joissa on käytettävissä Java Virtual Machine (JVM). JVM on ajoympäristö, virtuaalikone, joka suorittaa Javan virtuaalikoneen kielellä olevaa koodia. Virtuaalikone on tietokoneen abstrakti määrittely, jonka käskyt toteutetaan erityisen tulkin avulla. Tomcat on virallinen mallitoetus Sun:n JSP- ja servlet -palvelimesta.

Tällä hetkellä Tomcatista on saatavilla versiot 3.3.2, 4.1.31, 5.0.28 ja 5.5.9. Versiot 5.5.x ja 5.0.x tukevat samoja JSP- ja servlet-määrittelyjä. Kehittäminen keskittyy tällä hetkellä versioon 5.5.x ja siinä on merkittäviä suorituskykyyn ja vakauteen liittyviä parannuksia muihin versioihin nähden. Tomcat 5.5.x-versiot vaativat oletuksena Java Runtime Environment (JRE) 5.0:n käyttöä. Tomcatin Versiota 4.1.x näkee käytettävän yhä laajasti vaikkakin Jakarta-Tomcat-projekti suosittelee uusimpien versioiden käyttöä aina kuin mahdollista (Apache Tomcat 2006). Tätä vanhempia versioita ei juurikaan enää käytetä niiden ominaisuuksien puutteellisuuden vuoksi.

Tomcatia on perinteisesti käytetty yhdessä Apache web-palvelimen kanssa. Sitä voidaan käyttää myös yksin itsenäisenä web-palvelimenä. Aikaisemmin tämä ei ollut suositeltua ja Tomcatia käytettiin sellaiseen ainoastaan kehitysympäristöissä ja palveluissa, jotka eivät vaati-

neet nopeutta. Nykyään Tomcatia käytetään kasvavissa määrin sellaisenaan myös korkeaa saatavuutta vaativissa ympäristöissä. Ajettaessa Apachea ja Tomcatia yhdessä Apache vastaa staattisen ja Tomcat dynaamisen sisällön tarjoamisesta (Wiggers ym. 2002:235). Staattista sisältöä voivat olla esimerkiksi kuvat ja tyyli tiedostot. Apachea voidaan käyttää rinnalla myös muistakin syistä:

- Apache mahdollistaa hyvät mahdollisuudet klusterointiin. Yhdestä Apache web-palvelimesta voidaan ottaa yhteyttä useisiin Tomcat-palvelimiin. Tällöin yhden Tomcatin kaatuessa Apache ohjaa liikenteen muille Tomcateille.
- Apache tarjoaa enemmän työkaluja palveluiden turvaamiseksi. Sitä voidaan käyttää myös välityspalvelimena Tomcatien edessä, jolloin Tomcatit voidaan sijoittaa tiukemmin suojattuun verkon osaan.
- Apache tukee laajaa kirjoa lisäominaisuuksia, joiden käyttöönotto ja käyttö on nopeampaa kuin Tomcatissä. Nämä lisäominaisuudet mahdollistavat paljon sellaista, mikä ei ole toteutettavissa pelkän Tomcatin avulla.

Jotta Apache ja Tomcat voisivat toimia yhdessä, tarvitaan connector-moduuli, jonka avulla kommunikointi onnistuu. Tomcat tukee Apache Jserv Protocol (AJP) ja WARP-protokollaa käyttäviä connector-moduuleja (Harrison & McFarland 2003:34).

AJP on alkuperäinen Tomcat 3.x-versioita varten tehty connector-moduuli. Sitä ei enää kehitetä eikä se ole tuettu Tomcatin uudemmissa julkaisuissa. Mod_jk ja mod_jk2 ovat uusimpia connector-moduuleja. Näistä jälkimmäisen kehitys on kokonaan lopetettu eikä sitä enää ylläpidetä. Ensimmäistä vaihtoehtoa pidetään parhaana sen yksinkertaisuuden, vakauden ja tuen vuoksi. Se on myös aktiivisen kehityksen alla (Tomcat FAQ 2005).

WARP-protokolla luotiin Tomcat 4.x-versioita varten. Ainoan tällä hetkellä olemassa olevan connector-moduulin nimi on mod_webapp. WARP-protokollan oli tarkoitus olla joustavampi ja suorituskykyisempi kuin AJP-protokollan. Tällä hetkellä kehittäjien kiinnostus connector-moduulin kehittämiseksi on hiipunut ja AJP-connector-moduuleja pidetään parempina vaihtoehtoina (Tomcat FAQ 2005).

Oletuksena Tomcat kuuntelee porttia tcp/8080 (Harrison & McFarland 2003:29). Apachen Common Daemon projektiin kuuluvan Jsvc:n avulla voidaan käynnistää Java-sovellus prosessiksi taustalle sekä käyttää portteja, jotka eivät normaalisti ole muiden kuin pääkäyttäjän (root) käytettävissä.

Vaihtoehtoja Tomcatille on useita. Tomcatiin päädytään usein sen vähäisten laitteistovaatimusten ja keveyden vuoksi. Perinteisesti se ei ole ollut yhtä suorituskykyinen kuin kaupalliset kilpailijansa, mutta yleensä sen tehot ovat riittäneet varsin hyvin palveluissa, joiden liikenne pysyy kohtuullisena.

Kaupalliset vaihtoehdot:

- Resin Server
- BEA WebLogic Server
- Borland Enterprise Server
- Oracle Application Server
- IBM WebSphere
- Sun iPlanet
- Jboss
- IronFlare Orion Application Server
- WebObjects

Ei-kaupalliset vaihtoehdot:

- Jetty
- Java System Application Server
- Jaminid
- Enhydra
- jo!
- Java Mini Daemon
- Winstone

2.5 MySQL

MySQL:ää pidetään laadukkaana ja suorituskykyisenä avoimen lähdekoodin tietokantapalvelimena. Ohjelmistoa kehittää MySQL AB, joka levittää MySQL:ää GNU GPL -lisenssillä ja kaupallisella lisenssillä, jos GPL ei ole sopiva yrityksen tarpeisiin. Siihen on saatavilla kattava määrä erilaisia ajureita sekä hallinnointityökaluja.

Ilmaisia tai lähdekoodiltaan avoimia vaihtoehtoja ovat muun muassa PostgreSQL, Ingres, SQLite, Cloudscape ja Firebird. Tomcatin konfiguraatio-esimerkeissä käytetään yleensä MySQL:ää ja PostgreSQL:ää, joten niiden käyttöönotto on suhteellisen helppoa.

MySQL:n ja Tomcatin välisiin yhteyksiin käytetään yleisesti Connector/J-ajuria. Se on natiivi Java-ajuri, joka konvertoi Java Database Connectivity (JDBC) kutsuja MySQL tietokannan käyttämäksi verkkoprotokollaksi. Tämän ajurin avulla Tomcatin alla ajettavat sovellukset luovat yhteyden tietokantaan. JDBC on rajapinta Java-ohjelmointikielille, joka määrittelee, kuinka asiakasohjelma voi päästä

käsiksi tietokantaan (Harrison & McFarland 2003:271). Se tarjoaa tapoja kysellä sekä päivittää tietoja tietokannassa. MySQL Connector/J on JDBC-3.0 "Type 4" ajuri. Tämä tarkoittaa sitä, että se on kirjoitettu Java:lla ja kommunikoi suoraan MySQL-palvelimen kanssa käyttäen MySQL-protokollaa (Wiggers ym. 2002:235). Lisäksi sen tulee täyttää version 3.0 JDBC määrittelyt.

2.6 Kannel SMS Gateway

Kannel on avoimen lähdekoodin Wireless Application Protocol (WAP) ja Short Message Service (SMS, tekstiviesti) -yhdyskäytävä. Tekstiviestit ovat lyhyitä, yleensä 160 merkkiä pitkiä viestejä, joita lähetetään matkapuhelinverkossa tyypillisesti matkapuhelimesta toiseen. Kannel on laajalti käytössä ja erittäin suorituskykyinen. Palvelua varten tarvitaan joko puhelin tai gsm-modeemi vastaanottamaan ja välittämään tekstiviestejä palvelusta. Oman modeemin sijasta voidaan käyttää myös jonkun palveluntarjoajan valmista yhdyskäytävää. Sille on lukuisia kaupallisia vaihtoehtoja sekä jonkin verran avoimen lähdekoodin vaihtoehtoja:

- Tambur Messaging Gateway
- SMS Server Tools
- Alamin GSM SMS Gateway
- SMSLink
- iSMS

Avointen vaihtoehtojen kohdalla kehitys on varsin usein lopetettu eikä kaupallisten yhdyskäytävien ole taloudellisesti kannattavaa pienimuotoista palvelua pystytettäessä.

Selaimella sovelluksia käytettäessä pyyntö menee Tomcatille http- tai https-pyyntönä. Tomcat kommunikoi MySQL-tietokannan kanssa jdbc-kutsuilla (Liite 1). Tekstiviesteillä käytettäessä käytäntö on muuten sama, mutta tekstiviesti ei mene suoraan Tomcatille vaan Kannel SMS Gatewaylle, joka ohjaa pyynnön edelleen Tomcatille http pyyntönä ja lähettää kuittausviestin takaisin käyttäjälle (Liite 1).

3 Vaatimusmäärittely

Toyme Lab:lla aikaisemmin käytössä olleen palvelimen tietoturva oli heikohko johtuen käytetyn distribuution iästä (Redhat 7.2, julkaistu vuonna 2001), päivitysten puutteesta sekä asennusvaiheessa avoimiksi jääneistä palveluista. Vanhaa palvelinta ei kannattanut lähteä päivittämään, vaan rinnalle päätettiin rakentaa uusi palvelin. Tarkoituksena oli heti alusta alkaen kiinnittää huomiota tietoturvaan ja päivitettävyyteen, jotta vastaavalta tilanteelta välttyttäisiin tulevaisuudessa.

Mitään kriittisen palvelun muodostavaa ohjelmistoa ei haluttu vaihtaa toiseen ohjelmistoon, koska haluttiin säilyttää mahdollisimman hyvä yhteensopivuus edellisen palvelimen kanssa. Muutokset heijastuisivat laajemmin myös valmiiden asiakkaille tehtyjen sovellusten muokkaamisena uuteen ympäristöön, joka toisi ylimääräistä työtä.

Kriittisen palvelun muodostavat ohjelmistot ovat laajamittaisessa käytössä, jatkuvan kehityksen alla sekä toiminnaltaan luotettavia, joten niiden korvaaminen muilla ratkaisuilla ei ollut ajankohtaista näistäkään syistä.

Tietokannoista tuli ottaa varmuuskopiot säännöllisesti siltä varalta, että käytössä oleva kanta korruptoituisi tai palvelin hajoaisi fyysisesti. Aiemmin varmuuskopiot oli otettu manuaalisesti päivittäin ja kopiot oli lähetetty sähköpostilla tallennettavaksi sähköpostipalvelimelle. Mitään automatisoitua järjestelmää ei ollut vielä kokeiltu. Kantoja alkoi olla jo niin paljon, että inhimillisten virheiden riski lisääntyi. Näitä virheitä ovat muun muassa näppäilyvirheet kopioita otettaessa sekä varmuuskopioinnin unohtaminen.

Vanhaa palvelinta ei ollut suojattu myöskään sähkökatkoja ja virtapiikkejä vastaan. Tämä oli näkynyt satunnaisina palvelimen sammumisina ja siitä aiheutuneena ylimääräisenä työnä.

Palvelimelle asetetut vaatimukset listattuna osa-alueittain:

- Uuden palvelimen asennus, konfigurointi ja käyttöönotto vanhan palvelimen rinnalle. Palveluita oli tarkoitus ajaa rinnakkain siihen saakka kunnes uusi palvelin on toiminnalliselta tasoltaan saavuttanut vanhan palvelimen. Tämän jälkeen kaikki palvelut siirrettäisiin uuteen palvelimeen ja vanha palvelin jäisi reserviin siltä varalta, että jotain odottamatonta tapahtuisi.
- Automatisoidun varmuuskopiointijärjestelmän kehittäminen. Lisäksi tarvittiin menetelmä varmuuskopioiden siirtämiseen suojatussa yhteydessä muualla sijaitsevalle palvelimelle.

- Vikasietoisuus sähkökatkojen aikana. Palvelimet kärsivät pienistä verkkovirran katkoksista, jotka aiheuttavat koneen hallitsemattoman alasajon. Seurauksena tästä olivat katkot palveluissa sekä mahdollisuus tiedon korruptoitumiseen tilanteessa, jossa levyille kirjoittamista odottavaa muistissa sijaitsevaa dataa ei ole ehditty tallentaa levyille virran katketessa.
- Yrityksessä tarvittiin tallennustilaa jaetuille resursseille. Tallennustilaan tarvitaan pääsy Linux-pohjaisilta kehitysympäristöiltä. Tämän tallennustilan avulla varsinaiset sovellukset siirrettäisiin kehitysympäristöistä palvelimelle.
- Palvelimen käyttöjärjestelmä tulisi uusia nykyaikaisemmaksi ja päivitettävyyteen sekä distribuution elinkaaren pituuteen täytyisi kiinnittää huomiota.
- Kannel tuotti paljon päänvaivaa, koska yhteys puhelimen sekä palvelimen välillä katkeili välillä jostakin syystä. Tällöin viestit jäivät puhelimeen, eivätkä välittyneet enää palvelimelle asti. Kannelin uudelleen käynnistäminen palautti yhteyden mutta viestit jäivät jumiin puhelimen muistiin.
- Palvelin sijoitetaan sille erikseen varattuun tilaan, joten ylläpidon tulisi onnistua etänä ilman, että kenenkään tarvitsee käydä fyysisesti koneen luona. Palvelimen tulee olla turvallinen, luotettava ja helposti ylläpidettävä.

4 Käyttöönoton toteuttaminen

4.1 Alkuvalmistelut

Aluksi selvitin yrityksen verkon rakenteen ja laitteistoresurssit. Näillä tiedoilla pystyin jatkamaan tarkoitukseen sopivan Linux-jakeluversion etsimistä. Tavoitteena oli pitää asennettujen ohjelmien lukumäärä minimissä ja asentaa ainoastaan pakolliset sovellukset.

Vanhalla palvelimella olivat asennettuina seuraavat sovellukset ja versionumerot.

Sovellus	Versio
Jakarta-Tomcat	4.1.31
MySQL	4.0.22-2
J2SDK	1.4.2_06
Kannel SMS Gateway	1.2.1-5
Connector/J	3.0.1.16-ga

4.1.1 Palvelimessa käytettävä laitteisto

Uudeksi palvelimeksi valittiin aikaisemmin testikäytössä ollut laitteisto, jonka suorituskyvyn arvioin olevan riittävä nykyisiä tarpeita ajatellen. Laitteiston suorituskyky ei tulisi pullonkaulaksi palveluiden määrän lisääntyessä, koska tarvittaessa laitteiston suorituskykyä voidaan päivittää vielä helposti lisää. Otin ylös tarvittavat tiedot (Liite 2) ja selvitin yhteensopivuuden, ennen kuin aloin asentamaan palvelimelle käyttöjärjestelmää.

Vanhan palvelimen Kannel käyttää SMS-viestien välittämiseen Nokia 6310i matkapuhelinta, joka on kiinni palvelimen sarjaportissa RS-232 datakaapelilla (DLR-3P). Päätin ottaa saman kokoonpanon käyttöön uudelle palvelimelle. Palvelinta varten ostettiin myös keskeytymättömän virranjakelun mahdollistava laite, UPS.

Uuden palvelimen prosessori käytti Hyper Threadingia (HT), jolloin se näkyy käyttöjärjestelmälle kahtena erillisenä prosessorina. Tällöin voidaan käyttää normaalia moniprosessori ydintä (Symmetric Multi Processing, SMP). Tästä on hyötyä varsinkin ajettaessa rinnakkain useampaa sovellusta, tai sovelluksia, jotka suoraan tukevat tätä ominaisuutta.

4.1.2 Käyttöjärjestelmän asentaminen

Käyttöjärjestelmäksi valitsin ilmaisen ja vapaasti saatavissa olevan Debian GNU/Linux 3.1 ”Sarge” distribuution. Pohjaksi palvelimelle riittää Debian Sargen verkkoasennus minimi pakettivalikoimalla. Asennusta ei käydä läpi sen tarkemmin sillä se on käsitelty hyvin kattavasti Debianin dokumentaatioissa (Debian GNU/Linux 2005).

Suoritin asennuksen verkkoasennuksena lataamalla ensin noin sadan megatavun kokoisen käynnistyslevykuvan, jolta aloitin asennuksen. Tämän jälkeen latasin tarvittavat paketit verkosta tarpeen mukaan asennuksen aikana sekä sen jälkeen.

Perusasennuksen päätteeksi varmistin, että järjestelmä ja asennuspuu olivat ajan tasalla. Näin Aptitude-paketinhallintasovellus lataisi uusimmat versiot tarvittavista ohjelmista ja päivittäisi tarvittaessa vanhemmat versiot uudemmilla.

```
shell> su -c 'aptitude update && aptitude upgrade'
```

Varmistaakseni sen, että järjestelmässä on käytössä oikeat lokalisaatioasetukset ajoin alla mainitun komennon. Valitsin tämän jälkeen listasta ”fi_FI@euro ISO-8859-15” ja järjestelmän oletus kielisyydeksi ”none”, jolloin ohjelmien kieli pysyy englantina.

```
shell> su -c 'dpkg-reconfigure locales'
```

Muokkasin vielä /etc/environment tiedostoa sopivaksi. Alla muokkaamani tiedoston sisältö (Mäkelä 2001).

```
LANG=C
LC_CTYPE=fi_FI@euro
LC_PAPER=fi_FI@euro
LESSCHARSET=latin1
```

Palvelimessa on Hyper Threadingia hyödyntävä prosessori, joten oli tärkeää antaa järjestelmää käynnistäessä parametri ”acpi=ht”, muuten kone kaatuu ja jää jumiin käynnistysvaiheessa. Tämä tehdään helpoiten antamalla parametri käynnistyslataajalle (grub) optiona. Parametri voidaan antaa joko kaikille asennetuille ytimille yhteisesti, tai yhdelle ytimelle kerrallaan (jos käytössä on lisäksi muukin kuin smp-ydin).

Mikäli kaikki asennetut ytimet ovat moniprosessori-ytimiä, kannattaa parametri antaa kaikille samalla kertaa. Parametri voidaan antaa myös yhdelle ytimelle kerrallaan (Liite 3).

```
shell> su -c 'nano -w /boot/grub/menu.lst'
```

```
...
# kopt=root=/dev/hda2 ro acpi=ht
...
```

```
shell> su -c 'update-grub'
```

4.1.3 Palvelimen TCP/IP – asetukset

Yrityksessä oli käytössä G.SHDSL (G. Symmetric High-speed Digital Subscriber Line) -reititin yhdistettynä kytkimeen. Kytkin jakoi yhteyden palvelimille, joilla käytössä julkiset ip-osoitteet väliltä 62.236.92.224 - 62.236.92.255. Ensimmäinen laiteosoite (225) oli varattu reitittimelle. Reitittimen ylläpito on täysin palveluntarjoajan vastuulla.

- Verkon osoite 62.236.92.224
- Yleislähetys osoite 62.236.92.255
- Aliverkon peite 255.255.255.224

Asetin yhdelle verkkokortille käyttöön useampia ip-osoitteita. Osoitteet 62.236.92.245 ja 62.236.92.246 toimivat samassa fyysisessä laitteessa ja osoitteiden määrää voidaan jatkossakin kasvattaa tarpeen mukaan. Määrittelin verkko-asetukset (Liite 4) asetustiedostoon /etc/network/interfaces. Tämän jälkeen käynnistin verkon uudelleen, jotta tehdyt muutokset astuisivat voimaan.

```
shell > su -c '/etc/init.d/networking restart'
```

4.2 Kriittiset palvelut

4.2.1 Java 2 Software Development Kit (J2SDK)

Purin Sunin sivuilta lataamani paketin /tmp hakemistoon ja siirsin syntyneen hakemiston /usr/local/lib hakemistoon. Tämän jälkeen Loin symbolisen linkin, jotta versioiden päivitys olisi myöhemmin helpompaa.

```
shell> su -
shell> ln -s /usr/local/lib/j2sdk1.4.2_06
/usr/local/lib/jdk
shell> exit
```

Lisäsin /etc/profile tiedostoon seuraavat rivit, jotta sain JAVA_HOME ympäristömuuttujan käyttöön automaattisesti sisään kirjautuessa.

```
export JAVA_HOME="/usr/local/lib/jdk"
export JDK_HOME="${JAVA_HOME}"
```

Loin lopuksi asennetusta Javasta dummy paketit paketinhallintaa varten (Liite 5).

4.2.2 Jakarta-Tomcat Servlet Container

Purin paketin suoraan /opt hakemiston alle. Loin lisäksi symbolisen linkin syntyneestä hakemistosta osoittamaan /opt/tomcat hakemistoon.

```
shell> su -
shell> cd /opt
shell> wget
http://www.theshell.com/pub/apache/jakarta/tomcat-4/v4.1.31/bin/jakarta-tomcat-4.1.31.tar.gz
shell> tar -zxvf jakarta-tomcat-4.1.31.tar.gz
shell> ln -s jakarta-tomcat-4.1.31 /opt/tomcat
shell> exit
```

Aloitin konfiguroinnin luomalla Tomcatia varten oman käyttäjätunnuksen, jonka alla palvelua ajetaan.

```
shell> adduser --system --no-create-home --home \
/opt/tomcat --group tomcat
```

Tomcatin käynnistämiseen ja sammuttamiseen käytin jsvc-sovellusta, jonka avulla Tomcatin toimintaa voidaan hallita paremmin ja tietoturvaa parantaa.

Asentamani Tomcat 4.1.31:n mukana ei tule jsvc-sovelluksen asentamiseen tarvittavia tiedostoja, mutta löysin ne Tomcat 5.0.28 version bin hakemiston alta. Tämän jälkeen purin tiedostot ja käänsin niistä tarvittavan binääriin, sekä kopioin mukana tulevan mallikonfiguraatitiedoston käynnistyskriptiksi. Varmistin vielä, että skriptillä on tarvittavat suoritusoikeudet.

```
shell> tar -zxvf jsvd.tar.gz
shell> cd jsvc-source
shell> aptitude install autoconf
shell> sh support/buildconf.sh
shell> ./configure --with-java=/usr/local/lib/jdk
shell> make
shell> cp jsvc /opt/tomcat/bin/
shell> cp native/Tomcat.sh /etc/init.d/tomcat4
shell> chmod 700 /etc/init.d/tomcat4
```

Muutin Tomcatin tiedostojen omistajuudet oikeiksi ja määrittelin suoritusoikeudet.

```
shell> chown -R tomcat.devel /opt/tomcat
shell> chown -R tomcat.tomcat /opt/tomcat/conf
shell> chown -R tomcat.tomcat /opt/tomcat/bin
shell> chmod -R 770 /opt/tomcat
```

Tiedostossa `/etc/init.d/tomcat4` sijaitsevaan käynnistyskriptiin täytyi tehdä pari muutosta, jotta Tomcat voisi käynnistyä (Liite 6). Muutokset koskevat lähinnä asennettujen ohjelmien sijaintia ja Tomcatin suorituskykyä parantavia parametreja.

Asetin Tomcatin käynnistymään automaattisesti palvelimen ylösajon yhteydessä ja sammumaan hallitusti samalla kun konetta ajetaan alas.

```
shell> su -c 'update-rc.d tomcat4 default'
```

Asetin tämän jälkeen `/opt/tomcat/conf/server.xml` tiedostossa Tomcatin kuuntelemaan portteja 80 ja 443, joita http ja https palvelut normaalisti käyttävät (Liite 7).

Loin seuraavaksi sertifikaatin palvelimen autentikointia varten https yhteyksiä käytettäessä. Annoin kysyttäessä salasanan ”changeit” sertifikaatin ja keystoren salasanoiksi.

```
shell> $JAVA_HOME/bin/keytool -genkey -alias tomcat -
keyalg RSA
```

Muutin `$TOMCAT_HOME/conf/server.xml` tiedostoa seuraavasti, jotta palvelin ei listaisi kaikkia hakemistossa olevat tiedostoja siinä tapauksessa, ettei ennalta määritelyä index-dokumenttia löydy.

```
...
<init-param>
  <param-name>listings</param-name>
  <param-value>>false</param-value>
</init-param>
```

...

4.2.3 MySQL

MySQL:n asentaminen onnistui Debianissa helpoiten käyttäen valmiita paketteja paketinhallinnasta. Tämän jälkeen asetin MySQL:n pääkäyttäjälle salasanan tietokannan hallintaa varten.

```
shell> su -  
shell> aptitude install mysql-server  
shell> mysql_install_db  
shell> mysqladmin -u root password 'salasana'  
shell> mysqladmin -u root -h localhost password 'salasana'  
shell> exit
```

Varmistin vielä, että tiedostossa /etc/mysql/my.cnf oli asetettu hyväksymään ainoastaan paikalliset yhteydet.

```
bind-address = 127.0.0.1
```

Käynnistin lopuksi vielä MySQL palvelimen uudelleen, jotta asetuksiin tekemäni muutokset tulisivat voimaan.

```
shell> /etc/init.d mysql restart
```

4.2.4 MySQL Connector/J

Purin lataamani paketin tilapäishakemistoon. Kopioin paketissa olevan kirjaston Tomcatissä sitä varten olevaan hakemistoon, josta Tomcat ja ajettavat sovellukset löytäisivät sen. Tämän jälkeen käynnistin Tomcatin uudelleen, jotta uutta connector-moduulia voitaisiin käyttää.

```
shell> su -  
shell> cd /tmp  
shell> tar -zxvf mysql-connector-java-3.0.16.tar.gz  
shell> cp /tmp/MySQL-Connector-Java/mysql-connector-java-3.0.16/mysql-connector-java-3.0.16-bin.jar  
opt/tomcat/common/lib/  
shell> /etc/init.d tomcat restart
```

4.2.5 Kannel SMS Gateway

Kannelin asentaminen onnistuu helpoiten Kannelin kotisivuilla olevista Debian 3.1:stä varten tehdyistä paketeista.

```
shell> wget
http://kannel.org/download/1.2.1/deb/sarge/kannel_1.4.0-0_i386.deb
shell > su -c 'dpkg --install kannel*.deb'
```

Pystyin käyttämään vanhan palvelimen konfiguraatiota osittain hyväksi myös uudella palvelimella. Yhdistin entisestä ja mukana tulleista esimerkki-konfiguroinneista tarpeiden mukaiset asetukset, jotka käyttävät viestien välittämiseen Nokia 6310i puhelinta (Liite 8, Liite 9).

Mukana tuleva käynnistyskripti käynnisti oletuksena wapboxin smsboxin sijaan. Jotta oikea sovellus käynnistyisi, muokkasin tiedostoa /etc/default/kannel seuraavasti:

```
#START_WAPBOX=1
START_SMSBOX=1
```

4.2.6 Firestarter palomuri

Asensin palomuurin pakethallinnasta valmiiksi löytyvistä paketeista. Tämän jälkeen konfigurointi voidaan hoitaa tarpeiden mukaan graafisen käyttöliittymän kautta. Seuraamalla ensimmäisellä käynnistyskerrolla käynnistyvän konfigurointiohjelman ohjeita oli palomuri toimintakunnossa parissa minuutissa.

```
shell> su -c 'aptitude install firestarter'
```

4.3 Tehostavat palvelut

4.3.1 Automysqlbackup

Automysqlbackup on skripti, joka luo halutuista tietokannoista varmuuskopiot päivittäin, viikoittain ja kuukausittain omiin hakemistoihinsa. Sitä on mahdollista ajaa tarvittaessa joko käsin, tai ajastaa toiminto cron-ohjelman avulla.

```
shell> su -c 'mkdir /backups'
```

MySQL:ään täytyy luoda uusi käyttäjä (backup), jolla on oikeudet lukea (select) ja lukita (lock tables) jokainen varmuuskopioitava kanta

(Liite 10). Oikeudet kantoihin voidaan myöntää joko yksitellen, tai kaikkiin palvelimilla oleviin kantoihin kerralla.

Automysqlbackup-skriptiä tarvitsi muokata tarpeiden mukaan (Liite 11). Oleellisimmat asetukset koskivat kopioitavien kantojen nimiä sekä luotua MySQL:n käyttäjätunnusta.

Tämän jälkeen kopioin skriptin /etc/cron.daily hakemistoon, josta se ajettaisiin päivittäin. Varmistin vielä, että skriptillä oli suoritusoikeudet, jotta se voitaisiin suorittaa.

```
shell> su -c 'cp automysqlbackup /etc/init.d/'
shell> su -c 'chmod 750 /etc/cron.daily/automysqlbackup'
```

Tein yksinkertaisen skriptin (Liite 12), joka kopioi joka yö ainoastaan muuttuneet tiedostot etäpalvelimelle ja pitää samalla varmuuskopiot ajan tasalla. Etäpalvelimelta ei kuitenkaan poisteta mitään tiedostoja, vaikka ne varmistettavalta koneelta poistettaisiinkin. Esimerkiksi joka yö poistettava kuusi päivää vanha kopio kannasta jää silti talteen etäpalvelimelle. Tekemäni skripti voidaan myös ajaa käsin tai se voidaan ajastaa cronilla.

Loin ssh:ta varten julkisen avaimen palvelimella ja kopioin sen etäpalvelimelle. Tämän jälkeen kopioin luodun avaimen etäpalvelimen sallittujen avainten listaan.

```
shell> ssh-keygen -t dsa
shell> scp id_dsa.pub user@remotehost:
shell> ssh user@remotehost
shell> cat id_dsa.pub >> .ssh/authorized_keys
shell> rm id_dsa.pub
shell> exit
```

4.3.2 Häiriötön virransaanti

Yritykseen hankittiin APC-merkkinen keskeytymättömän virranjake-lun mahdollistava laite (UPS). Latasin ensin laitteen valmistajan ohjeiden mukaisesti ja sen jälkeen kytkin palvelimen virtajohdon laitteen takana olevaan virtaliittimeen.

Laite keskustelee palvelimen kanssa johdolla, joka on kytketty palvelimen Universal Serial Bus (USB) -porttiin ja UPS:n omaan rj-45 liitäntää muistuttavaan porttiin. Palvelin on tietoinen laitteen toiminnasta, kuten onko käytössä akku vai verkkovirta ja kuinka kauan akku kestää. Palvelin ajaa itsensä hallitusti alas siinä tapauksessa, että akun arvioitu kesto aika alittaa asetetun rajan (Apcupsd's User Guide 2005).

Asensin apcupsd-ohjelman, joka hoitaa kommunikoinnin laitteiden välillä. Konfiguroin apcupsd:n ohjeiden mukaisesti ”net master” laitteeksi. (Liite 13). Muutin lopuksi tiedostosta /etc/default/apcupsd asetuksen, jotta ohjelma käynnistyisi. Asetuksen tarkoitus on estää sovellusta käynnistymästä ennen kuin se on varmasti konfiguroitu.

```
ISCONFIGURED=yes
```

4.3.3 Levytilan jako

Päädyn käyttämään helppoa ja turvallista Secure FTP:tä (sftp) levyresurssien jakoon käyttäjille. Loin käyttäjille sen jälkeen käyttäjätunnukset, joita käyttämällä heidän oli mahdollista myös kirjautua palvelimelle ja käsitellä tiedostoja aivan kuten ne olisivat omalla koneella. Ainoa tarvittava toimenpide oli varmistaa, että tiedostosta /etc/ssh/sshd_config löytyi seuraava rivi:

```
Subsystem      sftp      /usr/lib/sftp-server
```

4.3.4 OpenSSH

Varmistin konfiguraatiodiedostosta, että käytössä on ainoastaan protokolla 2 ja sallin X-palvelimen käytön yhteyden aikana. Poistin mahdollisuuden kirjautua root tunnukseksi ja määrittelin oman käyttäjätunnukseni ainoaksi tunnukseksi, jolla järjestelmään voidaan kirjautua. Alla oleellimmat tekemäni muutokset tiedostoon /etc/ssh/sshd_config.

```
Protocol 2
PermitRootLogin no
AllowUsers jluostar
X11Forwarding yes
```

Tiedostoon /etc/hosts.allow määrittelin sen mistä osoitteista voidaan muodostaa yhteys ssh-palveluun.

```
sshd: 62.236.92.226
```

Tiedostossa /etc/hosts.deny kielletään lopuksi kaikki muut kuin erikseen sallitut yhteydet ssh palveluun.

```
sshd: ALL
```

5 Pohdinta

Käyttöjärjestelmän valinnan kriteerejä olivat tietoturva, helppo päivitettävyyttä sekä ylläpidon vaivattomuus. Valmiiksi paketoitujen sovellusten määrä oli myös huomioon otettava seikka sillä se helpottaa monesti sovellusten käyttöönottoa. Lisäksi kaikkien vanhassa palvelimessa olevien sovellusten tuli toimia moitteetta uudessa ympäristössä.

Tulimme siihen tulokseen, että Debian Sarge oli hallinnaltaan ja vakaudesta lähinnä sitä tasoa, jota palvelimelta yrityksen puolelta odotettiin (Laine 2004). Debianin vahvuuksina olivat mielestäni sen vakaus, tarpeeksi pitkä päivitysväli, laadukas päivitystyökalu sekä omat myönteiset kokemukseni kyseisestä distribuutiosta. Olen käyttänyt Debiania kolmen vuoden aikana useassa palvelimessa ja se on kokemusteni mukaan hyvä valinta myös tuotantoympäristöön.

Kolmessa tasossa etenevä kehitysmalli tekee Debianista erittäin vakaan. Ohjelmia käyttää ja testaa suuri määrä ihmisiä ennen kuin ne lisätään varsinaiseen jakeluversioon. Ohjelmaa, joka ei toimi tai häiritsee muiden toimintaa, ei oteta mukaan ennen korjauksien saamista.

Ylin taso on ”stable” eli vakaa. Sen ohjelmistot on todettu vakaiksi ja niistä on korjattu testikäytössä esiin tulleet viat. Tämä ei kuitenkaan tee tästäkään tasosta virheetöntä. Uusia ohjelmistovirheitä löytyy yhä. Virallisesti Debian suosittelee tätä tasoa käytettäväksi.

Joissakin tilanteissa haittapuolena on se, etteivät vakaan tason ohjelmistot ole kaikkein uusimpia. Yrityskäytössä olevalle palvelimelle korkea laatu ja paljon testatut ohjelmistot ovat yleensä tärkeämpi kriteeri valintaa tehdessä.

Toinen taso on ”testing” eli testattava. Ohjelmat ovat jo kohtalaisen vakaita, mutta niistä löytyy myös uusia ongelmia ja aivan kaikkia pieniä ongelmia ei ehkä vielä ole korjattu. Kotikäyttäjälle ja vähemmän kriittisissä ympäristöissä tämä yleensä riittää. Kolmas taso on ”unstable” eli epävakaa on tarkoitettu lähinnä testaamiseen, jotta pahimmat ohjelmistovirheet saataisiin esille ja korjattua.

Päivitysväli tarkoittaa uuden version muuttumista seuraavaksi vakaaksi julkaisuksi. Muilla distribuutiolla tämä väli lasketaan yleensä kuukausissa, Debianilla vuosissa. Päivittäminen uuteen versioon ei vaadi käyttöjärjestelmän uudelleen asennusta. Debianin käyttämä dpkg-paketinhallinta osaa hoitaa päivityksen, aivan kuten kyseessä olisi normaali tietoturvapäivitys.

Tavoitteena oli pitää asennettujen ohjelmien määrä minimissä ja asentaa ainoastaan tarpeelliset sovellukset. Moni järjestelmä sisältää haavoittuvuuksia ja aukkoja, jotka usein johtuvat siitä ettei järjestelmä ja siihen asennetut ylimääräiset ohjelmat ole tietoturvaltaan ajan tasalla. Yhdenkin ohjelman haavoittuvuuksia hyväksikäyttämällä koko järjestelmä voidaan ottaa haltuun. Tällöin kaiken järjestelmässä olevan tiedon oikeellisuus on kyseenalaista, eikä sitä voida välttämättä enää varmentaa. Ohjelmistoista ei ole pakko käyttää uusinta versiota, vaan sellaista versiota, jossa löydetty haavoittuvuudet on paikattu

Käyttöjärjestelmä lokalisaatioasetuksiin kannattaa kiinnittää huomiota, sillä esimerkiksi Tomcatin toimintaan vaikuttavat järjestelmän lokalisaatioasetukset. Väärillä asetuksilla esimerkiksi skandit voivat jäädä näkymättä vaikka kaikki toimisi muuten hyvin.

5.1 Palvelin

Kovalevy osioitiin boot-, swap- ja root-osioihin. Swap-osion koko on kaksinkertainen fyysisen muistin määrään verrattuna. Boot-osiolle varasin oman 100 MB osion, jolloin riski ytimen ja muiden käynnistyksessä tärkeiden tiedostojen korruptoitumiselle pienenisi. Lopputila varattiin root-osiolle. Tämä on varsin yleinen ja helppo tapa osioida kovalevy siinä tapauksessa, ettei muiden osioiden kokoa voida etukäteen arvioida. LVM olisi tuonut osiointiin joustavuutta jatkossa, mutta sitä ei ollut aikaisemmin kokeiltu ja sen testaaminen tuotantokäytössä olevalla palvelimella oli mielestäni turha riski.

RAID:ia ei otettu käyttöön, koska palvelimessa oli asennettuna vain yksi kovalevy. Vikasietoisuutta ajateltiin parantaa huolellisella varmuuskopioinnilla sekä pitämällä palvelimen rinnalla toista mahdollisimman identtistä palvelinta, joka voidaan ottaa käyttöön vikatilanteissa. RAID:illa saavutettava etu suorituskyvyssä ei myöskään ollut oleellinen, koska palvelimella ei tässä vaiheessa ollut niin paljoa levyintensiivistä käyttöä. Käyttöönottoa rajoitti myös se, ettei Linux-pohjaista RAID-järjestelmää oltu vielä testattu käytännössä.

Palvelimella on toiminnassa useamman asiakkaan sovelluksia ja joissakin tapauksissa asiakasta varten halutaan varata oma ip-osoite. Tällöin olisi voitu lisätä verkkokortteja jokaista ip-osoitetta varten, mutta koska koneessa on pci-paikkoja vain rajoitettu määrä, ei ratkaisu olisi ollut kestävä. Jokainen verkkokortti olisi vaatinut oman portin kytkimestä, jolloin kustannukset nousisivat entisestään. Tämän vuoksi yhdelle verkkokortille päätettiin määrittää useita osoitteita. Laitteistokustannukset pysyvät kurissa ja konfigurointi ei juuri eroa tilanteesta, jossa käytetään useaa verkkokorttia. Myöhemmin voidaan ottaa käyttöön tarvittaessa useampia verkkokortteja, joilla on useampia osoitteita. Näin yksittäisen kortin verkkoliikenne ei kasva kohtuuttomaksi.

Ohjelmien versioiden toivottiin mahdollisuuksien mukaan säilyvän pääosin samoina, jotta välttyttäisiin yhteensopivuus ongelmilta, joita päivityksistä olisi voinut aiheutua. Pienet erot numeroinnissa eivät yleensä aiheuta ongelmia niin pitkään kuin major -numero pysyy samana.

Käytin UPS:n konfiguroinnissa ”net master” konfigurointia, koska samaan laitteeseen tuli kiinni muitakin koneita. Nämä koneet keskustelevat palvelimen kanssa ethernetin yli ja ovat samalla sitä kautta tietoisia UPS:n tilasta. Näin myös muut koneet voidaan ajaa alas hallitusti häiriötilanteissa (Liite 14). Samaan UPS:iin liitetään myös muut verkon laitteet, kuten kytkimet ja reitittimet. Näin vältetään lyhyiden virtakatkosten aiheuttamat verkkolaitteiden uudelleen käynnistymiset. Tämä on myös välttämätöntä, jotta muut kytkimessä ja UPS:issa kiinni olevat laitteet saavat tiedon sähkönjakelun häiriöistä.

5.2 Java

Tomcatia varten riittäisi pelkkä JRE:n asentaminen, mutta koska palvelimella myös käännetään asiakkaiden sovelluksia, on Java Development Kit (JDK) tarpeen. Asennettu ”java-common” paketti toimii ”dummy”-pakettina, jonka avulla paketinhallinta saadaan tietoiseksi asennetusta Javasta. Tämä vaihe ei ole pakollinen mutta se helpottaa palvelimen ylläpitoa jatkossa (Debian GNU/Linux Java FAQ 2005). Tällöin paketinhallinta saadaan tietoiseksi asennetusta Java:sta, jolloin sen kautta voidaan asentaa ohjelmia jotka ovat riippuvaisia Javasta.

5.3 Tomcat

Tomcat asetettiin niin sanottuun Stand-alone tilaan, jossa se vastaa itse kaiken sisällön tarjoamisesta. Tämä on helpoin ja nopein tapa ottaa Tomcat käyttöön. Toinen vaihtoehto olisi ollut asettaa Apache webpalvelin Tomcatin rinnalle siten, että Tomcat vastaa dynaamisesta ja Apache staattisesta sisällöstä. Suorituskyky paranee, mutta kyseinen kokoonpano on vaikeampi ottaa käyttöön ja alttiimpi konfigurointivirheille.

Palvelut ovat kokonaisuudessaan dynaamista sisältöä, joten Apache ei olisi tuonut suorituskykyyn juurikaan parannusta. Ainoastaan ssl-suojatuissa yhteyksissä voisi olla marginaalinen ero Apachen hyväksi sen hoitaessa salauksen Tomcatin puolesta. Päätin siirtää Apachen ottamisen rinnalle tulevaisuuteen siinä tapauksessa, että palvelut sitä vaativat. Apache tulisi kyseeseen siinä tilanteessa, jos olisin halunnut käyttää joitain Apachen edistyneempiä ominaisuuksia, kuten esimerkiksi osoitteiden uudelleen ohjausta. Tällöin normaalit http yhteydet

voitaisiin pakottaa https-yhteyksiksi tekemättä asiakkaan sovellukseen yhtään muutosta.

Koska palveluntarjoaja on tukkinut kaikki muut portit paitsi tcp/22, tcp/80 ja tcp/443, ei verkon ulkopuolelta olisi ollut mahdollista päästä käsiksi Tomcatin tarjoamiin sovelluksiin oletusasetuksia käytettäessä. Käytettäessä porttia tcp/80 Tomcat-prosessi pyörii ainoastaan pääkäyttäjän (root) käyttöoikeuksilla, sillä ainoastaan root voi käynnistää palveluita jotka käyttävät portteja väliltä 1-1024. Tällöin prosessilla olisi kaikki oikeudet järjestelmän resursseihin, jolloin sovellusten toimintaa ei rajoittaisi mikään.

Tomcatia varten kannattaakin luoda oma käyttäjätunnukseksi ja ajaa prosessia tällä tunnukseksi. Näin oikeudet voidaan helposti rajoittaa ja tietoturvaa kohentaa. Loin tomcat- käyttäjätunnuksen, jolla ei ole mahdollista kirjautua sisään ennen kuin salasana on asetettu. Salasanaa ei aseteta tunnuksen luonnin yhteydessä, joten kirjautuminen on mahdotonta eikä ole riskiä, että joku pääsisi tunnukseksi sisään järjestelmään. Jsvc:llä prosessi saatiin pyörimään tarkoitusta varten luodun käyttäjätunnuksen alle haluttuun porttiin. Prosessilistausta katsottaessa Tomcatille näkyy jsvc:n vuoksi kaksi prosessitunnusta. Näistä rootin tunnukseksi pyörivä on ainoastaan käynnistänyt varsinaisen palvelun, joka pyörii tomcatin tunnukseksi ja oikeuksilla.

Kehittäjiä varten järjestelmään luotiin oma käyttäjäryhmä (devel), jotta resurssien jakaminen helpottuisi. Tällöin voidaan käyttää ryhmäkohtaisia rajoituksia resurssien oikeuksia myönnettäessä. Sallin develop-ryhmään kuuluvien työntekijöiden siirtää sovelluksia Tomcatin webapps hakemiston alle, josta Tomcat suorittaa ne. Suojasin loput Tomcatin hakemistot ja tiedostot siten, että ainoastaan tomcat tunnuksen alla pyörivät sovellukset voivat kirjoittaa ja lukea niitä.

Tomcat haluttiin käynnistyväksi automaattisesti järjestelmän käynnistyessä. Tätä varten tein skriptin, joka hoitaa palvelun käskemisen automaattisesti. Skripti linkitetään halutuille ajotasolle, jolloin se suoritetaan kyseiselle ajotasolle siirryttäessä. Pakotin vielä Tomcatin käyttämän merkistön halutuksi siksi, että merkistön kanssa voi joskus olla ongelmia käyttäjärjestelmän oikeista kielisyysasetuksista huolimatta (Liite 6).

Eräs asiakkaalle tehty sovellus vaati oletuksena yhteyden graafiseen palvelimeen (Xfree86 tai Xorg) kuvien tuottamiseksi annetusta datasta. Tuotannossa olevilla palvelimilla ei yleensä ajeta X palvelinta, koska se vie resursseja sekä lisää mahdollisten haavoittuvuuksien määrää tarpeettomasti. Ratkaisuna ongelmaan Tomcat käynnistettiin "headless" tilassa, jolloin X palvelinta ei tarvita (Tomcat FAQ 2005).

Tietoturvasyistä Tomcatin sovellusten hakemistoja ei haluttu listattavan. Normaalisti web-palvelin listaa kaikki hakemistossa olevat tiedostot, jos ennalta määriteltyä index-dokumenttia ei löydy. Tämä ei ole tietoturvan kannalta toivottu toimintatapa, koska palvelu on yleensä kaikille avoin. Tällöin kuka tahansa pääsisi näkemään esimerkiksi kansion sisällön arvaamalla sen nimen.

Päätin vaikuttaa Tomcatin muistinkäyttöön määrittelemällä sille varauksi ala- ja ylärajoiksi 256 megatavua keskusmuistia. Oletuksena muistia varataan maksimissaan 64 megatavua, jonka epäilin olevan riittämätön kattamaan usean sovelluksen tarpeita. Tomcat varaa muistia dynaamisesti tarpeen mukaan. Määrittelemällä alarajan muistille se varaa heti käynnistyessään tarpeeksi muistia käyttöönsä eikä käyttöjärjestelmän tarvitse vapauttaa sitä myöhemmin muilta prosesseilta.

5.4 MySQL

MySQL:ää ei voitu vaihtaa muuhun tietokantaan siksi, että sovellukset ja niiden käyttämät tietokannat on suunniteltu MySQL:n varaan. Kannan vaihtaminen johonkin muuhun aiheuttaisi tietokantojen ja sovellusten uudelleen suunnittelua, johon ei tässä tilanteessa haluttu lähteä.

MySQL hyväksyy oletuksena yhteydet vain paikallisesti. Koska muilla käyttäjillä ei ollut tarvetta päästä käsiksi kantaan etäkoneilta, oli kyseinen järjestely tietoturvan kannalta perusteltu. Tällä pyrin estämään ulkopuolisten tahojen pääsyä käsiksi tietokantoihin.

Pääkäyttäjän salasana tulisi asettaa aina, koska se estää oikeudettomien käyttäjien pääsyä käsiksi kantaan. Ilman salasanaa kuka tahansa järjestelmässä oleva käyttäjä voisi vahingossa tai tarkoituksella tuhota kokonaisia tietokantoja. Tämä estää myös tunnistautumattomien käyttäjien yhteydenotot kantaan paikallisesti siinä tapauksessa, että joku ulkopuolinen olisi päässyt kirjautumaan palvelimelle.

Joissakin vanhemmissa Debiania varten paketoituissa MySQL palvelimen versioissa on asetus ”skip-networking”, joka estää verkon kautta tapahtuvat yhteydet. Samalla se estää Tomcatin pääsyn käsiksi kantoihin, joten asetus tulee ottaa pois päältä. Tämän hetkisessä versiossa mainittu asetus on korvattu turvallisemmalla ja yhteensopivammalla tavalla, jossa oletuksena MySQL kuuntelee verkon kautta tapahtuvia yhteydet ainoastaan localhostilta. Tässä tapauksessa Tomcatia ajetaan samalla koneella kuin MySQL:ää, joten yhteydet muodostuvat localhostin kautta ongelmitta. Myöhemmin yhteydet voidaan tarvittaessa sallia joko sisäverkon koneille tai vaikka ainoastaan yhdelle käyttäjälle yhdeltä koneelta tarpeen mukaan. Tällainen tilanne voisi olla esimerkiksi silloin kun tietokanta halutaan pitää eri palvelimella kuin Tomcat.

Tietokantojen varmuuskopioiden ottaminen on päivittäin tapahtuva toimenpide, joten se tuli mielestäni automatisoida. Kannat haluttiin varmistaa yöllä kun käyttäjiä on vähemmän. Muokkaamalla /etc/crontab tiedostoa aikatalutus voidaan säätää mieleiseksi. Liitteenä oleva esimerkki siirtää cronin käynnistämään skriptit kello 23.25 jälkeen (Liite 15).

Kopioiden pakkaukseen kannattaa valita bzip2 formaatti, koska se on pakkausalgoritmiltaan tehokkaampi, joskin jonkin verran hitaampi. Korruptoituneista paketeista on myöhemmin mahdollista palauttaa tietoja (Bzip2 and libbzip2 official home page 2005). Varmuuskopioita kannattaa säilyttää fyysisesti eri levyillä, koneella tai mieluiten jossakin muussa paikassa kuin varmistettava järjestelmä. Tämä sen vuoksi, että esimerkiksi tulipalon sattuessa data on silti turvassa.

Varmuuskopiot siirretään talteen rsync:illä ssh-putken läpi, jolloin lienne on suojattu eivätkä ulkopuoliset tahot pääse siihen käsiksi. Etuna on myös se, ettei rsync:ille tarvitse avata omaa porttia palomuriin, vaan voidaan käyttää ssh:n käyttämää porttia. Koska yhteys muodostetaan ssh:n kautta, kysyy skripti oletuksena aina käynnistyessään etäkoneen käyttäjän salasanaa. Tämä ei toimi automatisoidussa ympäristössä joten käytin autentikointiin julkisen avaimen menetelmää. Koneelle jää salainen avain ja etäkoneelle kopioidaan julkinen avain. Etäkone tunnistaa yhteyttä muodostavan koneen vertaamalla sen lähettämää avainta listassaan oleviin avaimiin ja sallii yhteyden jos avain löytyy.

Cron vastaa sekä paikallisten varmuuskopioiden ottamisesta, että niiden siirtämisestä etäkoneelle (Liite 16). Skriptien ajoitus on säädetty niin, että paikalliset kopiot otetaan ennen kuin tietojen päivitys etäkoneelle tapahtuu.

5.5 Kannel

Kannel SMS-yhdyskäytävää oli testattu yrityksessä sen verran kauan, että siihen luotettiin siinä määrin, ettei korvaavan ohjelmiston etsiminen ollut ajankohtaista. Viestien välitystä varten otettiin käyttöön oma laite. Palvelut ovat näin tavoitettavissa tekstiviestin välityksellä siinäkin tapauksessa, että verkkoyhteys yrityksen ja palveluntarjoajan SMS yhdyskäytävän välillä katkeaisi.

Päivitin Kannelin uudempaan versioon, koska se tuo mukanaan ominaisuuksia, jotka auttavat aiemmin ilmenneissä ongelmissa. Tarkoituksena oli pitäytyä pääosin vanhemman palvelimen versionumerossa. Vanhempi versio sisälsi kuitenkin selkeitä puutteita, jotka uudem-

massa versiossa oli paikattu, joten päivityksen katsottiin olevan tarpeellinen.

”*Keepalive*”-asetus kääkee Kannelia tarkistamaan minuutin välein yhteyden puhelimeen ja muodostamaan sen uudelleen, jos se on jostain syystä katkennut (Kannel 1.4.0 User's Guide 2004).

”*Sim-buffering*”-asetus takaa sen, että kaikki puhelimeen tänä aikana tulleet viestit välitetään edelleen Kannelille yhteyden palattua. Uudet viestit luetaan sim-kortilta, jonne käytettävä puhelin ne oletuksena tallentaa (Kannel 1.4.0 User's Guide 2004). Viesti poistetaan kortilta sen jälkeen, kun se on välitetty eteenpäin, joten sim-kortin ei pitäisi päästä täyttymään missään vaiheessa.

5.6 Tietoturva

Järjestelmän ylläpitoon ja varmuuskopioiden tunnelointiin käytetään pääasiassa SSH:ta. Lisäksi sen avulla siirretään sovellukset kehitysympäristöistä palvelimille ja jaetaan levytilaa verkossa käyttäjien omille sekä jaetuille tiedostoille. Tietoturvan kannalta suositeltavaa on poistaa mahdollisuus kirjautua järjestelmään etänä root tunnuksella, koska ssh:n portteihin on viime aikoina tehty paljon järjestelmällisiä kirjautumisyriytyksiä kyseisellä tunnuksella (SSH: The Secure Shell 2005).

Rajoin palvelun sallimaan yhteydet ainoastaan erikseen sallituilta käyttäjiltä. Tällöin ei synny riskiä, jossa jonkin asennetun palvelun oletus salasanalla päästäisiin sisään järjestelmään. Salasanojen on silti oltava laadukkaita, mutta riskinä on silti se, että ne murretaan käyttäen koneen tehoa salasanojen generoimiseen ja kirjautumisyriytyksiin luoduilla salasanoilla. Suojattu yhteys ei sellaisenaan riitä estämään murtautumista.

Halusin tiukentaa tietoturvaa vielä entisestään määrittelemällä etukäteen osoitteet, joista yhteyden otot ovat sallittuja. Muista osoitteista ei ole mahdollista kirjautua järjestelmään, vaikka tietäisi käyttäjätunnuksen sekä salasanan. Kirjautumiset hyväksytään ainoastaan käytettäessä ssh-protokollasta versiota 2.

Sallin graafisten ohjelmien ajon palvelimelta, jotta palomuurin ja tietokantojen hallinta voidaan hoitaa graafisella käyttöliittymällä suojatussa yhteydessä, aivan kuten käyttäjä olisi koneen ääressä. Yhteyttä muodostettaessa määritellään xserverin uudelleen ohjaus optiolla -X.

```
shell> ssh -X user@remotehost
```


Käytettäessä sftp:tä levytilan jakoon mitään uutta palvelua ei myöskään tarvittu ottaa käyttöön, eikä palomuurin asetuksiin tarvinnut koskea. Tällöin hallinta ja ylläpito helpottuvat.

Vaikka yrityksellä on käytössä palveluntarjoajan palomuuuri, halusin suojata palvelimen palvelut vielä omalla palomuurilla. Tämän tarkoituksena oli suojata mahdollisesti omasta verkostamme tulevat hyökkäykset, joita esimerkiksi viruksella saastunut työasema saattaisi aiheuttaa. Rautapalomuurin käyttäminen ei ollut mielestäni järkevää, koska ne maksavat aina enemmän kuin Linuxin mukana tuleva ilmainen Iptables. Palomuurin tarkoituksena oli muutenkin olla paikallaan vain varmuuden vuoksi siltä varalta, että palveluntarjoajan palomuuuri pettäisi.

Tärkeimpiä ominaisuuksia palomuurin valintaa tehdessä olivat sisään päin tulevan liikenteen hallinnointi, porttien uudelleen ohjaus, liikenteen tarkkailu sekä ICMP sääntöjen muokkaus palvelun esto hyökkäyksen (Denial of Service, DoS) hyökkäysten estämiseksi. Vaihtoehtoina palvelimen palomuuriksi olivat alusta alkaen Firestarter sekä käsin kirjoitetut skriptit. Valitsin palomuuriksi Firestarterin, koska minulla oli siitä aikaisempia hyviä kokemuksia. Se myös yksinkertaisti hallintaa huomattavasti sekä mahdollisti nopeat muutokset palomuurin toiminnassa.

Firestarter on graafinen käyttöliittymä Linuxin sisäisen palomuurin (iptables) säätämistä sekä hallintaa varten (Firestarter 2005). Omien kokemusteni mukaan kyseessä on monipuolinen ja helposti hallittava ohjelmisto. Ohjelma tukee 2.4 ja 2.6 sarjan ytimiä, jotka molemmat ovat asennettavissa Debianiin. Tein palomuuriin yksinkertaiset säännöt sisään tulevalle liikenteelle. Ainoastaan TCP-portteihin 22 (SSH), 80 (http) ja 443 (https) tulevat yhteydet sallitaan. Sovelluskohtaiset pääsyräjoitukset hoidetaan tarvittaessa Tomcatilla.

6 Arvio tavoitteiden täyttymisestä

Uusi palvelin korvasi vanhan palvelimen varsin nopeasti. Käytännössä palvelimia ajettiin rinnakkain noin viikko samoilla sovelluksilla ja tietokannoilla. Tämän jälkeen vanha palvelin otettiin irti verkosta ja siinä olevat viimeisimmät tietokannat siirrettiin uudelle palvelimelle. Vanhan palvelimen ip-osoite siirrettiin uudelle palvelimelle, ja se kytkettiin verkkoon. Uudelle palvelimelle asennettiin pääosin samat versiot käytössä olleista kriittisten palveluiden ohjelmistoista. Tämä helpotti merkittävästi sovellusten siirtoa vanhalta palvelimelta uudelle.

Palvelimelle asennetut sovellukset versionumeroineen:

Sovellus	Versio
Jakarta-Tomcat	4.1.31
MySQL	4.0.23-4
J2SDK	1.4.2_06
Kannel SMS Gateway	1.4.0-1
Connector/J	3.0.1.16-ga

Uusi palvelin täyttää sille asetetut toiminnalliset vaatimukset. Entiseltä palvelimelta siirretyt asiakkaiden käytössä olevat sovellukset toimivat pienillä sovelluskohtaisten konfiguraatitiedostojen muokkauksilla. Lähinnä nämä muokkaukset koskivat Tomcatin asennuksen uutta sijaintia ja tietokannan käyttäjätunnuksia sekä salasanoja, joita muutettiin päivityksen yhteydessä turvallisemmiksi.

Palvelimen looginen rakenne ja IUS palvelun pohjan muodostava osa voidaan jakaa seuraavasti: Tomcat, Kannel ja MySQL muodostavat varsinaisen palvelun pohjan ja muut komponentit mahdollistavat sen luotettavan toiminnan. Apcupsd toimii häiriöttömän virransyötön ja palvelimen keskustelukanavana ja Automysqlbackup varmistaa asiakkaan datan säilymisen. Iptablesin tehtävänä on suojata palvelin verkosta tulevilta uhilta (Liite 17).

Tietokantojen varmuuskopioinnin toimintaa seurataan viikoittain ylläpitäjälle tulevista ilmoituksista. Tähän mennessä varmuuskopioinnissa ei ole ilmennyt ongelmia tai virhetilanteita..

Kaikki ylläpitotoimet voidaan suorittaa etäyhteyksien kautta. Palvelinta voidaan ylläpitää joko yrityksen tiloista, tai ennalta määritellyistä osoitteista, esimerkiksi ylläpitäjän kotoa. Ainoastaan palvelinta uudelleen käynnistettäessä on hyvä olla palvelimen välittömässä läheisyydessä ja valvoa, ettei käynnistyksen aikana ilmene ongelmia.

Henkilökohtaiseen tallennustilaan pääsee helpoiten käsiksi sftp-protokollaa osaavilla asiakasohjelmilla. Kehitysprojektin aikana myös kaikki Windows työasemat korvattiin Linux-työasemilla. Käytössä oleva KDE-työpöytäympäristö ja Konqueror-selain mahdollistavat tällaisten resurssien käytön aivan kuten ne olisivat omalla levyllä. Käyttäjien palautteet ovat olleet positiivisia.

Uusi palvelin ei ole tähän mennessä kaatunut kertaakaan. Kaikki uudelleen käynnistykset ovat olleet suunniteltuja ja ne ovat sujuneet ongelmitta. Palvelut ovat olleet asiakkaiden käytettävissä lähes koko ajan edellä mainituista toimenpiteistä aiheutuneet katkot pois lukien. Palomuuuri voisi mielestäni kuitenkin olla skriptipohjainen, sillä asetukseen ei juuri tarvitse puuttua, ja nykyinen graafinen asetusohjelma on varsin raskas etänä käytettäessä.

7 Jatkokehitys

Palvelimelle asetettujen ajastettujen toimintojen toimintaa voidaan seurata kätevästi sähköposteista. Järjestelmä lähettää sähköpostilla tulosteen ajetusta toiminnosta, josta näkee nopeasti onko kaikki sujunut niin kuin on suunniteltu. Varmuuskopiot kannattaa lisäksi polttaa säännöllisesti esimerkiksi cd:lle kaiken varmuuden vuoksi. Näin voidaan olla varmoja, että kannat voidaan aina palauttaa tilanteeseen, jossa ne olivat ennen häiriötä.

Käyttöjärjestelmän tietoturvapäivitysten tilanne olisi hyvä tarkistaa vähintään kerran viikossa. Päivitysten asentaminen kannattaa ajoittaa siten, ettei palveluilla ole paljoa käyttäjiä ja varautua kaiken varalta siihen, että jokin menee pieleen.

RAID tulisi mielestäni ottaa käyttöön mahdollisimman pian. Näin palvelin olisi sellaisenaan vikasietoisempi. Koko palvelimen tiedostojärjestelmää ei ole sellaisenaan mahdollista tai edes järkevää varmuuskopioida. LVM mahdollistaisi joustavan menetelmän, jolla esimerkiksi Tomcat:iä ja MySQL:ää varten voitaisiin luoda omat osionsa. Näin voitaisiin estää koko levyn täyttyminen virhetilanteissa. Tallennustilan lisääminen helpottuisi ja koko levykapasiteetti saataisiin paremmin hyödynnettyä. Tämä ei ole välttämätöntä, mutta suositeltavaa, sillä se helpottaa ylläpitoa

Kannel SMS Gatewayssä kiinni oleva gsm-modeemina toimiva Nokian puhelin riittää tällä hetkellä palvelemaan asiakkaiden tarpeita. Asiakkaiden määrän lisääntyessä se voidaan korvata oikealla gsm-modeemilla, jolloin palvelun käyttövarmuuden ja kapasiteetin pitäisi kasvaa. Ainakin Siemens M20 laitteelle löytyy valmiit konfigurointiasetukset Kannelin dokumentoinnista. Tällaisen laitteen käyttöönotto lienee helpompaa dokumentoimattomaan laitteeseen verrattuna.

Lähteet

Apache Tomcat 2006. [online] [viitattu 30.05.2005].
tomcat.apache.org

Apcupsd's User Guide 2005. [online] [viitattu 08.08.2005].
www2.apcupsd.com/3.10.x-manual/index.html

Bzip2 and libbzip2 official home page 2005. [online] [viitattu 30.05.2005].
www.digistar.com/bzip2/

Debian GNU/Linux 2005. [online] [viitattu 30.05.2005].
www.debian.org/

Debian GNU/Linux Java FAQ 2005. [online] [viitattu 30.05.2005].
www.nl.debian.org/doc/manuals/debian-java-faq/

Firestarter 2005. [online] [viitattu 30.05.2005].
www.fs-security.com/

Frisch Aeleen.2002. Essential System Administration, Third Edition.
Sebastopol: O'Reilly.

Harrison, Peter, McFarland Ian 2003. Mastering Tomcat Development. Wiley Publishing Inc.

Kannel 1.4.0 User's Guide 2004. [online] [viitattu 30.05.2005].
kannel.org/download/1.4.0/userguide-1.4.0/userguide.html

Laine, Kari 2004. Toyme Lab Oy, tuotantopäällikkö. Keskustelu 13.12.2004. Tampere.

Lewis, AJ 2005. LVM HOWTO [online] [viitattu 30.05.2005].
www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/html_single/LVM-HOWTO.html

Mäkelä, Ari 2001. Finnish HOWTO. [online] [viitattu 30.05.2005].
arska.org/finnish-howto/html/

Netfilter.org project 2005. [online] [viitattu 30.05.2005].
www.netfilter.org/

Puska, Matti 2001. Linux palvelimena. Helsinki: Talentum Media Oy

SSH: The Secure Shell 2005. Linux Format LXF63 February, 50-51

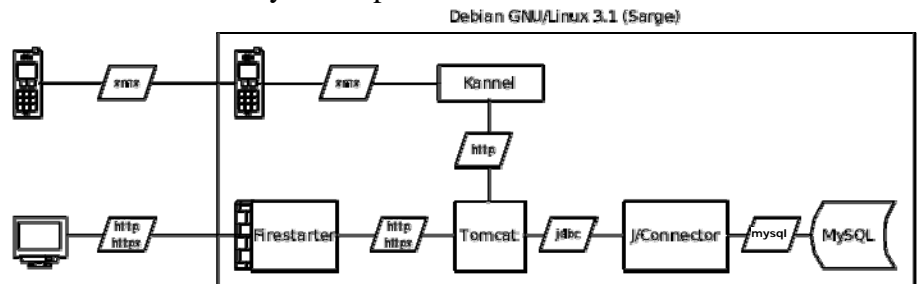
Tomcat FAQ 2005. [online] [viitattu 30.05.2005].
jakarta.apache.org/tomcat/faq

Wiggers, Chanoch, Galbraith, Ben, Chopra, Vivek, Li, Sing, Bhattacharjee, Debashish, Bakore, Amit, Irani, Romin, Bhattacharya, Sandip, Fowler, Chad 2002. Professional Apache Tomcat

Østergaard, Jakob & Bueso, Emilio 2004. The Software-RAID HOWTO. [online] [viitattu 30.05.2005].
www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/html_single/Software-RAID-HOWTO.html

Liitteet

Liite 1. Tiedon välittyminen palvelussa



Liite 2: Palvelimen laitteisto

CPU	Pentium 4, 2.6 GHz HT (SMP)	
RAM	512 MB	
Näytönohjaimen ajuri	i810	
Monitori (Hsync&Vsync)	30-70 & 50-160	
Kovalevy	80 GB	
Osiointi	100MB	/boot
	1GB	swap
	78 GB	/
Verkkokortin ajuri	e100 tai eepro100	

Liite 3: Käynnistyslataajan konfigurointi

```

...
### END DEBIAN AUTOMAGIC KERNELS LIST

title                Debian GNU/Linux, 2.6.8-2-686-smp
HT
root                 (hd0,0)
kernel               /vmlinuz-2.6.8-2-686-smp
root=/dev/hda2 ro acpi=ht
initrd               /initrd.img-2.6.8-2-686-smp
savedefault
boot
...
  
```

Liite 4: Verkko-asetukset

```

auto eth0 eth0:0

iface eth0 inet static
    address 62.236.92.246
    netmask 255.255.255.224
    broadcast 62.236.92.255
    gateway 62.236.92.225

iface eth0:0 inet static
  
```

```

address 62.236.92.245
netmask 255.255.255.224
broadcast 62.236.92.255
gateway 62.236.92.225

```

Liite 5: Java dummy-paketin luominen

```

shell> su -
shell> aptitude update && aptitude upgrade
shell> aptitude install equivs
shell> aptitude install java-common
shell> mkdir -p /var/install/java/pkg
shell> cd /var/install/java/pkg
shell> cp /usr/share/doc/java-common/dummy-
packages/*.control .
shell> equivs-build java-compiler-dummy.control
shell> equivs-build java-virtual-machine-dummy.control
shell> equivs-build java1-runtime-dummy.control
shell> equivs-build java2-compiler-dummy.control
shell> equivs-build java2-runtime-dummy.control
shell> exit

```

Liite 6: Tomcatin käynnistyskritepin asetukset

```

JAVA_HOME=/usr/local/lib/jdk
CATALINA_HOME=/opt/tomcat
DAEMON_HOME=/opt/tomcat/bin
TOMCAT_USER=tomcat
TMP_DIR=/var/tmp
CATALINA_OPTS="-Djava.awt.headless=true \
-Dfile.encoding=ISO-8859-15 \
-Xms256m -Xmx256m"
CLASSPATH=\
$JAVA_HOME/lib/tools.jar:\
$DAEMON_HOME/dist/commons-daemon.jar:\
$CATALINA_HOME/bin/bootstrap.jar

```

Liite 7: Tomcatin käyttämien porttien konfigurointi

```

<Server port="8005" shutdown="SHUTDOWN" debug="0">
  <Service name="Tomcat-Standalone">
    <Connector class-
Name="org.apache.coyote.tomcat4.CoyoteConnector"
port="80" minProcessors="5" maxProces-
sors="75"
enableLookups="true" redirectPort="443"
acceptCount="100" debug="0" connection-
Timeout="20000"
useURIVValidationHack="false" disable-
UploadTimeout="true" />
    <Connector class-
Name="org.apache.coyote.tomcat4.CoyoteConnector"
port="443" minProcessors="5" maxProces-
sors="75"

```



```

        enableLookups="true"
        acceptCount="100" debug="0"
scheme="https" secure="true"
        useURIValidationHack="false" disable-
UploadTimeout="true">
    <Factory class-
Name="org.apache.coyote.tomcat4.CoyoteServerSocketFactor
y"
        clientAuth="false" protocol="TLS" />
    </Connector>
    <Engine name="Standalone" defaultHost="localhost"
debug="0">

        <Logger class-
Name="org.apache.catalina.logger.FileLogger"
        prefix="catalina_log." suffix=".txt"
        timestamp="true"/>
        <Realm class-
Name="org.apache.catalina.realm.MemoryRealm" />
        <Host name="localhost" debug="0" appBase="webapps"
        unpackWARs="true" autoDeploy="true">
            <Valve class-
Name="org.apache.catalina.valves.AccessLogValve"
            directory="logs" pre-
fix="localhost_access_log." suffix=".txt"
            pattern="common"/>

            <Logger class-
Name="org.apache.catalina.logger.FileLogger"
            directory="logs" pre-
fix="localhost_log." suffix=".txt"
            timestamp="true"/>
        </Host>
    </Engine>
</Service>
</Server>

```

Liite 8: Kannelin konfigurointi

```

group = core
admin-port = 13000
smsbox-port = 13001
admin-password = bar
log-file = "/var/log/kannel/bearerbox.log"
box-deny-ip = "*"
box-allow-ip = "127.0.0.1"
unified-prefix = "00358,0"
access-log = "/var/log/kannel/acces.log"
#store-file = "/var/log/kannel/kannel.store"
store-file = "/tmp/kannel.store"
#ssl-certkey-file = "mycertandprivkeyfile.pem"
dlr-storage = internal

### NOKIA 6210 CONNECTIONS ###

```

```

group = smsc
smc = at
modemtype = nokiaphone
device = /dev/ttyS0
sim-buffering = true
keepalive = 60

### SMSBOX SETUP
group = smsbox
bearerbox-host = localhost
sendsms-port = 13013
global-sender = 13013
sendsms-chars = "0123456789 +-"
log-file = "/var/log/kannel/smsbox.log"

### SEND-SMS USERS -> user and password for SMSEND
cgi-script
group = sendsms-user
username = toyme
password = toyme123

### SERVICES

# Default
group = sms-service
keyword = default
text = "No service specified. Write A if you want to get ID!"

# Read modem definitions
include = "/etc/kannel/modems.conf"

```

Liite 9: Puhelinkohtaiset asetukset

```

group = modems
id = nokiaphone
name = "Nokia Phone"
detect-string = "Nokia Mobile Phone"
need-sleep = true
keepalive-cmd = "AT+CBC;+CSQ"
enable-mms = true
message-storage = "SM"

```

Liite 10: MySQL:n oikeuksien myöntäminen varmuuskopioinnille

```

shell> mysql -u root -p
mysql> GRANT SELECT, LOCK TABLES ON database1.* TO
backup@localhost IDENTIFIED BY 'password';
mysql> \q

```

Liite 11: Automysqlbackupin konfigurointi

```

# Käyttäjätunnus jota käytetään kannan kopiointiin
USERNAME=backup

```

```

...
# Käyttäjätunnuksen salasana
PASSWORD=mypassword
...
# MySQL palvelimen osoite
DBHOST=localhost
...
# Varmistettavien tietokantojen nimet
DBNAMES="database1 database2"
...
# Sijainti johon varmistukset tallennetaan
BACKUPDIR="/backups"
...
# Sähköpostiosoite johon lähetetään raportti varmistuk-
sesta
MAILADDR="webmaster@toymelab.com"
...
# Pakkaustapa (gzip or bzip2)
COMP=bzip2

```

Liite 12: SSH- putkitetut varmuuskopiot

```

#!/bin/bash
DESTSERVER="62.236.92.230"
DESTUSER="jluostar"
DESTDIR="backups"
SRCDIR="/backups"

if [ -d $SRCDIR ] && [ -x $SRCDIR ]
then
    rsync -e ssh -avz $SRCDIR \ $DES-
TUSER@$DESTSERVER:$DESTDIR
    sleep 5
else
    echo "Directory you specified does not exist or you
dont have permission to read it!"
fi

```

Liite 13: Apcupsd:n konfigurointi

```

# General configuration parameters
UPSNAME 62.236.92.246
UPSCABLE usb
UPSTYPE usb
DEVICE /dev/usb/hiddev[0-15]
LOCKFILE /var/lock

# Configuration parameters used during power failures
ONBATTERYDELAY 6
BATTERYLEVEL 5
MINUTES 3
TIMEOUT 0
ANNOY 300

```

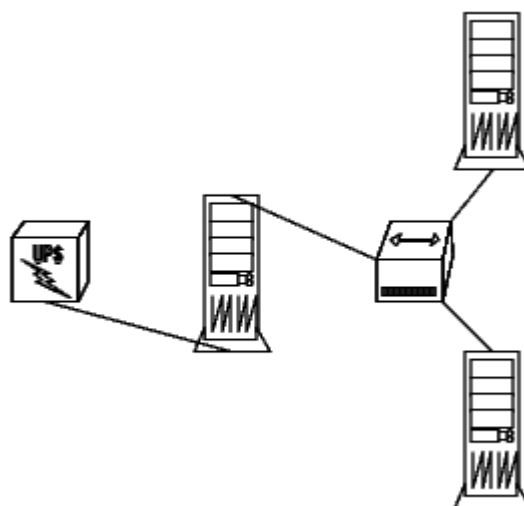
```
ANNOYDELAY 60
NOLOGON disable
KILLDELAY 0

# Configuration statements the network information
server
NETSERVER on
NISIP 0.0.0.0
NISPORT 3551
EVENTSFILE /var/log/apcupsd.events
EVENTSFILEMAX 10

# Configuration statements used if sharing
UPSCCLASS netmaster
UPSMODE net
NETTIME 100
NETPORT 6544
SLAVE 62.236.92.230
SLAVE 62.236.92.248

# Configuration statements to control apcupsd system
logging
STATTIME 0
LOGSTATS off
DATATIME 0
```

Liite 14: UPS netmaster konfiguraatio



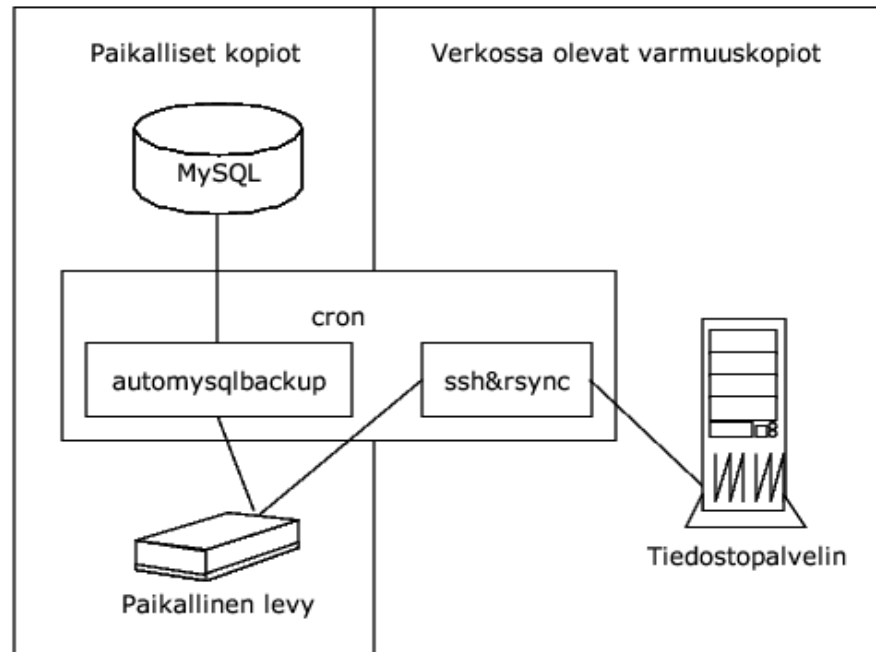
Liite 15: Crontabin ajastuksen muokkaaminen

```

17 * * * root run-parts --report /etc/cron.hourly
25 23 * * * root test -x /usr/sbin/anacron || run-parts --report /etc/cron.daily
47 23 * * 7 root test -x /usr/sbin/anacron || run-parts --report /etc/cron.weekly
52 23 1 * * root test -x /usr/sbin/anacron || run-parts --report /etc/cron.monthly

```

Liite 16: Varmuuskopioiden kulku



Liite 17: Lopputulos

