LAUREA AMMATTIKORKEAKOULU Uuden edellä

Intranet-järjestelmän käyttöönotto tietoturvallisessa GNU/Linux-ympäristössä

Kivinen, Veli-Matti

2013 Leppävaara

Laurea-ammattikorkeakoulu
Leppävaara

Intranet-järjestelmän käyttöönotto tietoturvallisessa GNU/Linux-ympäristössä

Kivinen, Veli-Matti
Tietojenkäsittelyn koulutusohjelma
Opinnäytetyö
Toukokuu, 2013

Laurea-ammattikorkeakoulu **Tiivistelmä**
Laurea Leppävaara
Tietojenkäsittelyn koulutusohjelma

Kivinen, Veli-Matti

Intranet-järjestelmän käyttöönotto tietoturvallisessa GNU/Linux-ympäristössä

Vuosi 2013 Sivumäärä 62

Tämän toiminnallisen opinnäytetyön tavoitteena oli ottaa käyttöön web-teknologioihin perustuva intranet-järjestelmä. Työssä on erityisesti keskitytty varmistamaan intranet-järjestelmän alustana toimivan web-palvelimen ja siihen liittyvien ohjelmistojen tietoturva. Tämän lisäksi tuotettiin englanninkielinen dokumentaatio, jonka avulla intranet-järjestelmän käyttöönottoprosessi voidaan toistaa.

Palvelinympäristön suojaamisen tavoitteena on ensisijaisesti varmistaa kohdeyrityksen tiedon luottamuksellisuuden säilyminen ja intranet-järjestelmän saatavuus.

Työn toimeksiantaja on ulkomaalainen yritys, joka hyödyntää intranet-järjestelmää taloudenhallinnassaan ja sisäisessä viestinnässään. Järjestelmä perustuu PHP-ohjelmointikieleen ja MySQL-tietokantaohjelmistoon.

Opinnäytetyön aihe valittiin tekijän henkilökohtaisten urasuunnitelmien ja mielenkiinnon kohteiden perusteella. Oppimistavoitteina oli kehittyä GNU/Linux-järjestelmien ylläpitotehtävissä ja syventää niihin liittyvää tietoturvaosaamista.

Työn tutkimuksellisessa osuudessa sovellettiin tutkimuksellisen kehitystyön periaatteita ja lähestymistavaksi valittiin konstruktiivinen tutkimus.

Työssä kartoitettiin kohdejärjestelmälle keskeisimmät uhat. Lisäksi selvitettiin menetelmät, joiden avulla näiltä uhilta voi suojautua. Palvelinympäristö suojattiin ulkoisilta uhilta hankitun tietoperustan pohjalta. Opinnäytetyöprosessin aikana intranet-järjestelmä otettiin onnistuneesti käyttöön ja se on kohdeyrityksen aktiivisessa käytössä.

Asiasanat tietoturva, web-palvelin, web-sovellus, GNU/Linux, LAMP

Laurea University of Applied Sciences **Abstract**
Laurea Leppävaara
Bachelor's Degree Programme in Business Information Technology

Kivinen, Veli-Matti

Deployment of an Intranet System in a Secure GNU/Linux Environment

Year 2013 Pages 62

The objective of this practice-based thesis was to deploy a web-based intranet system. The work is focused on ensuring the security of the server platform on which the intranet system is run. In addition, documentation describing the deployment process was produced to enable the repetition of the process.

The objective of securing of the server environment is to ensure the confidentiality of the private information of the client company and the availability of the intranet system.

The client of this project is a foreign company. The intranet system is used for fiscal management and internal communications of the company. The system is based on the PHP programming language and MySQL database software.

The subject of the thesis was chosen based on the personal career plans and points of interest of the author. The learning objectives for this thesis were to develop the author's GNU/Linux administration and information security related skills.

The research is based on the principles of research-development and a constructive research approach.

The most relevant threats and related safeguards for the server environment were researched. The web server was secured against external threats on the basis of the acquired knowledge base. During the process of the thesis the intranet system was successfully deployed and is currently in active use in the client company.

Keywords information security, web server, web application, GNU/Linux, LAMP

Sisällys

1 Johdanto

Tämän toiminnallisen opinnäytetyön aiheena on kohdeyrityksen ydintoimintaa tukevan web-sovelluksen käyttöönotto. Työssä on kiinnitetty erityistä huomiota web-sovelluksen alustana toimivan palvelinympäristön tietoturvaan. Työssä on selvitetty keskeiset kohdejärjestelmään liittyvät tietoturvauhat ja menetelmät, joiden avulla voidaan suojautua näiltä uhilta. Lisäksi tuotettiin dokumentaatio, jonka avulla web-sovelluksen ja palvelinympäristön tietoturvallinen käyttöönotto voidaan tarvittaessa toistaa.

Työ on osa ohjelmistokehitysprojektia, jonka toimeksiantaja on ulkomaalainen yritys. Työssä on keskitytty tämän ohjelmistokehitysprosessin käyttöönotto- ja ylläpitovaiheisiin ja erityisesti ylläpitoon liittyvään tietoturvaan.

Tässä luvussa käydään läpi opinnäytetyön tavoitteet ja kerrotaan työn kannalta olennaista taustatietoa. Luvussa myös rajataan työ ja määritellään keskeiset työhön liittyvät käsitteet.

1.1 Tavoitteet

Työn tavoitteena oli saada toimeksiannettu järjestelmä otettua käyttöön ja luoda englanninkielinen dokumentaatio, jonka avulla käyttöönottoprosessi voidaan toistaa. Dokumentaatiosta on pyritty tekemään joustava; esimerkiksi käyttöjärjestelmän vaihtaminen ei estä web-sovelluksen käyttöönottoa. Palvelimen tasolla on pyritty vaikuttamaan myös itse web-sovelluksen tietoturvaan niin paljon kuin se on mahdollista; muun muassa tietoliikenteen salauksen avulla. Palvelimen suojaamisella pyritään varmistamaan yrityksen tietojen luottamuksellisuuden säilyminen ja varmistamaan web-sovelluksen saatavuus.

Palvelimen käyttöjärjestelmänä toimii avoimen lähdekoodin GNU/Linux. GNU/Linux-järjestelmien määrä yrityksissä lisääntyy nopeammin kuin minkään muun käyttöjärjestelmän (Linux Adoption Trends 2012; Enterprise End User Report 2013) ja erityisesti GNU/Linux-järjestelmien ylläpito-osaamisen tarve tulee kasvamaan ICT-alalla (Linux Jobs Report 2013). Ympäristön valinta on siis perusteltu ja valintaan liittyy myös opinnäytetyön tekijän henkilökohtaiset urasuunnitelmat ja mielenkiinnon kohteet. Käyttöjärjestelmävalinta kattaa myös web-sovelluksen ja toimeksiantajan asettamat vaatimukset ja tarpeet.

Oppimistavoitteita on GNU/Linux-ympäristöjen ylläpitotehtävissä kehittyminen ja näihin ympäristöihin liittyvän tietoturvaosaamisen laajentaminen. Syvällisempää osaamista halutaan myös web-sovelluksen alustana toimivista web-ohjelmistoista.

Työssä on etsitty vastauksia kysymyksiin: Millainen on tietoturvallinen palvelinympäristö? Kuinka voidaan parantaa GNU/Linux-järjestelmän tietoturvaa? Mitkä ovat suurimmat web-palvelimiin kohdistuvat uhat? Millaisia uhkia kohdistuu Apache-, PHP- ja MySQL-ohjelmistoihin? Mitkä näistä uhista ovat kriittisimpiä? Mitkä näistä uhista konkretisoituvat todennäköisimmin? Miten näitä uhkia vastaan voi suojautua? Mitkä ovat suojausmenettelyistä tärkeimpiä?

1.2 Taustaa

Kohdeyrityksen vanha web-pohjainen intranet-järjestelmä oli korvattava uudella monipuolisemmalla ratkaisulla. Yritys tilasi uuden järjestelmän opinnäytetyön tekijältä ja sitä on kehitetty kesäkuusta 2012 lähtien. Järjestelmä valmistui käyttöönotettavaksi tammikuussa 2013. Ennen käyttöönottoa oli kuitenkin perehdyttävä järjestelmän käyttöönottoon ja ylläpitoon liittyviin riskeihin ja uhkiin. Intranet-järjestelmä on web-pohjainen, PHP-ohjelmointikieleen ja MySQL-tietokantaohjelmistoon perustuva intranet-järjestelmä, jota käytetään yrityksen taloudenhallinnassa ja sisäisessä viestinnässä.

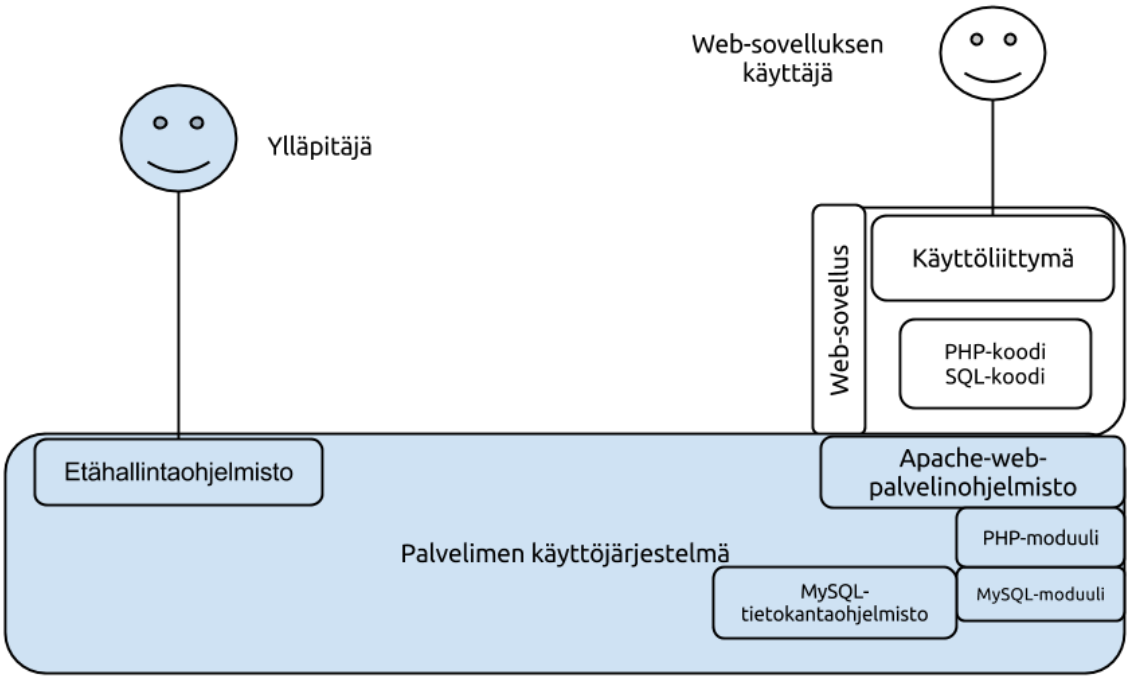
Työn toimeksiantajan määritteleminen ei ole yksinkertaista, sillä järjestelmän käyttöönotto on vain osa sitä kokonaisuutta, jonka kohdeyritys tilasi. Intranet-järjestelmä tilattiin siis täydellisenä, eikä pelkästään palvelimen asennusta, intranet-järjestelmän käyttöönottoa ja siihen liittyvän dokumentaation tuottamista. Tästä syystä opinnäytetyön tekijää voidaan pitää osittain työn toimeksiantajana ja hyödyntäjänä, sillä intranet-järjestelmän käyttöönottoprosessi on osa ohjelmistokehitysprosessia. Opinnäytetyön aiheeksi olisi voinut valita intranet-järjestelmän ohjelmistokehitysprosessin käyttöönottoa myöten, mutta tällöin työn aiheesta olisi tullut liian laaja. Opinnäytetyön tekniset valinnat on tehty melko pitkälti ohjelmistokehitysprojektin vaatimusten puitteissa.

Työn toteuttaminen vaati kokemusta GNU/Linux-käyttöjärjestelmästä ja tietoturvaan liittyvää osaamista, jotta työn vaatimaa työmäärää voitaisiin pitää kohtuullisena suhteutettuna opinnäytetyöopintojakson opintopisteiden määrään. Ennen työn aloittamista koettiin, että opinnäytetyön tekijällä oli valmiudet ja riittävä osaaminen työn toteuttamiselle.

Kohdeyritys on maksanut työhön liittyvät kulut, jotka koostuvat lähinnä palvelimen vuokrauskuluista. Työ on tehty teknisestä näkökulmasta melko itsenäisesti, sillä yrityksessä ei ole työn arviointiin vaadittavaa spesifistä asiantuntijuutta. Työn arviointi vaatii ymmärrystä GNU/Linux-järjestelmistä ja erityisesti niihin liittyvästä tietoturvasta.

1.3 Työn rajaus

Työ käsittelee erityisesti tietojärjestelmien tietoturvallisuuden teknistä puolta. Työssä kartoitetaan web-sovelluksen ylläpitämiseen GNU/Linux-ympäristössä liittyvät uhat ja niihin varautuminen GNU/Linux-käyttöjärjestelmässä. Intranet-järjestelmän ohjelmistohaavoittuvuuksiin on varauduttu järjestelmän ohjelmistokehitysvaiheessa, ja koska työn pääpaino on haluttu pitää palvelinympäristöjen ylläpitoon liittyvässä tietoturvassa, on web-sovelluksien haavoittuvuuksiin perustuvat uhat rajattu ulos työstä. Tavoitteena on parantaa web-sovelluksen tietoturvaa palvelimen tasolla, mutta ei web-sovelluksen koodin tasolla. Kuviossa 1 on havainnollistettu työn rajaus. Valkoinen alue on rajattu ulos työstä.



Kuvio 1: Työn rajaus.

Uhkien kartoituksessa ja niihin varautumisessa keskitytään verkosta kohdistuviin hyökkäyksiin. Pääpaino on ollut varautua satunnaisten ja tuntemattomien tahojen tekemiä verkkohyökkäyksiä vastaan, sillä niiden on koettu olevan suurin riski kohdejärjestelmän kaltaisille kohteille (IT-Grundschutz Kataloge 2005, 231).

Henkilöstöön ja organisaatioon liittyvät tietoturvanäkökulmat on jätetty huomioimatta, sillä intranet-järjestelmän käyttäjät eivät ole suorassa vuorovaikutuksessa palvelimen tietoturvan kanssa, vaan ainoastaan käyttöönotettavan web-sovelluksen. Työstä on rajattu ulos kaikki palvelimen fyysiseen turvallisuuteen liittyvät seikat, sillä palvelin ei sijaitse kohdeyrityksen toimitiloissa, mikä tekee tästä näkökulmasta hallitsemattoman. Myös verkkolaitteisiin ja -topologiaan liittyvät tietoturvanäkökulmat on rajattu ulos edellä mainitusta syystä.

GNU/Linux valittiin palvelimen käyttöjärjestelmäksi kasvavan suosionsa ja kustannustehokkuutensa vuoksi. GNU/Linux-jakelu ja web-palvelinohjelmisto on valittu perustellusti luvussa 3.1.1 ja tietoturvalla oli valinnassa suuri painoarvo.

1.4 Keskeiset käsitteet

Apache HTTPD Server	Avoimen lähdekoodin HTTP-palvelinohjelmisto, jota käytetään web-sivujen välittämisessä (About the Apache HTTP Server project 2012).
Dedikoitu palvelin	Fyysinen palvelin, jota ei ole jaettu muiden tahojen käyttöön (vrt. web-hotellipalvelu). Ympäristö on kokonaisvaltaisemmin hallittavissa kuin jaettu ympäristö.
GNU/Linux	Avoimeen lähdekoodin perustuva Unixin kaltainen käyttöjärjestelmä. Usein käytetään myös nimitystä Linux, jonka kuitenkin voidaan tulkita viittaavan ainoastaan GNU/Linux-käyttöjärjestelmän ytimeen (*kernel*). (Stallman 2013.)
HTML	Kuvauskieli, jota käytetään WWW-sivujen esittämiseen web-selaimessa (Nixon 2012, 1).
HTTP	HTTP eli Hypertext Transfer Protocol on internet-protokolla, jolla siirretään erityisesti HTML-, eli web-sivuja (Nixon 2012, 1).
HTTPS	HTTPS eli Hypertext Transfer Protocol Secure on internet-protokolla, joka kuljettaa web-liikenteen salatussa muodossa **(**IT-Grundschutz Kataloge 2005, 230).
Hyökkäyspinta-ala	Ne tekijät, jotka mahdollistavat tietoturvahyökkäyksen tai helpottavat sen onnistumista, muodostavat hyökkäyspinta-alan. Näitä tekijöitä ovat esimerkiksi kuuntelevat taustaprosessit sekä ohjelmointi- ja konfiguraatiovirheet. (Manadhata & Wing 2004, 1.)
Kuunteleva taustaprosessi	Järjestelmän taustalla ajettava prosessi, joka käsittelee verkosta saapuvaa liikennettä. (*Listening service/daemon*).

Mandatory Access Control	Tietoturvajärjestelmä, jolla rajoitetaan prosessien oikeuksia järjestelmän tiedostoihin ja toisiin prosesseihin (Bauer 2006).
MySQL	Avoimen lähdekoodin relaatiotietokantaohjelmisto, jota käytetään laajalti osana web-sovelluksia (MySQL 2005, 1).
Nollapäivähyökkäys	Tietoturvahyökkäys, joka kohdistuu haavoittuvuuteen, johon ei ole saatavilla korjaavaa päivitystä (Bauer 2007).
Pakettisuodatus	IP-pakettien suodattaminen otsaketietojen, esimerkiksi kohdeportin tai lähettäjän tai vastaanottajan IP-osoitteen perusteella. Pakettisuodatus voidaan tehdä myös TCP-istuntojen perusteella, jolloin puhutaan tilapohjaisesta pakettisuodattamisesta (*stateful packet-filtering*). (Bauer 2003; Garfinkel, Schwartz & Spafford 2003.)
Palomuuri	Järjestelmä, jonka avulla voidaan suorittaa jonkintasoista verkon pääsynhallintaa muun muassa pakettisuodatuksen avulla. Palomuurilla voidaan viitata ohjelmaan tai fyysiseen laitteeseen. (Bauer 2003.)
PHP	Avoimen lähdekoodin ohjelmointikieli ja -ympäristö, jonka avulla voidaan luoda dynaamisia web-sovelluksia.
Root-käyttäjä	Unixin ja sen kaltaisten käyttöjärjestelmien järjestelmänvalvojan käyttäjätunnus, jolla on täydet käyttöoikeudet järjestelmään.
Salaus	Tiedon muuttaminen sellaiseen muotoon, ettei ulkopuolinen pysty sitä tulkitsemaan. Salaus puretaan avaimella, joka voi olla esimerkiksi salasana.
SSH	Internet-protokolla sekä salaus- ja etähallintaohjelma, jota käytetään erityisesti salattujen etäyhteyksien muodostamiseen (Dwivedi 2004, 3).
Varmuuskopiointi	Tiedon kopiointi ja säilöminen, jonka tavoitteena on estää tiedon tuhoutuminen ja varmistaa tiedon eheyden säilyminen (IT-Grundschutz Kataloge 2005, 38).

Verkkolinnake	*Bastion host*. Verkon haavoittuvaan kohtaan sijoitettu laite, joka on kovetettu hyökkäysten varalta. Esimerkiksi web-palvelin. (Valtionhallinnon tietoturvasanasto 2008, 133.)
Web-ohjelmisto	Tässä työssä web-ohjelmistolla viitataan niihin ohjelmistoihin, jotka toimivat web-sovelluksen alustana ja muodostavat käyttöjärjestelmän kanssa niin kutsutun ratkaisupinon (*solution stack*).
Web-sovellus	Tässä työssä web-sovelluksella viitataan pääosin PHP-, SQL-, HTML-, CSS- ja JavaScript-koodiin perustuvaan intranet-järjestelmään, joka ajetaan web-ohjelmistopinon päällä.

2 Menetelmät

Tässä luvussa käydään läpi opinnäytetyöprosessin eteneminen sekä tutkimukselliset menetelmät, joita on käytetty työn toteuttamisessa.

2.1 Opinnäytetyöprosessi

Opinnäytetyön tekeminen alkoi toiminnallisen opinnäytetyön työstöprosessiin tutustumisella. Lähteenä käytettiin Vilkan ja Airaksisen teosta *Toiminnallinen opinnäytetyö*. Tätä teosta käytettiin tiedonlähteenä myös pitkin koko opinnäytetyöprosessia. Opinnäytetyöprosessiin tutustumisen jälkeen tutkittiin erilaisia kehittämistyön menetelmiä. Työn toteuttamisessa päädyttiin soveltamaan tutkimuksellisen kehittämisen periaatteita.

Tutkimuksellisen kehittämisen tavoite on ennen kaikkea ratkaista käytännön ongelmia johdonmukaisesti valittua teoreettista tietoa hyödyntäen. Tavoitteena on käytännön ongelmien ratkaisemisen lisäksi luoda myös uutta tietoa tiedeyhteisöön. Tutkimuksellinen kehittämisen periaatteet soveltuivat opinnäytetyöhön, sillä työn tavoitteena oli ratkaista käytännön ongelma. Koska opinnäytetyö on toimintakeskeinen, sen ei tarvitse nojautua yhtä vahvasti teoriaan kuin tutkimuksellisen työn (Eskola & Suoranta 1996, ks. Vilkka & Airaksinen 2003, 57). (Ojasalo, Moilanen & Ritalahti 2009, 17-20.)

Kehittämistyön prosessi alkoi opinnäytetyön aiheen, eli kehittämiskohteen, valitsemisella. Opinnäytetyön aiheelle on tärkeää, että se on tekijää motivoiva ja tekijä kokee voivansa syventää asiantuntemustaan aihealueesta. On myös suositeltavaa valita aihealue henkilökohtaisten urasuunnitelmien perusteella. Nämä kriteerit täyttyivät opinnäytetyön aiheen osalta. Valitun aiheen pohjalta kirjoitettiin aiheanalyysi, jossa ideoitiin työn alustavia tavoitteita. (Vilkka & Airaksinen 2003, 23-24.)

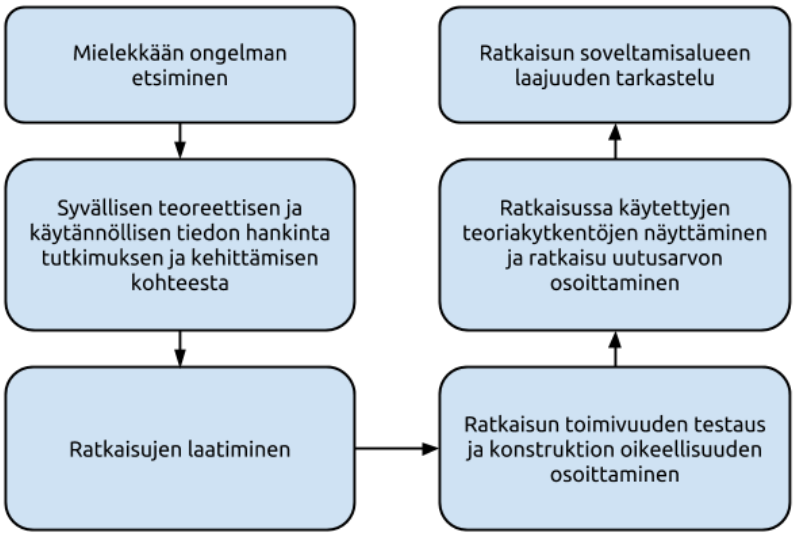
Aiheanalyysin pohjalta laadittiin toimintasuunnitelma, jossa määritettiin työn tavoitteita yksityiskohtaisemmin ja rajattiin työn laajuus. Toimintasuunnitelmassa kartoitettiin myös tilanteen lähtökohdat ja arvioitiin työn toteuttamiseen vaadittavaa osaamista. Työn perusteellinen suunnittelu ja suunnitelman seuraaminen on kehitysprosessin kannalta tärkeää (Ojasalo ym. 2009, 20).

2.2 Kehittämistyön lähestymistavat

Kehittämistyölle oli valittava sille parhaiten soveltuva lähestymistapa. Lähestymistavan määrittää melko pitkälti työlle asetetut tavoitteet. Lähestymistavoista perehdyttiin tapaus- ja toimintatutkimukseen, konstruktiiviseen tutkimukseen sekä innovaatioiden tuottamiseen. Tapaustutkimus ei yleensä sovellu kehitystyöhön, jossa tehdään konkreettinen tuotos. Tapaustutkimuksen avulla pyritään yksityiskohtaisesti ymmärtämään kehittämiskohteen tilannetta. Sen avulla tuotetaan lähinnä erilaisia kehitysehdotuksia. Toimintatutkimus on taas hyvin ongelmakeskeinen lähestymistapa, jota käytetään erityisesti kun tavoitteena on organisaation tai ihmisten toiminnan muuttaminen. Tämänkaltainen lähestymistapa ei sopinut työlle, sillä tavoitteet liittyivät erityisesti teknisiin näkökulmiin. (2009, 36-38; 52; 58.)

Konstruktiivinen lähestymistapa osoittautui opinnäytetyölle sopivimmaksi vaihtoehdoksi. Konstruktiivinen lähestymistapa sopii kehittämistöihin, joiden tavoitteena on ratkaista käytännön ongelma konkreettisen tuotoksen avulla ja olemassa olevaa tietoa hyödyntäen. Tämä tuotos voi olla esimerkiksi uusi tietojärjestelmä, www-sivusto tai käyttöohjeistus. Tässä opinnäytetyön tämä tuotos on intranet-järjestelmän käyttöönottodokumentaatio. (2009, 37-38; 65.)

Konstruktiivinen lähestymistapa eroaa innovaatioiden tuottamisesta siinä, että sen tuotos ei yleensä ole innovaatio, vaan ratkaisu, joka perustuu olemassa olevan tiedon avulla uudenlaisen todellisuuden rakentamiseen. Innovaatioiden tuottamisessa tavoitteena on luoda jotain täysin uutta, eikä se sen takia soveltunut kehittämistyön lähestymistavaksi. Kuviossa 2 kuvataan konstruktiivisen tutkimuksen prosessi. (2009, 39; 65.)



Kuvio 2: Konstruktiivisen tutkimuksen prosessi (Kasanen, Lukka & Siitonen 1991, ks. Ojasalo ym. 2009).

Tavoitteena oli luoda kohdeorganisaation liiketoimintaa tukeva ratkaisu käytännön ongelmaan käyttämällä olemassa olevaa aineistoa. Tarkoituksena oli saada aikaan tuotos, jota voitaisiin hyödyntää myös kohdeorganisaation ulkopuolella osana vastaavanlaista projektia. Tavoitteiden määrittämisen jälkeen kehittämiskohde rajattiin. Rajaamisessa hyödynnettiin luvussa 2.3 läpikäytyjä viitekehyksiä.

Opinnäytetyöprosessi dokumentoitiin aiheanalyysin ja toimintasuunnitelman lisäksi muun muassa päiväkirjan muodossa. Dokumentaatio sisältää tietoa opinnäytetyön selvitysosuuden etenemisestä, aikataulusuunnitelmista, oppimistapahtumista ja ohjauskeskusteluista. Ohjauskeskusteluihin valmistauduttiin laatimalla dokumentaatioon opinnäytetyöhön ja -prosessiin liittyviä kysymyksiä.

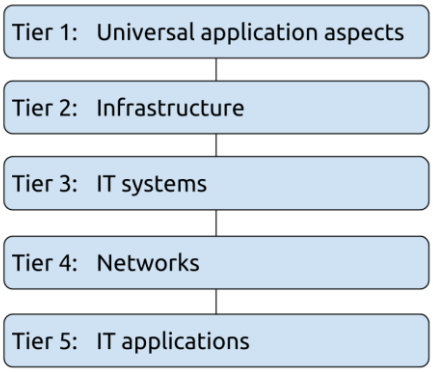
2.3 Tiedonhankinta

Opinnäytetyön tietoperusta muodostettiin tässä luvussa kuvatulla menetelmällä. Tämän tietoperustan avulla luotiin luku 3 ja opinnäytetyön tuotos.

Lähdekirjallisuuden etsimisessä hyödynnettiin Viestintäviraston laatimaa *Kansallista turvallisuusauditointikriiteristö KATAKRI:a* ja *Bundesamt für Sicherheit in der Informationstechnik*in (BSI) laatimaa, tietoturvaa käsittelevää *IT-Grundschutz*-dokumentaatiota. BSI on Saksassa toimiva, Suomen Viestintävirastoa vastaava elin. Näiden viitekehysten avulla pystyttiin pääpiirteittäin määrittämään työlle oleellisimmat tietoturvanäkökulmat ja niitä hyödynnettiin työn rajaamistyössä.

KATAKRI koostuu neljästä osa-alueesta; hallinnollinen turvallisuus, henkilöstöturvallisuus, fyysinen turvallisuus ja tietoturvallisuus. Neljästä osa-alueesta keskityttiin ainoastaan tietoturvallisuus-osioon, sillä se koettiin osa-alueista selvästi olennaisimmaksi ja kaikki muut osa-alueet rajattiin työstä.

Työn rajauksessa käytettiin myös *IT-Grundschutz*-dokumentaatiossa kuvattua näkökulmien jaottelua (kuvio 3). Jaottelussa tietoturvanäkökulmat on jaettu viiteen aihealueeseen. Näistä olennaisimmiksi osoittautuivat aihealueiden 3 ja 5 sisältämät näkökulmat, sillä ne sisältävät tietoa erityisesti palvelimiin kohdistuvista uhista ja keskeisimmistä suojausmenettelyistä. Muut aihealueet sisältävät lähinnä organisaatioon, verkon topologiaan ja fyysiseen turvallisuuteen liittyviä näkökulmia. Nämä näkökulmat eivät joko ole hallittavissa tai oleellisia työn kannalta ja ovat sen takia rajattu ulos.



Kuvio 3: Tietoturvanäkökulmien jaottelu (IT-Grundschutz Kataloge 2005, 26).

IT-Grundschutz- ja *KATAKRI*-viitekehystä käytettiin myös teknisten suositeltujen tietoturvakäytäntöjen selvittämiseen. Näiden viitekehysten avulla löydettiin osa tärkeistä suojausmenettelyistä, kuten asianmukaisen varmuuskopioinnin järjestäminen. Suojausmenettelyiden määrittelyt eivät tosin aina olleet riittävän konkreettisia ja kohdejärjestelmälle spesifisiä. Ennen kaikkea nämä viitekehykset kuitenkin mahdollistivat johdonmukaisen lähdemateriaalin löytämisen.

Lähdemateriaalina käytettiin erityisesti GNU/Linux- ja tietoturva-aiheisia kirjoja ja järjestelmässä käytettyjen ohjelmistojen virallista dokumentaatiota. Lisäksi Theseus-opinnäytetyötietokannan samoja aiheita käsitteleviin teoksiin tutustuttiin. Paikoin myös internetin keskustelupalstoja käytettiin kirjallisuudessa esiintyneen tiedon ajantasaisuuden varmistamiseen. Prosessin aikana konsultoitiin myös aihealueeseen perehtyneitä ICT-alan asiantuntijoilta.

Lähdemateriaali käytiin läpi ja siitä poimittiin työhön liittyvä, erityisesti siis web-ohjelmistojen ja käyttöjärjestelmän tietoturvaan liittyvä tieto. Erityisesti keskityttiin sellai-

seen tietoon, jota esiintyi useassa eri teoksessa. Tiedon avulla muodostettiin luvun 3 teoria, jonka avulla taas luotiin työn produkti, intranet-järjestelmän käyttöönottodokumentaatio.

Terminologian käännöstyössä on käytetty *Valtiohallinnon tietoturvasanasto VAHTI*a.

3 Teoria

Tämä luku luo kokonaisuuden, jonka pohjalta ratkaisut palvelinympäristön suojaamiseksi ulkoisilta uhilta on laadittu. Järjestelmän käyttöönottodokumentaatio on tuotettu tämän tiedon perusteella.

Luvussa 3.1 käydään läpi yleistä tietoa palvelimen käyttöjärjestelmästä, web-ohjelmistoista ja järjestelmän hallinnasta. Luvussa 3.2 käydään läpi järjestelmään kohdistuvat uhat ja luvussa 3.3 käsitellään suojausmenettelyt, joita on käytetty uhilta suojautumiseen.

3.1 Palvelinympäristö

Tässä luvussa kerrotaan käyttöjärjestelmän valinnasta sekä yleistä tietoa käytettävistä web-ohjelmistoista ja käydään läpi menetelmät ja työkalut, joilla järjestelmää hallitaan. Luvussa 3.1.3 käydään läpi web-sovelluksen alustana toimivat välttämättömät ohjelmistot. Käyttöjärjestelmän valinnassa tietoturvalla on ollut suuri painoarvo.

Intranet-järjestelmää varten on vuokrattu dedikoitu palvelin (*dedicated server*). Dedikoitu palvelin valittiin web-hotellipalvelun sijaan kustannustehokkuutensa vuoksi. Moni web-hotellipalveluntarjoaja veloittaa esimerkiksi web-liikenteen salauksesta huomattavia summia, mutta dedikoidulla palvelimella liikenteen salauksen voi toteuttaa ilman lisäkustannuksia. Dedikoitu palvelin mahdollistaa myös käyttöjärjestelmän täysivaltaisen hallinnan, minkä johdosta voidaan myös hallita tietoturvaa kokonaisvaltaisemmin. On otettava myös huomioon, että nämä suuremmat mahdollisuudet tarkoittavat myös suurempaa vastuuta.

3.1.1 Käyttöjärjestelmän valinta

Käyttöjärjestelmäehdokkaiksi valittiin ne GNU/Linux-jakelut, joilla oli web-palvelimista suurimmat markkinaosuudet. Suosituimmat GNU/Linux-jakelut vuonna 2012 olivat Debian, CentOS ja Ubuntu (Gelbmann 2012). Suosituilla GNU/Linux-jakeluilla on yleensä myös aktiiviset käyttäjäyhteisöt, joiden kautta on mahdollista saada tukea. Ubuntun kehityksestä vastaavan yrityksen, Canonicalin, kautta on mahdollista saada myös maksullista tukea.

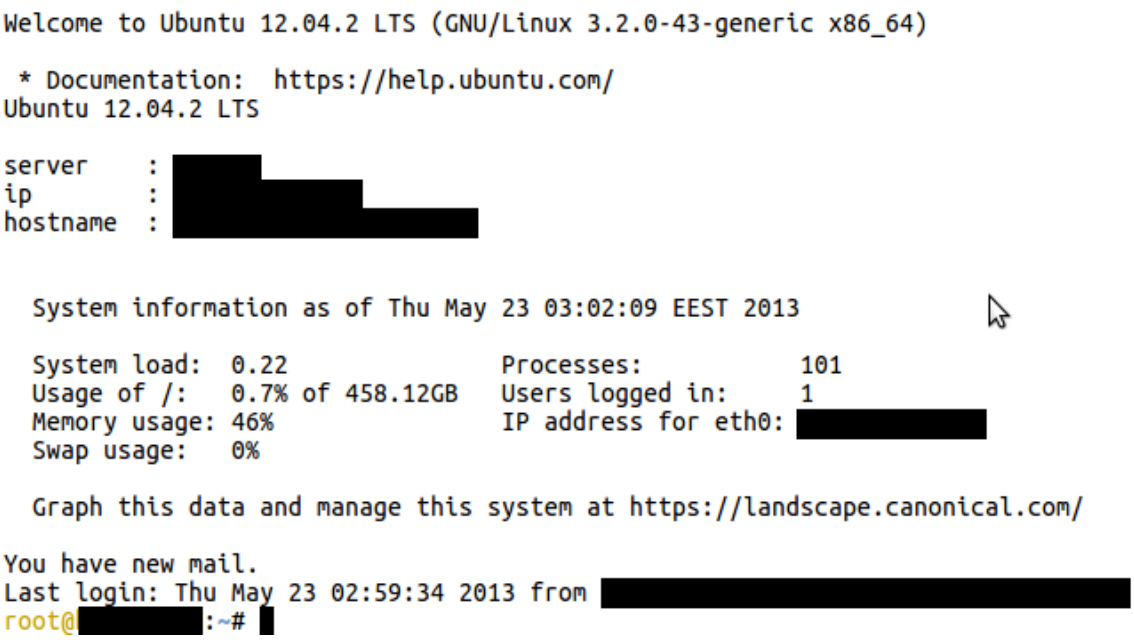
Käyttöjärjestelmän tukiajalla tarkoitetaan ajanjaksoa, jonka aikana käyttöjärjestelmä saa päivityksiä. Käyttöä on mahdollista jatkaa tämän ajan jälkeenkin, mutta se on erittäin riskialtista, sillä järjestelmä ei saa tietoturvapäivityksiä laisinkaan. Ubuntu-käyttöjärjestelmästä julkaistaan joka toinen vuosi Long Term Support –versio, jota Canonical tukee 5 vuotta (Download Ubuntu Server 2013). CentOS:n tukiaika on noin 7-10 vuotta (The CentOS Project) ja Debianille ei ole määritetty tukiaikaa, mutta tukiaika on ollut yleensä noin kolme vuotta (Debian Releases 2013).

Ubuntuun on oletusarvoisesti asennettu AppArmor-niminen Mandatory Access Control –järjestelmä, jolla voidaan parantaa käyttöjärjestelmän tietoturvaa (Ubuntu Official Documentation: AppArmor). Tämän työn kirjoitushetkellä Debianin mukana ei tullut AppArmorin vaatimia kernel-moduuleita. CentOS:ään on myös oletusarvoisesti asennettu MAC-järjestelmä, SELinux. SELinuxia pidetään kuitenkin AppArmoria huomattavasti monimutkaisempana konfiguroida (Bauer 2006).

Palvelimen käyttöjärjestelmäksi valittiin Ubuntu Server 12.04 LTS. Valinta perustuu käyttöjärjestelmän tietoturvaominaisuuksiin, tuen kestoon, markkinaosuuteen ja opinnäytetyön tekijän henkilökohtaisiin valmiuksiin. Henkilökohtaista osaamista löytyi ennestään lähinnä Debian GNU/Linux -käyttöjärjestelmästä sekä sen johdannaisista, joten oli luontevaa valita Debianiin perustuva käyttöjärjestelmä kuten Ubuntu. Ubuntu valittiin Debianin sijaan pidemmän tukiaikansa ja AppArmor-tukensa vuoksi. CentOS:n sijaan Ubuntu valittiin taas oletusarvoisen MAC-järjestelmän helppokäyttöisyyden ja opinnäytetyöntekijän olemassa olevan osaamisen takia.

3.1.2 Palvelimen hallinta

Palvelimelle ei asennettu graafista käyttöliittymää, joten sitä hallitaan komentorivirajapinnassa SSH-yhteyden välityksellä. Graafinen käyttöliittymä lisää hyökkäyspinta-alaa ja onnistunut hyökkäys käyttöliittymän ikkunointijärjestelmää vastaan antaa hyökkääjälle käytännössä rajattoman pääsyn järjestelmään (McClure, Scambray & Kurtz 2009, 291). Tästä syystä koettiin, ettei graafista käyttöliittymää ollut syytä asentaa. Palvelimelle kirjautuessaan ylläpitäjä saa yleistietoa järjestelmän tilasta (kuva 1). Palvelimen yksilöivät tiedot on sensuroitu kuvankaappauksesta.



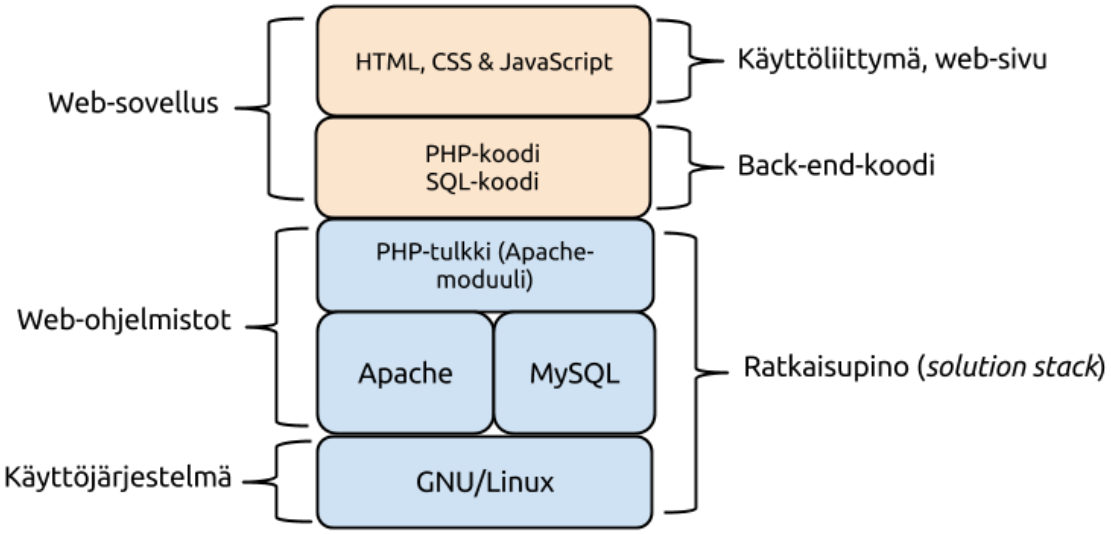
Kuva 1: Komentorivirajapinta

SSH on turvallisempi ratkaisu etähallinnan toteuttamiseen kuin esimerkiksi Telnet tai Remote Shell (*rsh*), sillä tietoliikenne palvelimen ja SSH-asiakasohjelman välillä on aina salattu. Esimerkiksi *Kansallisen turvallisuusauditointikriteeristön* määrittämät vaatimukset edellyttävät kaiken hallintaliikenteen salausta (KATAKRI 2011, 80). (Stanger, Lane & Danielyan 2001, 361; Koski 2010, 499; Turnbull 2005, 169.)

SSH-protokollaa voidaan hyväksikäyttää myös tiedostojen siirron salauksessa ja tietoliikenteen tunneloinnissa. SSH-palvelinohjelmistoksi valittiin OpenSSH, koska se on oletusarvoisesti asennettu Ubuntu Server -käyttöjärjestelmässä. (Stanger ym. 2001, 362; Dwivedi 2004, 3; 10.)

3.1.3 Palvelimella ajettavat web-ohjelmistot

Web-sovellus vaatii toimiakseen alustan, niin kutsutun ratkaisupinon (*solution stack*). Tässä työssä ratkaisupino koostuu GNU/Linux-käyttöjärjestelmästä, Apache-web-palvelinohjelmistosta, MySQL-tietokantaohjelmistosta ja PHP-tulkista (kuvio 4). Tähän kokonaisuuteen viitataan usein lyhenteellä LAMP.



Kuvio 4: Järjestelmäkokonaisuus

Apache on internetin ylivoimaisesti suosituin web-palvelinohjelmisto (Usage statistics and market share of Apache for websites 2013; McClure ym. 2009, 272; April 2013 Web Server Survey 2013). Apache valittiin markkinaosuutensa ja saatavilla olevan AppArmor-profiilin vuoksi. AppArmor-profiili mahdollistaa Apachen eristämisen muusta järjestelmästä (katso luku 3.3.5). Apachesta oli myös opinnäytetyön tekijällä henkilökohtaista kokemusta. Web-palvelinohjelmistoksi harkittiin Nginx:ää ja Lighttpd:tä, mutta kumpaankaan ei ollut saatavilla AppArmor-profiilia, eikä ohjelmistoissa koettu olevan juurikaan etuja Apacheen verrattuna.

Apachelle on saatavilla runsaasti moduuleja, joiden avulla sen ominaisuuksia pystyy laajentamaan (Koski 2010, 426). Tässä työssä käsitellään moduulien mod_php5, mod_ssl, mod_rewrite, mod_security, mod_evasive, mod_headers ja mod_apparmor toiminta.

PHP-koodin voi ajaa järjestelmässä usealla eri tavalla. Kohdejärjestelmässä koodi ajetaan Apachen PHP-moduulin avulla. Apache-moduulin käyttö on helppokäyttöinen ratkaisu koodin ajamiselle. Varteenotettava vaihtoehto olisi ajaa koodi PHP-FPM-toteuksella. PHP-FPM:n etuja ovat muun muassa sen nopeus ja tehokkuus, mutta niiden ei koeta olevan erityisen kriittisiä ominaisuuksia kohdejärjestelmälle.

Intranet-järjestelmän tietokantaohjelmistona toimii MySQL. Tietokantaohjelmiston valinta tapahtui jo järjestelmän ohjelmistokehitysprosessin alkuvaiheessa, joten tietokantaohjelmiston valintaprosessia ei tässä opinnäytetyössä käydä läpi. Käytännössä olisi kuitenkin mahdollista käyttää MySQL:n kanssa API-yhteensopivaa (*application programming interface* eli ohjelmointirajapinta) tietokantaohjelmistoa, kuten MariaDB:tä, mutta tätä ei nähty tarpeelliseksi. Kirjoitushetkellä näyttää kuitenkin siltä, että MariaDB tulee tulevaisuudessa syrjäyttämään MySQL:n standardina avoimen lähdekoodin tietokantaratkaisuna.

3.2 Ympäristöön kohdistuvat uhat

Tässä luvussa käydään läpi keskeisimmät web-palvelimeen kohdistuvat uhat. Lisäksi käydään läpi ne hyökkäykset, joita todennäköisimmin tulee kohdistumaan järjestelmään. Näiden uhkien kartoittaminen ja analysoiminen on erittäin tärkeä osa järjestelmän suojaamista. Kaikkia uhkia ei ole mahdollista ennakoida, mutta suurimman riskin muodostavat uhat on syytä selvittää, jotta niitä varten voisi myös varautua. (Bauer 2003.)

Kaikkiin web-sovellukseen liittyviin uhkiin ei voi varautua sovelluksen ohjelmistokehitysvaiheessa. Vaikka itse web-sovelluksessa ei olisi haavoittuvuuksia, voi palvelimen web-sovelluksen toimintaan vaadittavat ohjelmistot olla haavoittuvia. Web-palvelimen haavoittuvuudet vaikuttavat olennaisesti myös web-sovelluksen tietoturvaan. (Hacking Exposed Linux 2008, 382; IT-Grundschutz Kataloge 2005, 231.)

3.2.1 Erilaiset tietoturvahyökkäykset

Suurin osa tässä työssä käsiteltävistä ja kohdejärjestelmään kohdistuvista uhista on verkkohyökkäyksiä. Useimmat verkkohyökkäykset ovat automatisoituja, tiettyyn tietoturvahaavoittuvuuteen perustuvia, niin kutsuttuja purkitettuja hyökkäyksiä (*canned exploit*). Purkitetut hyökkäykset ovat yleensä datapohjaisia hyökkäyksiä (*data-driven attack*), jotka ovat kaikista hyökkäyksistä yleisimpiä (McClure ym. 2009, 227). Datapohjainen hyökkäys perustuu tiedon lähettämiseen haavoittuvalle kuuntelevalle taustaprosessille, mikä saa prosessin käyttäytymään asiaankuulumattomalla tavalla. Tavoitteena tällaisella hyökkäyksellä on usein kaapata järjestelmä. (McClure ym. 2009, 231.)

Internetiin kytkettyihin palvelimiin kohdistuu usein satunnaisesti suoritettuja haavoittuvuusskannauksia, joilla yritetään löytää haavoittuvia ohjelmia. Mikäli palvelimesta paljastuu haavoittuvuus, käynnistyy usein myös automaattinen hyökkäys. Koska kohdeyritystä ei voida pitää niin kutsuttuna korkean profiilin kohteena, juuri nämä automatisoidut hyökkäykset koetaan suurimmaksi uhaksi kohdejärjestelmälle.

Nollapäivähyökkäyksellä tarkoitetaan hyökkäystä ohjelmiston haavoittuvuuteen, johon ei ole saatavilla korjaavaa päivitystä. Erityisen vaarallisia ovat nollapäivähaavoittuvuuksiin perustuvat hyökkäykset, sillä niihin on erittäin vaikea varautua. (Bauer 2003.)

GNU/Linux-järjestelmiin kohdistuvat tietoturvahyökkäykset voidaan jakaa paikallis- ja etähyökkäyksiin. Etähyökkäyksellä tarkoitetaan hyökkäystä, jonka käytännössä voi tehdä kuka tahansa internetin välityksellä ilman erityisiä oikeuksia järjestelmään. Paikalliset hyökkäykset taas edellyttävät pääsyä järjestelmän komentorivirajapintaan. Koska kenellekään ei ylläpitäjän lisäksi ole pääsyä käyttöjärjestelmän komentorivirajapintaan, paikalliset hyökkäykset on

rajattu ulos tästä työstä. Paikalliset hyökkäykset ovat kuitenkin mahdollisia, mikäli hyökkääjä saa pääsyn komentorivirajapintaan onnistuneen etähyökkäyksen johdosta. Ensisijaiseksi tavoitteeksi on siis otettu etähyökkäyksiltä suojautuminen. (McClure ym. 2009, 225-226.)

Yleisiä motiiveja automatisoiduille verkkohyökkäyksille on muun muassa pyrkiä tekemään kohteesta niin sanottu sillanpääasema, jonka kautta hyökkääjä voi reitittää omaa verkkoliikennettään tai suorittaa esimerkiksi palvelunestohyökkäyksiä (*denial-of-service attack*) muihin järjestelmiin. Murrettuja järjestelmiä käytetään myös muun muassa laittomien tiedostojen jakamiseen ja yrityksen luottamuksellisen tiedon varastamiseen. (Gagné 2003, 447; Bauer 2003.)

3.2.2 Konfiguraatiovirheisiin perustuvat uhat

Konfiguraatiovirheet ohjelmistoissa ja käyttöjärjestelmässä voivat aiheuttaa suuria uhkia järjestelmälle. Väärin konfiguroituna esimerkiksi *Network File System* –verkkolevyjärjestelmä voi johtaa organisaation kaiken luottamuksellisen tiedon vuotamiseen julkisesti koko internetin saataville. Tällainen virhe voi tehdä myös järjestelmän kaappaamisesta hyvin yksinkertaista. Myös web-palvelinohjelmisto voi vuotaa luottamuksellista tietoa, mikäli se on konfiguroitu huolimattomasti. (McClure ym. 2009, 257.)

Oletusasetuksilla Apache kertoo käyttäjälle hyvin paljon järjestelmän toiminnasta. Esimerkiksi käyttöjärjestelmän ja käytössä olevien moduulien versiotiedot paljastetaan vierailijalle. Näiden tietojen paljastaminen voi altistaa tietoturvahyökkäyksille, jos palvelin paljastaa esimerkiksi haavoittuvan ohjelmiston versionumeron. (Miller 2012; Ahmad, Dubrawsky, Flynn, Grand, Graham, Johnson, Kaminsky, Lynch, Manzuik, Permeh, Pfeil, Puppy & Russell 2002, 60.)

Apacheen liitettyjen tiedostojen ja kansioiden tiedosto-oikeudet voivat väärin konfiguroituna aiheuttaa ongelmia. Tiedosto-oikeudet voivat pahimmassa tapauksessa mahdollistaa mielivaltaisen koodin ajamisen (*arbitrary code execution*) palvelimella. Myös tiedostojen omistajiin on syytä kiinnittää huomiota, jotta esimerkiksi haavoittuva web-sovellus ei mahdollista koodi-tiedostojen päälle kirjoittamista, jolloin koko web-sovelluksen koodi voi tuhoutua. Tiedosto-oikeuksia käsitellään enemmän luvussa 3.3.4.. (Miller 2012.)

Palomuurin huolimattomalla konfiguroinnilla voi esimerkiksi sulkea itsensä ulos järjestelmästä estämällä verkkoyhteydet etähallintaohjelmistoon. Tämä voi aiheuttaa suuria ongelmia, varsinkin jos palvelimeen ei ole fyysistä pääsyä. Myös etähallintaohjelmaa konfiguroidessa on oltava varovainen edellä mainitusta syystä.

3.2.3 Keskeisiin ohjelmistoihin kohdistuvat uhat

Tässä työssä keskeisimmät ohjelmistot ovat Apache-web-palvelinohjelmisto, siihen liittyvä PHP-tulkkimoduuli, MySQL-tietokantaohjelmisto sekä OpenSSH-etähallintaohjelmisto. Nämä ohjelmistot ovat suorassa yhteydessä internetiin (tai Apachen välityksellä) ja siksi on tärkeää, että erityisesti näiden ohjelmistojen tietoturvaan kiinnitetään huomiota.

Suosionsa vuoksi Apache on hyvin yleinen kohde verkkohyökkäyksille (McClure ym. 272). Apachen koodia pidetään kuitenkin hyvälaatuisena ja ohjelmistoa turvallisena. Lisäksi Apachessa on tietoturvaominaisuuksia, jotka suojaavat sitä erilaisilta hyökkäyksiltä (Hacking Exposed Linux 2008, 73). Suurin osa Apacheen liittyvistä haavoittuvuuksista perustuukin huolimattomasti tehtyihin konfiguraatioihin, jotka mahdollistavat luottamuksellisen tiedon vuotamisen Apache-palvelinohjelmiston kautta. Kohdejärjestelmässä Apache voi vuotaa esimerkiksi web-sovelluksen lähdekoodin, jos Apachea ei ole konfiguroitu käsittelemään PHP-koodia asiaankuuluvalla tavalla, tai jos pääsyä Git-versionhallintajärjestelmän dataan ei ole estetty. (Hacking Exposed Linux 2008, 380; Bauer 2003.)

MySQL-tietokantaohjelmiston ja erityisesti sen sisältämien tietokantojen suurimmat uhat ovat SQL injection –hyökkäykset. Tällaisessa hyökkäyksessä hyökkääjä ajaa mielivaltaista koodia SQL-palvelimella, jolloin tiedon varastaminen, tuhoaminen tai muuttaminen voi olla erittäin yksinkertaista. Näihin hyökkäyksiin varaudutaan pääosin ohjelmistokehityksessä, joten niitä ei erikseen käsitellä tässä työssä. (Hacking Exposed Linux 2008, 385-388.)

Esimerkki vakavasta web-ohjelmiston haavoittuvuudesta, johon kohdistui satunnaisia nollapäivähyökkäyksiä, on maalikuussa 2012 paljastunut ongelma PHP:ssä. Haavoittuvuus mahdollisti mielivaltaisen koodin ajamisen kohdejärjestelmässä ja PHP-sivun lähdekoodin paljastamisen. Tämä haavoittuvuus kuitenkin koski vain tietynlaisia PHP-konfiguraatioita. (CVE-2012-1823 2012; Eindbazen PHP-CGI advisory 2012.)

Etähallintaohjelmistoihin, kuten OpenSSH:hon kohdistuu hyvin paljon niin kutsuttuja brute-force-hyökkäyksiä (Garfinkel ym. 2003). Brute-force-hyökkäyksessä pyritään ratkaisemaan merkkijono, esimerkiksi käyttäjätunnus ja/tai salasana, kokeilemalla kaikkia mahdollisia yhdistelmiä. Tavoitteena näillä hyökkäyksillä on saada pääsy järjestelmän komentorivirajapintaan. Brute-force-hyökkäyksiä on jo havaittu kohdistuvan kohdejärjestelmän SSH-palvelinohjelmistoon. (McClure ym. 2009, 228.)

3.2.4 Haittaohjelmat

GNU/Linux-palvelinten haittaohjelmat eroavat huomattavasti esimerkiksi Windows-työasemien haittaohjelmista, erityisesti leviämismenetelmiltään. Kun työasemalla haittaohjelma leviää usein esimerkiksi saastuneen asiakirjatiedoston välityksellä, palvelimelle haittaohjelma asennetaan palvelinohjelmiston haavoittuvuuden mahdollistaman koodin ajamisen kautta tai järjestelmän kaappauksen seurauksena.

Rootkitit ovat haittaohjelmia, jotka toimivat syvällä järjestelmän ytimessä ja antavat hyökkääjälle täyden pääsyn järjestelmään. Rootkitit muun muassa korvaavat olemassa olevia ohjelmia muokatuilla versioilla, jotka voivat sallia ohjelman hallinnan uudella tavalla. Rootkitit usein myös piilottavat olemassaolonsa tuhoamalla lokitiedostoja ja muokkaamalla prosessien seurantaan käytettäviä ohjelmia. (McClure ym. 2009, 292; 304; Hacking Exposed Linux 2008, 113; Ahmad ym. 2002, 68; Turnbull 2005, 282.)

Esimerkki GNU/Linux-palvelimen saastuttamisesta on Linux-kernelin jakelusivuston, *kernel.org*in, palvelimelle asennettu rootkit. Kesällä 2011 palvelimen ylläpitäjät huomasivat erikoisia virheilmoituksia, ja alkoivat tutkia asiaa tarkemmin. Selvisi, että järjestelmään oli asennettu Phalanx-rootkit. Rootkitin löytymistä auttoi se, että sitä ei ollut räätälöity piiloutumaan yleisesti käytetyiltä torjuntatyökaluilta. Alun perin pääsy järjestelmään oli saatu vuotaneen salaamattoman SSH-avaimen avulla. (Goodin 2011.)

3.2.5 Järjestelmän näkyvyys ja saatavuus

Järjestelmän tietoturvallisuutta voidaan tarkastella sen näkyvyyden (*visibility*) ja saatavuuden (*availability*) kannalta. Mikäli palvelimen olemassaolo voidaan jollain keinolla selvittää, on palvelin näkyvä. Esimerkiksi *ping*-kyselyihin vastaaminen tekee järjestelmästä näkyvän. Järjestelmän näkyvyys ulkomaailmaan luo erilaisille hyökkäyksille mahdollisuuden. Tietynasteinen näkyvyys on kuitenkin välttämättömyys, kun kyse on web-palvelimesta, joka tulee olla saavutettavissa internetistä käsin. Web-palvelimesta ei siis voida tässä projektissa tehdä täysin näkymätöntä, mutta näkyvyyttä voidaan pyrkiä vähentämään. (Hacking Exposed Linux 2008, 8-9.)

Saatavuudella viitataan järjestelmän interaktiivisuuteen; siihen, kuinka paljolti se toimii vastaanotettujen verkosta saapuvien pyyntöjen perusteella. Myös saatavuutta ei voida juurikaan rajoittaa web-ohjelmistojen osalta, kun kyse on web-palvelimesta. Verkkoliikennettä kuuntelevien taustaprosessien määrä on kuitenkin syytä minimoida, jolloin myös hyökkäyspinta-ala pienenee. Tärkeää on myös suodattaa ylimääräinen verkkoliikenne palomuurin avulla. (2008, 8-9.)

3.3 Ympäristön suojaaminen

Tässä luvussa käydään läpi ne keinot, joilla voidaan suojautua luvussa 3.2 kuvatuilta uhilta. Web-sovellusta suojatessa on tietoturvallisten ohjelmointikäytäntöjen lisäksi huomioitava myös palvelinalustan asianmukainen suojaus. Palvelinympäristön suojaamisella on siis tärkeä merkitys myös palvelimella ajettavan web-sovelluksen tietoturvalle. (2008, 382.)

Verkkolinnakkeella (*bastion host*) viitataan laitteeseen, joka ei sijaitse palomuurin suojaamassa sisäverkossa, vaan on esimerkiksi suorassa yhteydessä internetiin. Verkkolinnakkeelle on ominaista, että se on vastustuskykyinen erilaisille hyökkäyksille. Kohdejärjestelmää voidaan tämän kuvauksen perusteella pitää verkkolinnakkeena. (Valtionhallinnon tietoturvasanasto, 133.)

Tietoturvan kannalta on olennaista pienentää hyökkäyspinta-ala mahdollisimman pieneksi ja ajaa ainoastaan tarvittavia kuuntelevia taustaprosesseja. Hyökkäyspinta-alaa kasvattavat tekijät on hyvä olla tiedossa. Kohdejärjestelmässä näitä tekijöitä ovat muun muassa keskeiset web-ohjelmistot sekä SSH-palvelinohjelmisto. (Bauer 2003; Hacking Exposed Linux 2008, 12.)

Web-palvelimiin kohdistuvat uhat perustuvat pääsääntöisesti palvelimella ajettavien ohjelmistojen haavoittuvuuksiin ja virheellisesti tehtyihin ohjelmien konfiguraatioihin (McClure ym. 2009, 288). Koska suurin osa uhista perustuu ohjelmistoissa ilmeneviin haavoittuvuuksiin, kohdejärjestelmässä on pyritty eristämään mahdollisesti haavoittuvat ohjelmistot muusta järjestelmästä sen sijaan, että olisi pyritty etsimään keinoja ehkäistä näiden haavoittuvuuksien syntymistä. Ohjelmistotoimittajat ovat usein hitaita korjaamaan haavoittuvuuksia, minkä takia tavoitteena on ollut varautua haavoittuvuuksiin proaktiivisesti (Hacking Exposed Linux 2008, 60). Toisin sanoen järjestelmän turvallisuutta ei haluta jättää minkään yksittäisen ohjelman tietoturvan varaan.

3.3.1 Päivitysten merkitys tietoturvalle

Yksi tärkeimmistä tietoturvan osa-alueista on pitää järjestelmän ohjelmistot ajan tasalla päivitysten avulla. Tietoturvapäivitykset paikkaavat ohjelmistoissa ilmenneitä haavoittuvuuksia ja onkin kriittistä, että tietoturvapäivitykset asennetaan viiveettä, mielellään automaattisesti. Yksikin vakava haavoittuvuus voi mahdollistaa järjestelmän täydellisen kaappauksen (Rantala, 322). Kriittisintä on pitää ajan tasalla verkkoa kuuntelevat taustaprosessit ja niihin liittyvät ohjelmistot. (Hacking Exposed Linux 2008, 60; McClure ym. 2009, 233-234; IT-Grundschutz Kataloge 2005, 12.)

Koska suurimmaksi uhaksi kohdejärjestelmässä koetaan luvussa 3.2.1 kuvatut "purkitetut hyökkäykset", on tietoturvapäivitysten viiveettömällä asentamisella korkea painoarvo, sillä tämä suojelee tehokkaasti tämänkaltaisilta hyökkäyksiltä. Poikkeuksena ovat nollapäivähyökkäykset, sillä niille on ominaista, ettei tietoturvapäivitystä haavoittuvuuteen ole saatavilla. Kohdejärjestelmässä automaattinen päivitysten asentaminen on toteutettu *unattended-upgrades*-ohjelman avulla.

Käyttöjärjestelmän asennuksen jälkeen järjestelmä voi olla hyvin altis verkkohyökkäyksille, sillä silloin järjestelmä ei usein ole saanut uusimpia tietoturvapäivityksiä. Käyttöjärjestelmän asennuksen jälkeen on siis järjestelmä päivitettävä välittömästi. (Hacking Exposed Linux 2008, 103.)

Käyttöjärjestelmäkohtaisten ja Computer Emergency Response Team -organisaatioiden (CERT) postituslistojen välityksellä saa ajantasaista tietoa ohjelmistojen haavoittuvuuksista. On suositeltavaa, että ylläpitäjä liittyy tämänkaltaisille listoille pysyäkseen ajan tasalla ja saadakseen mahdollisuuden varautua vakaviin haavoittuvuuksiin tietoturvapäivityksillä tai jopa järjestelmän väliaikaisella sulkemisella. (Gagné 2003, 472; Bauer 2003; Turnbull 2005, 75-76.)

3.3.2 Taustaprosessien tietoturva

Taustaprosessit eli daemonit ovat prosesseja, jotka toimivat järjestelmän taustalla ja suorittavat tiettyä toimintoa. Taustaprosesseista käytetään myös nimitystä palvelu (*service*). Esimerkiksi Apache-web-palvelinohjelmisto on taustaprosessi. Taustaprosesseilla on olennainen merkitys järjestelmän tietoturvalle.

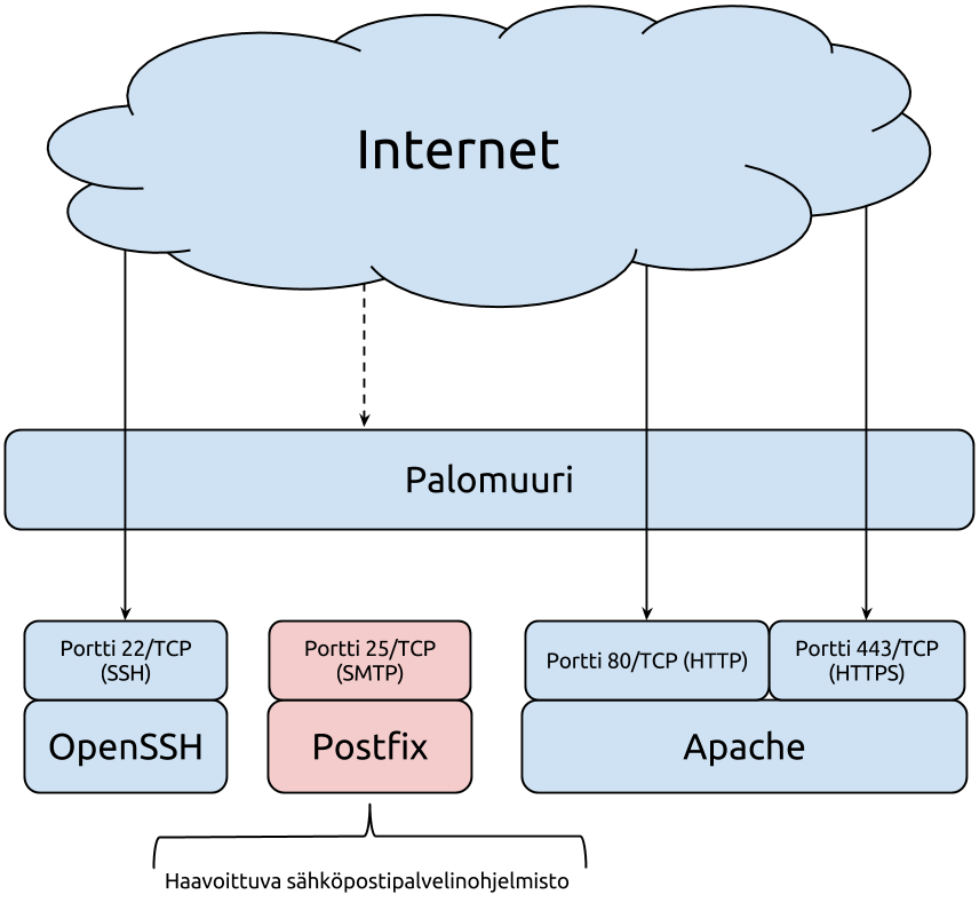
Taustaprosessit voidaan kuvitella ikkunoina talossa, johon murtovaras yrittää murtautua. Mitä enemmän niitä on, sitä todennäköisempää on se, että varas löytää yhden avonaisen. Avonainen ikkuna tässä tilanteessa symbolisoi haavoittuvaa tai väärin konfiguroitua taustaprosessia. Taustaprosessien määrä on siis syytä minimoida, sillä ne kasvattavat järjestelmän hyökkäyspinta-alaa. Huomiota on erityisesti kiinnitettävä verkkoliikennettä käsitteleviin taustaprosesseihin. (Hacking Exposed Linux 2008, 103; Bauer 2003; Miller 2012.)

Verkkoliikennettä kuuntelevia prosesseja voi seurata *ps*-, *netstat*- ja *lsof*-työkaluilla. On hyvä varmistaa, että yksikään taustaprosessi ei kuuntele verkosta saapuvaa liikennettä, ellei se ole välttämätöntä. (Hacking Exposed Linux 2008, 104; McClure ym. 2009, 231; 233-234.)

3.3.3 Verkkoliikenteen suodatus palomuurin avulla

Tässä työssä keskitytään erityisesti verkosta peräisin oleviin hyökkäyksiin. Kohdejärjestelmä ei sijaitse erillisen palomuurilaitteen suojaamassa sisäverkossa ja siksi on erityisen tärkeää, että järjestelmä on konfiguroitu itse suodattamaan siihen kohdistuvaa verkkoliikennettä.

Palomuurilla voidaan tarkoittaa fyysistä laitetta tai ohjelmistoa. Tässä työssä palomuurilla tarkoitetaan ohjelmaa, jonka avulla suodatetaan verkkoliikenteen IP-paketteja otsake- ja istuntotietojen perusteella. Kohdejärjestelmässä on tiedossa ne portit, jotka ovat välttämättömiä web-sovelluksen ja etähallinnan toiminnalle. Nämä portit täytyy siksi olla auki ulkomaailmaan. Toisin sanoen näihin portteihin saapuva liikenne tulee käsitellä ja kaikki muu järjestelmään kohdistettu liikenne on syytä suodattaa. Suodatuksen avulla voidaan vähentää järjestelmän näkyvyyttä ja voidaan varmistua, ettei esimerkiksi haavoittuva kuunteleva taustaprosessi vastaanota internetistä saapuvaa verkkoliikennettä. Kuviossa 5 kuvataan tilanne, jossa palomuuri suojelee haavoittuvaa sähköpostipalvelinta suodattamalla siihen kohdistuvan liikenteen. Sähköpostipalvelimen olemassaolo on voitu esimerkiksi unohtaa. (Hacking Exposed Linux 2008, 374; Koski & Kajala 2005, 40.)



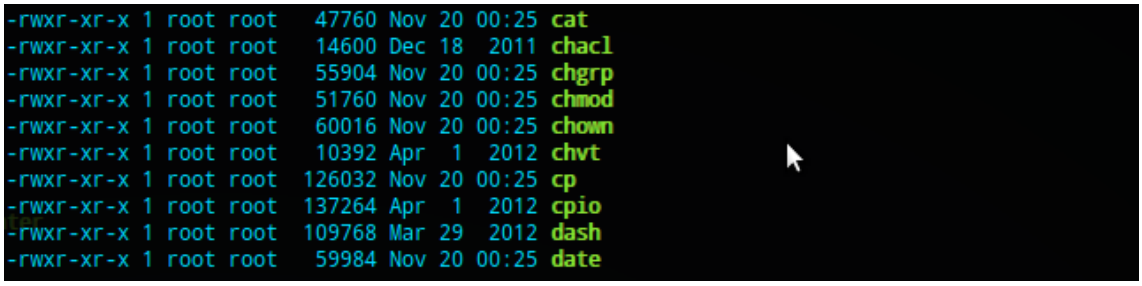
Kuvio 5: Palomuurin toiminta.

Useimpien GNU/Linux-jakeluiden mukana tulee IPTables-palomuuriohjelma, jolla hallitaan Linux-kernelin Netfilter-pakettisuodatusmoduulia. Netfilter kykenee niin sanottuun tilapohjaiseen pakettisuodatukseen, eli se pystyy suodattamaan verkkoliikennettä myös istuntojen tilan perusteella, eikä pelkästään IP-pakettien otsaketietojen. Ubuntu Serverin mukana tulee lisäksi Uncomplicated Firewall -ohjelma, joka yksinkertaistaa IPTables-ohjelman käyttöä (Miller 2012). Palomuuria konfiguroidessa tulee olla erityisen varovainen, ettei estä verkkoyhteyksiä etähallintaohjelmistoihin, jolloin ylläpitäjä ei enää pääsisi käsiksi järjestelmään. (Turnbull 2005, 80-81; Hacking Exposed Linux 2008, 373.)

3.3.4 Pääsynvalvonta (*Access Control*)

GNU/Linuxin oletusarvoista pääsynvalvontamallia kutsutaan *Discretionary Access Controliksi* (DAC). DAC-mallissa jokaiselle tiedostolle on määritetty omistaja ja ryhmä. Tämän lisäksi tiedostoille on määritetty omistaja- ja ryhmäkohtaiset luku-, kirjoitus- ja ajo-oikeudet. DAC-mallissa datan omistajalla on täydet oikeudet omistamaansa dataan, mikä voi tietyissä tilanteissa aiheuttaa ongelmia. (Hacking Exposed Linux 2008, 70; Haeder, Addison Schneiter, Gomes Pessanha & Stanger 2010, 176-177.)

Tiedostojen ja kansioiden oikeudet kuvataan tyypillisesti yhdeksänmerkkisenä jonona. Ensimmäinen merkki (*d*) kertoo, onko kyseessä kansio vai ei. Seuraavat kolme merkkiä määräävät tiedoston omistajan oikeudet, luku (*r*), kirjoitus (*w*) ja ajo (*x*). Seuraava kolmimerkkinen osuus määrää tiedostoon liitetyn ryhmän (*group*) oikeudet ja kolme viimeistä määrittävät kaikkien muiden käyttäjien (*other*) oikeudet tiedostoon. Viiva kirjaimen tilalla tarkoittaa, että kyseistä oikeutta ei ole. Kuvassa 2 ajettavien binääritiedostojen tiedosto-oikeuksia on tulostettu *ls*-komennon avulla.



Kuva 2: Tiedosto-oikeudet.

GNU/Linux-järjestelmässä myös jokaisella prosessilla on omistajansa. DAC-mallissa prosessilla on omistajansa oikeudet järjestelmään. Prosessin omistaja on yleensä prosessin käynnistänyt käyttäjä. Jos ajettavalla tiedostolla (*executable file*) on kuitenkin tiedostojärjestelmässä

SUID-merkintä (*set user ID upon execution*), prosessin omistajaksi määräytyy aina suoritettavan tiedoston omistaja.

Mikäli prosessi kaapataan esimerkiksi haavoittuvuuden avulla, hyökkääjä saa siis yleensä tämän prosessin omistajan oikeudet järjestelmään. On siis erityisen vaarallista, jos kaapatun prosessin omistaja on *root*-käyttäjä. Tällöin hyökkääjä saa käytännössä täydet oikeudet kaikkiin järjestelmän toimintoihin (Bauer 2007). Suuri osa hyökkäyksistä perustuu juurikin *root*-käyttäjän omistamien prosessien ja SUID-merkittyjen suoritettavien tiedostojen hyväksikäyttöön. Tästä syystä on järjestelmässä vältettävä pitämästä SUID-merkittyjä ajettavia tiedostoja, joiden omistaja on *root*-käyttäjä. SUID-merkintä on tosin hyödyllinen ja toivottava ominaisuus silloin, kun käyttäjä tarvitsee hetkellisesti laajennettuja oikeuksia järjestelmään, kuten esimerkiksi salasanaa vaihtaessa. (Haeder ym. 2010, 177-178; McClure ym. 2009, 288; 290.)

Edellä mainituista syistä järjestelmässä tulee minimoida *root*-käyttäjänä ohjelmien ajaminen ja ajaa keskeisten ohjelmistojen prosesseja käyttäjillä, joilla on ainoastaan ne oikeudet, joita ne tarvitsevat normaaliin toimintaansa. Tämä on erityisen tärkeää kun kyseessä on prosessi, joka ottaa vastaan ja käsittelee verkosta saapuvaa liikennettä. Esimerkiksi web-palvelinohjelmisto tulisi *root*-käyttäjän sijaan ajaa *www-data*-käyttäjänä, jolle on määritetty *root*ia huomattavasti suppeammat oikeudet. (McClure ym. 2009, 233-234, 246; Hacking Exposed Linux 2008, 61.)

3.3.5 Mandatory Access Control ja AppArmor

Pääsynvalvonnan hallintaa voidaan laajentaa Mandatory Access Control ja Role-Based Access Control -mallien ja -järjestelmien avulla. Mandatory Access Control -malli on DAC-mallia tuoreempi ja monimutkaisempi pääsynvalvontamalli. MAC-malli täydentää DAC-mallin ominaisuuksia, mutta ei kuitenkaan korvaa sitä. MAC-järjestelmän avulla voidaan rajoittaa prosessien ja käyttäjien oikeuksia kohdejärjestelmään ja sen tiedostoihin ja prosesseihin. Oikeuksia on syytä jakaa ainoastaan prosessin normaaliin toimintaan tarvittava määrä, eikä yhtään enempää. Tämä käsite tunnetaan pienimmän oikeuden periaatteena (*least privilege principle*). (Bauer 2007; Hacking Exposed Linux 2008, 70-72.)

Haavoittuvuuksien paljastuminen ja tietoturvapäivityksen julkaiseminen on eräänlaista kissa-hiiri-leikkiä. Jatkuvasti paljastuu uusia haavoittuvuuksia ja niitä käytetään laajalti hyväksi. Kriittisintä aikaa on haavoittuvuutta hyväksikäyttävän työkalun julkaisun ja tietoturvapäivityksen välinen aika; silloin järjestelmä on erittäin suuressa vaarassa. MAC-järjestelmä pystyy tietyissä tilanteissa suojelemaan järjestelmää näiltä nollapäivähyökkäyksiltä. Kaapattu prosessi ei esimerkiksi pystykään vaikuttamaan muuhun järjestelmään siten, että sen pystyisi kaappaamaan. (Hacking Exposed Linux 2008, 103.)

AppArmor on Novellin kehittämä, Ubuntuun oletusarvoisesti asennettu MAC-järjestelmä. AppArmor suojaa järjestelmää hyökkäyksiä vastaan, joissa hyväksikäytetään ohjelmassa olevaa haavoittuvuutta. AppArmoria pidetään SELinux-järjestelmää yksinkertaisempana konfiguroida (Bauer 2006). AppArmor ei ole yhtä kokonaisvaltainen järjestelmä kuin SELinux, mutta se keskittyy erityisesti ohjelmistohaavoittuvuuksiin perustuviin hyökkäyksiin, joita pidetäänkin suurimpana uhkana web-palvelinten tietoturvalle. (Bauer 2009; Novell AppArmor Administration Guide 2007, 1.)

Ohjelmalle, joka halutaan eristää, luodaan AppArmor-profiili. Profiili määrittää, mitä ohjelma voi järjestelmässä tehdä; esimerkiksi mihin kansioon sillä on pääsy ja minkä tiedoston päälle sillä on oikeus kirjoittaa. Osalle järjestelmän ohjelmista on valmiiksi määritetyt profiilit. Kohdejärjestelmässä on haluttu erityisesti rajoittaa keskeisten, verkkoa kuuntelevien ohjelmien oikeuksia muuhun järjestelmään. (Hacking Exposed Linux 2008, 61-62; Novell AppArmor Administration Guide 2007, 1.)

Ajettavan web-sovelluksen oikeuksia järjestelmään voidaan rajoittaa edelleen Apachen AppArmor-moduulin avulla. Web-sovellukselle luodaan oma profiili, joka toimii Apachen AppArmor-profiilin aliprofiilina. Tämän aliprofiilin avulla web-sovellukselle voidaan määrittää suppeammat oikeudet kuin Apachen pääprofiilille ja mahdollisille muille web-sovelluksille, joita ajetaan samalla palvelimella. Intranet-järjestelmälle luotiin tällainen profiili, mikä auttaa edelleen eristämään web-sovellusta muusta järjestelmästä. (Novell AppArmor Administration Guide 2007, 67.)

Kun intranet-järjestelmää varten vuokrattu palvelin otettiin käyttöön, siihen oli palveluntarjoajan toimesta asennettu GrSecurity-RBAC-järjestelmä. GrSecurity poistettiin käytöstä ja tilalle asennettiin AppArmor, koska tietoturvajärjestelmän ei haluttu olevan palveluntarjojakohtainen.

3.3.6 Web-ohjelmistojen suojaaminen

Tässä luvussa käydään läpi käytettävien web-ohjelmistojen suojaus. Web-ohjelmistoilla viitataan tässä työssä Apache-web-palvelinohjelmistoon, Apachen PHP-tulkkimoduuliin ja MySQL-tietokantaohjelmistoon. Nämä web-ohjelmistot toimivat yhteistyössä, jossa muutokset yhdessä osassa usein vaikuttavat koko pinon (*stack*) toimintaan.

3.3.6.1 Apache-web-palvelinohjelmisto

Apachea pidetään melko turvallisena web-palvelinohjelmistona ja se sisältää toiminnallisuuksia, joilla se eristää itsensä muusta järjestelmästä (Hacking Exposed Linux 2008, 73). Apachen prosessin oikeuksia halutaan kuitenkin rajoittaa edelleen muuhun järjestelmään AppArmor-profiilin avulla.

Apachen oletusarvoisia asetuksia on myös syytä muuttaa ennen käyttöönottoa. Apache-web-palvelin on tärkeä konfiguroida siten, ettei se vuoda luottamuksellista tietoa tai järjestelmän tiedostoja. Voi olla myös perusteltua konfiguroida Apache siten, että se antaa mahdollisimman vähän tietoa hyökkääjälle, kuten käyttöjärjestelmän ja käytettyjen moduulien versionumeroita. Myös kansioiden indeksointi (*indexes*), jonka avulla kuka tahansa näkee kansiorakenteen sisällön, on hyvä ottaa pois päältä. Kohdejärjestelmässä on erityisen tärkeää estää ulkopuolisten pääsy versionhallintadataa sisältävään *.git*-kansioon, sillä se sisältää web-sovelluksen lähdekoodin. Myös mahdollisuus ajaa CGI-skriptejä on syytä poistaa, sillä tälle ominaisuudelle ei ole tarvetta. (Hacking Exposed Linux 2008, 95; 376; 380.)

Yksi tapa lisätä järjestelmän tietoturvaa on konfiguroida se välittämään virheellistä tietoa järjestelmästä, esimerkiksi sen ohjelmistojen versioista. Tähän lähestymistapaan viitataan käsitteellä *security through obscurity*. Tämä menetelmä voi tuoda tietoturvaan yhden kerroksen lisää, sillä mahdollinen hyökkääjä saa kohdejärjestelmästä vähemmän hyökkäyksessä hyödynnettävää tietoa. Automaattiselle haavoittuvuusskannaukselle syötetty väärä versiotieto voi myös estää hyökkäyksen käynnistymisen. Apachen voi esimerkiksi *headers*-moduulin avulla konfiguroida ilmoittamaan olevansa Microsoft IIS –web-palvelinohjelmisto. Tällöin on mahdollista välttyä Apacheen kohdistetulta hyökkäykseltä. Kohdejärjestelmässä tätä metodia ei kuitenkaan ole hyödynnetty, sillä sen tuoma lisäturva on koettu pieneksi. (Hacking Exposed Linux, 94-95; Ahmad ym. 2002, 61-62.)

3.3.6.2 PHP, Suhosin ja MySQL

Apachen PHP-moduulin ja MySQL-tietokantaohjelmiston toiminta liittyy läheisesti web-palvelimen toimintaan. Löyhästi konfiguroituna PHP voi paljastaa haavoittuvuuksia web-sovelluksessa tai PHP-moduulissa. PHP voi esimerkiksi näyttää käyttäjälle yksityiskohtaisia virheilmoituksia, joita voidaan käyttää hyväksi hyökkäyksen suunnittelussa. PHP-moduuli on syytä konfiguroida siten, ettei se paljasta käyttäjälle taustalla tapahtuvia virheitä yksityiskohtaisesti. (Hacking Exposed Linux 2008, 376.)

PHP-tulkkiin on saatavilla Suhosin-niminen lisäosa, joka lisää tulkin tietoturvaa. Suhosin suojaa järjestelmää huonosti kirjoitetulta, haavoittuvalta PHP-koodilta. Koska ohjelmistokehityk-

seen liittyvät tietoturvakäytännöt on rajattu tästä työstä ulos, on perusteltua pitää intranet-järjestelmän koodin turvallisuutta epäluotettavana. Suhosin suojelee järjestelmää muun muassa URL include -hyökkäkysiltä ja puskuriylivuotohyökkäyksiltä, jotka johtuvat huonoista ohjelmistokehityskäytännöistä. (What is Suhosin?; Why Suhosin?.)

Vaikka ylläpitäjän lisäksi kenellekään ei ole suoraa pääsyä tietokantaan, on syytä tarkistaa, että kaikille tietokannan käyttäjätunnuksille on määritetty salasana. Erityisesti on varmistettava, että tietokannan pääkäyttäjäksi ei voi kirjautua ilman salasanaa. (Bauer 2003.)

3.3.7 Web-sovelluksen suojaaminen

Vaikka tämän työn ensisijaisena tavoitteena on suojata palvelinympäristöä siihen kohdistuvilta uhilta, pyritään palvelimen ohjelmistoilla ja niiden lisäosilla parantamaan myös web-sovelluksen tietoturvaa. Tässä luvussa käydään läpi, miten web-sovelluksen tietoturvaa voidaan parantaa palvelimen tasolla erilaisten moduulien avulla koskematta web-sovelluksen koodiin. Näillä työkaluilla voidaan muun muassa tarkastella käyttäjäsyötettä ja salata tietoliikenne.

3.3.7.1 Käyttäjäsyötteen tarkastelu ja suodatus

Käyttäjäsyötteen suodatus ja validointi on erittäin tärkeä osa web-sovelluksen tietoturvaa. Tämä tehdään lähtökohtaisesti ohjelmiston koodissa, mutta syötettä voi kuitenkin tarkastella ja suodattaa myös HTTP-palvelimen tasolla ModSecurity Web Application Firewall -moduulin avulla. Suodatusominaisuuksien ansiosta ModSecurityn avulla voidaan suoraan vaikuttaa myös web-sovelluksen tietoturvaan.

ModSecurity on Apachen moduuli, jonka avulla voi estää tietynlaisia web-sovellukseen kohdistuvia hyökkäyksiä. Sen avulla voidaan tarkastella HTTP-tasolla tietoa, jonka käyttäjä on syöttänyt järjestelmään esimerkiksi web-sovelluksen HTML-lomakkeen (*form*) kautta. Tietoa tarkastellaan ennen kuin se käsitellään web-palvelimen ja lopulta web-sovelluksen toimesta. Mikäli järjestelmä havaitsee syötteessä jotakin epäilyttävää, se voi olla välittämättä sitä eteenpäin web-palvelinohjelmistolle ja -sovellukselle. (Hacking Exposed Linux 2008, 105; Barnett 2006.)

3.3.7.2 Palvelunesto- ja brute-force-hyökkäyksiltä suojautuminen

Web-sovellukseen kohdistuvilta palvelunesto- ja brute-forcehyökkäyksiltä voi suojautua esimerkiksi Apachen *evasive*-moduulin avulla. Moduuli tarkkailee HTTP-liikennettä ja havaitessaan useita peräkkäisiä yhteydenottoja samasta IP-osoitteesta se rupeaa hylkäämään osoit-

teesta saapuvaa liikennettä. *Evasive*-moduuli suojaa sekä brute-force- että palvelunestohyökkäyksiltä, sillä molemmat perustuvat useiden peräkkäisten pyyntöjen tekemiseen. Hajautetut palvelunestohyökkäykset ovat usein kuitenkin niin voimakkaita, että niiltä on vaikea suojautua ilman valtavia resursseja (Garfinkel ym. 2003). Tällaisia hyökkäyksiä ei kuitenkaan pidetä suurena riskinä ympäristölle. (Barnett 2006.)

3.3.7.3 Web-liikenteen salaus

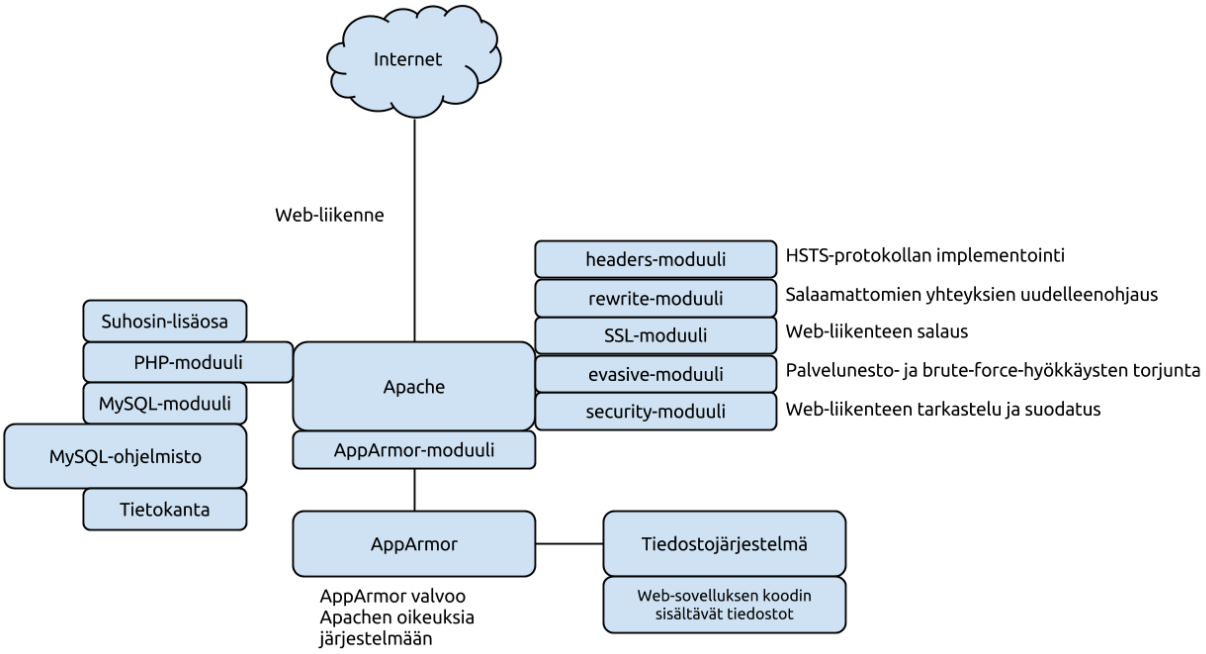
Salauksella tarkoitetaan tiedon muuttamista sellaiseen muotoon, että siitä ei saa selkoa ilman tiettyä avainta, esimerkiksi salasanaa. Tietoliikenteen salauksen avulla voidaan varmistua siitä, että tiedon luottamuksellisuus ja eheys säilyy. Web-liikenteen salaus on yksi keino, jolla voi palvelintasolla vaikuttaa suoraan web-sovelluksen tietoturvaan. Käytännössä tällä pyritään estämään esimerkiksi salasanojen ja intranet-järjestelmän luottamuksellisen tiedon vuotaminen kolmansille osapuolille. Näin voi tapahtua esimerkiksi, jos kolmas osapuoli pääsee tavalla tai toisella kuuntelemaan käyttäjän ja palvelimen välistä tietoliikennettä. Tiedonsiirron salakuuntelun mittakaava voi vaihdella salaamattoman WLAN-verkon kuuntelusta valtiotason vakoiluun, joka onkin syytä ottaa huomioon yrityksen kohdemaasta johtuen.

Oletusarvoisesti Apache-webpalvelinohjelmisto lähettää www-sivut salaamattomasti hyväksikäyttäen HTTP-protokollaa. Apachen SSL-moduulin avulla voidaan selkokielisen HTTP:n sijaan käyttää HTTPS-protokollaa, jolloin kaikki liikenne käyttäjän web-selaimen ja palvelimen välillä on salattua. Palvelin on syytä myös konfiguroida automaattisesti muuttamaan selkokieliset HTTP-pyynnöt salatuiksi HTTPS-pyynnöiksi, jotta palvelin ei missään tilanteissa lähetä tietoa salaamattomassa muodossa. Tämä onnistuu Apachen *rewrite*-moduulin avulla. (Koski & Kajala 2005, 31;441.)

Apache voidaan konfiguroida ilmoittamaan käyttäjien web-selaimille, että palvelimen kanssa tulee kommunikoida ainoastaan salatun HTTPS-yhteyden välityksellä. Tämä onnistuu HTTP Strict Transport Security –protokollan (HSTS) avulla. HSTS suojaa käyttäjää muun muassa SSL Stripping –hyökkäyksiltä. HSTS-protokolla voidaan ottaa käyttöön Apachen *headers*-moduulin avulla. (Hodges, Jackson & Barth 2012.)

Kuviossa 6 on kuvattu web-palvelinohjelmiston toiminta.

Intranet-järjestelmää varten web-palvelimelle hankittiin SSL-sertifikaatti. Sertifikaatin voidaan avulla varmentaa, että palvelin on se, joka se ilmoittaa olevansa (Koski & Kajala 2005, 451). Sertifikaattien ja julkisen avaimen salauksen toimintaperiaatteet on kuitenkin rajattu ulos tästä opinnäytetyöstä.



Kuvio 6: Web-palvelinohjelmiston toiminta yksinkertaistettuna.

3.3.8 Etähallintaohjelmiston suojaaminen

SSH-etähallintaohjelmiston avulla voidaan kirjautua palvelimen komentorivirajapintaan ja sitä kautta hallita kaikkia käyttöjärjestelmän ominaisuuksia. On siis äärimmäisen tärkeää, ettei luvaton henkilö pääsee kirjautumaan tätä kautta järjestelmään. Autentikointi SSH-palvelimelle tapahtuu yleensä salasanalla tai julkisen avaimen menetelmällä.

Käyttöjärjestelmät sisältävät usein käyttäjiä, joiden käyttäjänimi ja salasana ovat identtiset (McClure ym. 2009). Tällaisella käyttäjällä on äärimmäisen helppoa kirjautua järjestelmään. Tähän voidaan varautua sallimalla ainoastaan tiettyjen käyttäjien kirjautuminen. Lisäksi SSH:n salasana-autentikoinnin estäminen suojaa tältä haavoittuvuudelta. On siis syytä käyttää järjestelmään autentikoimiseen julkisen avaimen menetelmää ja kytkeä salasanalla kirjautuminen pois päältä.

OpenSSH-ohjelmistoa pidetään hyvin turvallisena ja sillä on ominaisuus, joka eristää sen muusta järjestelmästä. On kuitenkin varmistettava, että tämä ominaisuus on kytketty päälle. Ohjelmistolle on saatavilla myös AppArmor-profiili, jolla sitä voidaan eristää yksityiskohtaisemmin muusta järjestelmästä. Etähallintaohjelman konfigurointi tuotantokäytössä olevassa järjestelmässä on kuitenkin riskialtista, sillä epäonnistunut konfiguraatio voi lukita ylläpitäjän ulos järjestelmästä. (Hacking Exposed Linux 2008, 78; McClure ym. 2009, 270.)

SSH-etähallintaohjelmistoihin kohdistuu paljon brute-force-hyökkäyksiä. Salasana-autentikoinnin poistaminen suojaa tehokkaasti näiltä hyökkäyksiltä, mutta sen lisäksi voidaan käyttää Denyhosts-ohjelmaa. Denyhosts seuraa kirjautumisyrityksiä SSH-palvelimelle ja kun yhdestä IP-osoitteesta tulee liian monta yritystä, se lisää osoitteen mustalle listalle, jolloin tästä osoitteesta ei voi enää kirjautua palvelimelle.

Julkisen avaimen menetelmää käyttäessä on erittäin tärkeä säilyttää salaisen avaimen luottamuksellisuus. Salainen avain voidaan helposti salata salasanalla luomisensa yhteydessä. Tämä avain on syytä myös varmuuskopioida avaimen eheyden ja saatavuuden varmistamiseksi. Kun salasana-autentikointi on kytketty pois päältä, ilman tätä avainta ylläpitäjä ei pysty kirjautumaan palvelimelle.

3.3.9 Haittaohjelmilta suojautuminen

Haittaohjelmilta suojautumiseen ei käytetä erityisen paljon resursseja, vaan tavoitteena on pyrkiä kaikin keinoin estämään keinot, joiden kautta haittaohjelma voitaisiin asentaa. On kuitenkin suositeltavaa säännöllisesti tarkastaa järjestelmä haittaohjelmien varalta esimerkiksi Rootkit Hunter tai chkrootkit-ohjelmalla. Nimensä mukaisesti ohjelmat etsivät erityisesti rootkit-haittaohjelmia. Rootkitit ovat kuitenkin tehokkaita piilottamaan olemassaolonsa ja rootkitin tekijä on voinut suunnitella ohjelmansa piiloutumaan juuri näiltä ohjelmilta. Lisäksi on suositeltavaa seurata järjestelmän kriittisiä tiedostoja muutosten varalta esimerkiksi Debsums-ohjelman avulla. Debsums laskee tiedostojen hajautusarvoja ja ilmoittaa ylläpitäjälle, jos se huomaa muutoksia tiedostoissa. (Turnbull 2005, 283; 285.)

Rootkitin löytyessä tai tietomurron ilmetessä on syytä asentaa järjestelmä alusta alkaen uudestaan, sillä järjestelmän antamiin tietoihin ei voi enää luottaa (McClure ym. 2009, 294; Rantala 2003, 332; Bauer 2003; Stanger ym. 2001, 111; Turnbull 2005, 315-316). Tämänkaltaisesta toimenpiteestä on pyritty tekemään yksinkertainen työssä luodun dokumentaation avulla.

Intranet-järjestelmässä on ominaisuus, jonka avulla järjestelmään pystyy siirtämään tiedostoja, kuten esimerkiksi asiakirjoja. Käyttäjien tietoturvan kannalta on hyvä varmistaa, etteivät nämä tiedostot sisällä haittaohjelmia. Tähän tarkoitukseen sopii esimerkiksi avoimen lähdekoodin ClamAV. ClamAV pystyy etsimään haittaohjelmia asiakirjojen sisältä ja löytää myös muihin käyttöjärjestelmiin kohdistettuja haittaohjelmia (About ClamAV®). ClamAV ajastettiin etsimään haittaohjelmia päivittäin. Mikäli haittaohjelmia löytyy, asetetaan saastunut tiedosto karanteeniin, jotta sitä ei pystyisi lataamaan web-sovelluksen kautta.

3.3.10 Järjestelmän monitorointi

Järjestelmän monitorointi on tärkeä osa tietoturvaa. Perusteelliset ja tarkat lokitiedot ovat tärkeä osa toimivaa järjestelmää. Lokitietoja pitää myös tarkastella järjestelmävirheiden ja esimerkiksi tietoturvahyökkäysten varalta. Erityisen tärkeää on seurata järjestelmässä tapahtuvia muutoksia. Yksi ongelma lokien seuraamisessa on tiedon valtava määrä; tärkeä tieto on erotettava merkityksettömästä tiedosta. (Miller 2012; Bauer 2003; Stanger ym. 2001, 192; Turnbull 2005, 233.)

Onnistunut hyökkääjä pyrkii usein peittämään jälkensä tuhoamalla tietoa onnistuneesta murrosta. Tietoja hyökkäyksistä löytyy usein järjestelmän lokitiedostoista. Järjestelmään murtautunut henkilö voi kuitenkin poistaa näitä lokitietoja. Tästä syystä lokitietoja on syytä tarkastella säännöllisesti ja varmuuskopioida järjestelmän ulkopuolelle, jotta niiden eheys säilyy. Lokitietoja voi myös tallentaa medioille, joiden päälle ei ole mahdollista kirjoittaa uudelleen. (McClure ym. 2009, 298; 302-303; Ahmad ym. 2002, 67; Turnbull 2005, 233.)

Järjestelmää voi seurata muun muassa luvussa 3.3.9 läpikäydyillä Debsums- ja Rootkit Hunter -ohjelmilla. Järjestelmän monitorointiin voisi myös käyttää esimerkiksi avoimen lähdekoodin Snort-tunkeutumisenestojärjestelmää (*Intrusion Prevention System*). Snort on kuitenkin melko raskas ohjelma ja vaatii toimiakseen paljon järjestelmän resursseja. Tästä syystä kohdejärjestelmässä ei käytetä Snortia. (Bauer 2003.)

3.3.11 Varmuuskopiointi

Intranet-järjestelmä sisältää tietoa, joka on erittäin tärkeää kohdeyritykselle. On siis tärkeää, että tämä tieto pysyy tallessa. Järjestelmän sisältämä taloudenhallintaan liittyvän tiedon eheys koetaan tärkeimmäksi osaksi järjestelmän tietoturvaa. Tiedon eheys on siis tärkeämpää kuin tiedon luottamuksellisuus ja järjestelmän saatavuus. Tästä syystä varmuuskopiointiin on kiinnitetty työssä erityisen paljon huomiota.

Tiedon eheys ja säilyvyys varmistetaan varmuuskopioinnilla. Hyvin toteutettu varmuuskopiointi pienentää esimerkiksi onnistuneen hyökkäyksen tai laiterikon aiheuttamaa vahinkoa. Työn intranet-järjestelmässä arvokkain tieto on MySQL-tietokannassa, mutta myös esimerkiksi konfiguraatiotiedostot halutaan varmuuskopioida, jotta mahdollinen uudelleenkäyttöönottoprosessi voisi tapahtua nopeammin.

Järjestelmän varmuuskopioinnin laadun varmistamisessa hyödynnettiin KATAKRI-viitekehystä. KATAKRIN II:en määrittämät korkean viranomaistason vaatimukset ovat seuraavat:

- Toimintavaatimusten huomioiminen varmuuskopioinnissa
- Varmuuskopioiden säilyttäminen eri fyysisessä sijainnissa
- Varmuuskopioihin pääsee käsiksi ainoastaan valtuutetut henkilöt
- Varmuuskopiot säilytetään salatussa tilassa
- Varmuuskopioista on olemassa lista
- Varmuuskopiointijärjestelyn suhteuttaminen tiedon kriittisyyteen
- Palautusprosessin pitää olla riittävän nopea (toimintavaatimukset huomioiden)
- Varmuuskopioiden toimivuus testattava säännöllisesti
- Varmuuskopiointiin liittyvä dokumentaatio on riittävällä tasolla

(KATAKRI 2011, 117.)

Järjestelmän varmuuskopiointi on toteutettu usealla eri bash-koodikieltä hyödyntävällä skriptillä. Tietokanta varmuuskopioidaan kohdejärjestelmässä yhdeksi tiedostoksi *mysqldump*-komennon avulla. Tietokannan varmuuskopiointi tehdään ajastetusti tunnin välein cron-ohjelman avulla. Kolmen tunnin välein tietokanta ja muut tiedostot ladataan varmuuskopiointipalvelimelle, josta tiedot kopioidaan edelleen kolmeen eri varmuuskopiointijärjestelmään. Varmuuskopiointijärjestelmät sijaitsevat kohdejärjestelmän kanssa eri fyysisissä sijainneissa. Tiedonsiirtoon käytetään *rsync*-ohjelmaa ja se tunneloidaan SSH-yhteyden läpi, jolloin kaikki tietoliikenne on salattua.

Varmuuskopiointipalvelimelta varmuuskopioiden päiväykset tarkistetaan päivittäin manuaalisesti. Näin varmistetaan, että varmuuskopiot ovat saapuneet kohdejärjestelmästä varmuuskopiointipalvelimelle. Varmuuskopioista tehdään automaattisesti päivittäin uusi lista *ls*-komennolla, jonka tuloste ohjataan tekstitiedostoon.

Varmuuskopioiden toiminta varmistetaan tuomalla säännöllisesti varmuuskopiot intranet-järjestelmän kehitysympäristöön, joka sijaitsee eri sijainnissa kuin kohdejärjestelmä. Koko järjestelmän palautusprosessin nopeus on testattu ja se onnistuu alle kahdessa tunnissa. Opinnäytetyön produkti, käyttöönottodokumentaatio, kuvaa varmuuskopioiden palautusprosessin perusteellisesti.

Kohdejärjestelmän web-sovelluksen koodin sisältävät tiedostot (PHP, HTML, CSS ja JavaScript) eivät ole tärkeitä, sillä sovelluksen koodin uusimmat versiot löytyvät opinnäytetyön tekijän kehitysympäristöstä. Myös kehitysympäristön koodi varmuuskopioidaan useita kertoja päivässä, myös toisiin fyysisiin sijainteihin. Koodin hallinnassa käytetään Git-versionhallintaohjelmaa, joka mahdollistaa esimerkiksi web-sovelluksen koodin palauttamisen aikaisempaan tilaan.

Varmuuskopiot on salattu pääasiallisesti Truecrypt- ja *dm-crypt*-ohjelmilla.

Järjestelmän varmuuskopioinnin järjestely täyttää KATAKRI II:ssa määritellyt sekä elinkeinoelämän että korkean tason viranomaisvaatimukset liittyen varmuuskopiointiin. Varmuuskopiointijärjestely on siis myös hieman yliampuva, mutta tätä on vaikea nähdä negatiivisena asiana.

3.3.12 Haavoittuvuusskannaus

On suositeltavaa testata järjestelmää erilaisten haavoittuvuuksien varalta. Tässä luvussa käydään läpi kolmen eri haavoittuvuusskannausohjelman toiminta.

Yksinkertainen tapa saada tieto palvelimella ajettavista palveluista, on suorittaa niin kutsuttu porttiskannaus. Tämä onnistuu esimerkiksi *nmap*-ohjelmalla. Nmapin avulla voidaan testata muun muassa palomuurin pakettisuodatuksen toiminta ja testata mitä verkkoa kuuntelevat ohjelmat kertovat itsestään ulkomaailmalle. (Hacking Exposed Linux 2008, 371; Stanger ym. 2001, 131; Turnbull 2005, 296.)

Lynis on haavoittuvuusskannausohjelma, joka ajetaan paikallisesti palvelimella. Lynis soveltuu erityisesti konfiguraatiovirheiden kartoittamiseen, mutta se tunnistaa myös joitain ohjelmistohaavoittuvuuksia. Se antaa selkeitä ohjeita siitä, kuinka järjestelmän turvallisuutta voi parantaa konfiguraatioita muuttamalla. (Boelen 2009.)

Nessus on suosittu haavoittuvuusskanneri, jolla on ajantasaista tietoa ohjelmistohaavoittuvuuksista. Nessus skannaa verkon välityksellä muun muassa verkon kuuntelevien taustaprosessien versioita ja simuloi eri haavoittuvuuksiin perustuvia hyökkäyksiä. Tämän avulla ylläpitäjä saa ajantasaista tietoa järjestelmänsä ohjelmistohaavoittuvuuksista. Nessusin ilmaislisenssi ei kuitenkaan salli ohjelman käyttöä kaupallisessa käytössä. Tästä syystä Nessusia ei voida käyttää kohdejärjestelmän haavoittuvuuksien analysoinnissa. (Ahmad ym. 2002, 730; Stanger ym. 2001, 165; Turnbull 2005, 302.)

4 Johtopäätökset ja arviointi

Tässä luvussa arvioidaan opinnäytetyöprosessia ja laaditun ratkaisun toimivuutta. Luvussa käydään läpi prosessin aikana kohdattuja ongelmia ja arvioidaan oppimistavoitteiden ja opinnäytetyölle asetettujen tavoitteiden saavuttamista.

4.1 Opinnäytetyöprosessi

Opinnäytetyöprosessi dokumentoitiin melko hyvin, mutta työn eri vaiheet olisi voinut dokumentoida perusteellisemmin. Prosessin aikana pidettiin päiväkirjaa ja muistiinpanoja kirjattiin jatkuvasti ylös, kun uusia asioita tuli mieleen. Muistiinpanoja tehtiin myös erittäin paljon lähdemateriaalia läpikäydessä.

Opinnäytetyön suunnittelu aloitettiin hyvissä ajoin ja opinnäytetyöprosessin vaiheisiin tutustuttiin ennen kuin aihe oli valittu. Prosessin olisi kuitenkin voinut aikatauluttaa yksityiskohtaisemmin ja asettaa aikatauluun konkreettisempia tavoitteita.

Lähdemateriaali sisälsi runsaasti hyödynnettävää käytännön tietoa, jota voitiin soveltaa uhkien kartoittamisessa sekä ympäristön suojaamisessa. Lähdemateriaalissa esiintyi myös runsaasti opinnäytetyön tekijälle uutta tietoa, joten se myös auttoi oppimistavoitteiden saavuttamisessa. Tästä syystä koettiin, että lähdemateriaali valittiin onnistuneesti.

4.2 Ratkaisun arvioiminen

Tutkimuksellisen kehittämistyön konstruktiivisessa lähestymistavassa ratkaisun onnistuneisuutta arvioidaan sen kohdeorganisaatiolle tuoman käytännön hyödyn perusteella. Ratkaisua voidaan testata Ojasalon ym. kuvaamalla markkinatestillä. Heikon markkinatestin läpäisee, jos ratkaisu toimii kohdeorganisaatiossa käytännössä. Keskivahvan testin läpäisee, jos useampi organisaatio ottaa ratkaisun käyttöön. Vahvan markkinatestin läpäisee ainoastaan mikäli organisaatiot, jotka ovat ottaneet ratkaisun käyttöön, menestyvät entistä paremmin. (Ojasalo ym. 2009, 68.)

Opinnäytetyön ratkaisua on vaikea testata monipuolisin menetelmin, mutta se toimi käytännössä osana järjestelmän ohjelmistokehitysprosessia ja siitä on hyötyä kohdeyritykselle. Työ siis läpäisee ainakin heikon markkinatestin. Ratkaisu sisältää juuri kohdejärjestelmälle spesifistä tietoa, mutta käytännössä muissakin vastaavanlaisissa käyttöönottoprosesseissa voidaan hyödyntää ratkaisua ainakin osittain. Ratkaisua hyödyntäneet tahot voivat myös menestyä paremmin kuin ennen. On kuitenkin hyvin vaikea arvioida tuleeko näin tapahtumaan, joten keskivahvan ja vahvan markkinatestin läpäisyyn ei oteta kantaa.

4.3 Ongelmat

Ongelmia opinnäytetyöprosessiin toi se, että intranet-järjestelmä oli otettava käyttöön ennen työn valmistumista ja järjestelmän lopullista kokoonpanoa. Koska palvelin oli tuotantokäytössä, siihen oli riskialtista tehdä erilaisia konfiguraatiomuutoksia, sillä ne olisivat saattaneet aiheuttaa käyttökatkoksia palvelulle. Siksi välillä turvauduttiinkin käyttämään toista palvelinta, jolla erilaisia konfiguraatioita testattiin ennen niiden käyttöönottoa kohdejärjestelmässä.

Vaikeaksi koettiin opinnäytetyön tekijän olemassa olevan tiedon ja osaamisen hyödyntäminen, sillä osaamisen taustalla ei ollut kirjallisuutta, johon olisi voinut opinnäytetyössä viitata. Asioita olisi osattu tehdä oikein, mutta ratkaisuille ei ollut tukevia teoreettisia lähteitä. Suuri osa ajasta kuluikin lähteiden etsimiseen jo tälle olemassa olevalle tiedolle. Tutkimuksellisessa kehittämistyössä on tärkeää, ettei kehittäminen perustu tekijän perustelemattomiin päätelmiin, eli niin sanottuun arkiajatteluun. Tämänkaltaisessa työssä se voi kuitenkin paikoin olla hyvä lähestymistapa. (2009, 17.)

Yhteistyö ja vuorovaikutus toteuttajien ja hyödyntäjien kanssa tyypillisesti korostuvat tutkimuksellisessa kehittämisessä (2009, 18). Tämän näkökulman hyödyntäminen osoittautui kuitenkin ongelmalliseksi opinnäytetyön aihepiirin vuoksi, sillä palvelinympäristö ja sen toiminta eivät välity mitenkään loppukäyttäjälle muutamia poikkeuksia lukuun ottamatta, kuten esimerkiksi web-liikenteen salausta. Mutta kuten luvussa 1.2 mainittiin, myös opinnäytetyöntekijää voidaan pitää työn hyödyntäjänä, sillä hän on vastuussa koko ohjelmistokehitysprojektista. (2009, 66.)

Aiheeseen liittyvän lähdekirjallisuuden löytäminen oli melko vaikeaa. Esimerkiksi Laurea-ammattikorkeakoulun kirjastoista löytyi hyvin vähän aiheaiheeseen liittyvää kirjallisuutta. Käytetyn kirjallisuuden ajantasaisuus koettiin välillä myös puutteelliseksi, sillä ICT-ala kehittyy hyvin nopeasti. Tämä ilmenee esimerkiksi erilaisten ohjelmistojen ja teknologioiden vanhenemisena. Lähdekirjallisuuden sisällössäkin oli paikoin puutteita. Esimerkiksi Raimo Kosken kirjoittamien kirjojen (*Linux: Tehokas hallinta* ja *Linux: Ylläpitäjän käsikirja*) sisällön laatu koettiin erittäin heikoksi ja epäolennaiseksi.

4.4 Tavoitteiden saavuttaminen

Työn tavoitteena oli ottaa käyttöön intranet-järjestelmä ja varmistaa, että alustana toimiva palvelin toimii tietoturvallisesti. Lisäksi tavoite oli luoda käyttöönottodokumentaatio, jonka avulla edellä mainittu prosessi voidaan toistaa. Tavoitteena oli myös pyrkiä palvelimen tasolla vaikuttamaan web-sovelluksen tietoturvaan. Oppimistavoitteina oli kehittyä GNU/Linux-palvelinjärjestelmien ylläpitotehtävissä ja niihin liittyvissä tietoturva-asioissa.

Palvelin konfiguroitiin noudattamalla lähdekirjallisuuden muodostamaa tietoperustaa ja järjestelmän tietoturvallisuus voidaan myös osoittaa lähdemateriaalin teoriaan nojaten. Intranet-järjestelmä otettiin opinnäyteprosessin aikana käyttöön onnistuneesti ja on kirjoitushetkellä kohdeyrityksen aktiivisessa käytössä.

Oppimistavoitteissa onnistuttiin erittäin hyvin ja uutta osaamista kertyi runsaasti. Hankitusta web-palvelimen asennukseen, ylläpitoon ja tietoturvaan liittyvästä tiedosta muodostui opinnäytetyön tekijälle kokonaisvaltainen tietoperusta. Lisäksi AppArmor Mandatory Access Control –järjestelmän käyttöönotto ja *least privilege* –periaatteen hyödyntäminen koettiin opettavaiseksi. AppArmor-järjestelmään liittyvää osaamista onkin jo hyödynnetty opinnäytetyön tekijän muissa internet-palvelujen ylläpitotehtävissä.

Lähteet

About ClamAV®. 2013. Sourcefire. Viitattu 23.4.2013. http://www.clamav.net/lang/en/about/

About the Apache HTTP Server Project. 2012. The Apache Software Foundation. Viitattu 20.2.2013. https://httpd.apache.org/ABOUT_APACHE.html

Ahmad, D., Dubrawsky, I., Flynn, H., Grand, J., Graham, R., Johnson, N., Kaminsky, D., Lynch, F., Manzuik, S., Permeh, R., Pfeil, K., Puppy, R. & Russel, R. 2002. Hack Proofing Your Network – The Only Way to Stop sa Hacker is to Think Like One. Syngress Publishing.

April 2013 Web Server Survey. 2013. Netcraft. Viitattu 5.4.2013. http://news.netcraft.com/archives/2013/04/02/april-2013-web-server-survey.html

Barnett, R. 2006. Preventing Web Attacks with Apache (Kindle Edition). Addison Wesley Professional.

Bauer, M. 2003. Building Secure Servers with Linux. O'Reilly & Associates. Viitattu 7.4.2013. http://my.safaribooksonline.com/book/operating-systems-and-server-administration/linux/0596002173

Bauer, M. 2006. Paranoid Penguin - An Introduction to Novell AppArmor. Linux Journal. Viitattu 24.3.2013. http://www.linuxjournal.com/article/9036

Bauer, M. 2007. Paranoid Penguin - Introduction to SELinux. Linux Journal. Viitattu 24.3.2013. http://www.linuxjournal.com/article/9500

Bauer, M. 2009. Paranoid Penguin - AppArmor in Ubuntu 9. Linux Journal. Viitattu 24.3.2013. http://www.linuxjournal.com/magazine/paranoid-penguin-apparmor-ubuntu-9

Boelen, M. 2009. Lynis – Documentation. Viitattu 11.5.2013. http://www.rootkit.nl/files/lynis-documentation.html

The CentOS Project. 2013. Viitattu 22.2.2013. http://wiki.centos.org/Download

CVE-2012-1823. 2012. The MITRE Corporation. Viitattu 14.4.2013. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2012-1823

Debian Releases. 2013. The Debian Project. Viitattu 8.5.2013. http://wiki.debian.org/DebianReleases

Download Ubuntu Server. 2013. Canonical Ltd. Viitattu 22.2.2013. http://www.ubuntu.com/download/server

Dwivedi, H. 2004. Implementing SSH: Strategies for Optimizing the Secure Shell. Indianapolis: Wiley Publishing.

Eindbazen PHP-CGI advisory. 2012. Eindbazen. Viitattu 14.4.2013. http://eindbazen.net/2012/05/php-cgi-advisory-cve-2012-1823/

Enterprise End User Report. 2013. The Linux Foundation. Viitattu 28.3.2013. http://www.linuxfoundation.org/publications/linux-foundation/linux-adoption-trends-end-user-report-2013

Gagné, M. 2003. Linux: Järjestelmänhaltijan käsikirja. Helsinki: Edita Prima.

Garfinkel, S., Schwartz, A. & Spafford, G. 2003. Practical Unix & Internet Security (Kindle Edition). O'Reilly Media.

Gelbmann, M. 2012. W3Techs. Viitattu 1.3.2013. http://w3techs.com/blog/entry/debian_is_now_the_most_popular_linux_distrribution_on_web_servers

Goodin, D. 2011. Kernel.org Linux repository rooted in hack attack. The Register. Viitattu 11.5.2013. http://www.theregister.co.uk/2011/08/31/linux_kernel_security_breach/

Hacking Exposed Linux - Linux Security and Solutions. 2008. Institute for Security and Open Methodologies. The McGraw-Hill Companies.

Haeder, A., Addison Schneiter, S., Gomes Pessanha, B., Stanger, J., 2010. LPI Linux Certification in a Nutshell. Sebastopol: O'Reilly Media.

Hodges, J., Jackson, C. & Barth, A. 2012. HTTP Strict Transport Security (HSTS). Internet Engineering Task Force. Viitattu 4.5.2013. https://tools.ietf.org/html/rfc6797

IT-Grundschutz Kataloge. 2005. Bundesamt für Sicherheit in der Informationtechnik. Viitattu 20.2.2013. https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Grundschutz/download/it-grundschutz-kataloge_2005_pdf_en_zip.zip?__blob=publicationFile

KATAKRI. 2011. Puolustusministeriö. Viitattu 4.5.2013. http://www.defmin.fi/files/1870/KATAKRI_versio_II.pdf

Koski, R. & Kajala, T. 2005. Linux: Ylläpitäjän käsikirja.

Koski, R. 2010. Linux: Tehokas Hallinta. Readme.fi.

Linux Adoption Trends 2012: A Survey of Enterprise End Users. 2012. The Linux Foundation. Viitattu 30.1.2013. http://www.linuxfoundation.org/publications/linux-foundation/linux-adoption-trends-end-user-report-2012

Linux Jobs Report: Pressing Need for Linux Talent. 2013. The Linux Foundation. Viitattu 20.2.2013. http://www.linuxfoundation.org/publications/linux-foundation/2013-linux-jobs-report

Manadhata, P. & Wing, J. 2004. Measuring a System's Attack Surface. School of Computer Science Carnegie Mellon University. Viitattu 4.4.2013. http://www.dtic.mil/cgi-bin/GetTRDoc?Location=U2&doc=GetTRDoc.pdf&AD=ADA458115

McClure, S., Scambray, J. & Kurtz, G. 2009. Hacking Exposed 6: Network Security Secrets & Solutions. The McGraw-Hill Companies.

Miller, S. 2012. The Importance of Securing a Linux Web Server. Infosec Institute. Viitattu 5.5.2013. http://resources.infosecinstitute.com/securing-linux-web-server/

MySQL Language Reference. 2005. MySQL AB. Seattle: MySQL Press.

Nixon, R. 2012. Learning PHP, MySQL, JavaScript & CSS. Sebastopol: O'Reilly Media.

Novell AppArmor Administration Guide. 2007. Novell, Inc. Viitattu 28.4.2013. http://www.novell.com/documentation/apparmor/pdfdoc/apparmor201_sp10_admin/apparmor201_sp10_admin.pdf

Ojasalo, K., Moilanen, T. & Ritalahti, J. 2009. Kehittämistyön menetelmät. Helsinki: WSOYpro.

Rantala, A. 2003. Linux. Porvoo: WS Bookwell.

Stallman, R. 2013. Linux and the GNU System. Viitattu 3.2.2013. https://www.gnu.org/gnu/linux-and-gnu.html

Stanger, J., Lane, P. & Danielyan, E. 2001. Hack Proofing Linux: A Guide to Open Source Security. Syngress Publishing.

Turnbull, J. 2005. Hardening Linux. Apress.

Ubuntu Official Documentation: AppArmor. Canonical Ltd. Viitattu 22.2.2013. https://help.ubuntu.com/12.04/serverguide/apparmor.html

Usage statistics and market share of Apache for websites. 2013. Web Technology Surveys. Viitattu 7.3.2013. http://w3techs.com/technologies/details/ws-apache/all/all

Valtiohallinnon tietoturvasanasto. 2008. Valtiovarainministeriö. Viitattu 7.5.2013. http://www.vm.fi/vm/fi/04_julkaisut_ja_asiakirjat/01_julkaisut/05_valtionhallinnon_tietoturvallisuus/20081211Valtio/Vahti_8_NETTI%2b_KANNET.pdf

Vilkka, H. & Airaksinen, T. 2003. Toiminnallinen opinnäytetyö. Jyväksylä: Gummerus Kirjapaino.

Why Suhosin?. Hardened PHP Project. Viitattu 15.4.2013. http://www.hardened-php.net/suhosin/why.html

What is Suhosin?. Hardened PHP Project. Viitattu 15.4.2013. http://www.hardened-php.net/suhosin/#suhosin

Kuvat

Kuviot

Liitteet

Liite 1: Käyttöönottodokumentaatio

Table of contents

1 About this document

The purpose of this document is to provide the information needed to deploy the [name removed] intranet system in a secure server environment. Please note that the deployment requires experience in Unix-like systems and general knowledge of computer systems and networking technologies.

This document is designed for deployment on dedicated servers but it might prove useful in other environments as well, such as Virtual Private Servers on the Amazon Elastic Compute Cloud, for example. It also provides enough information for you to be able to deploy the system on a web hosting service.

In the examples of this document, Vim text editor is used for text editing. You may use the cli text editor of your preference (even Emacs).

1.1 About the symbols used

$ anything after a dollar sign is a command that is to be run on the command-line shell.
- anything after a minus sign is to be removed (from a configuration file, for example).
+ anything after a plus sign is to be added.
<> Anything between these symbols should be substituted with something.

For example:
The following line means that you are supposed enter the *vim* command into to the command-line shell interface (terminal).

```
$ vim my_name.txt
```

The following lines mean that you are supposed to use the text editor to modify the line that states your name.

```
- My name is Peter
+ My name is John
```

The following line means that you add a completely new line into the file.

```
+ My last name is Smith
```

The following example describes how you are supposed to substitute words between the "<>" signs with information.

```
- My name is <your_name>
+ My name is John
```

2 The operating system

The recommended operating system is Ubuntu Server because this manual includes some Ubuntu-specific instructions, mainly regarding the Mandatory Access Control system. If Ubuntu Server is unavailable, another Debian-based operating system is preferred (because of usage the *apt* packaging tool).

2.1 Installing the operating system on the server

The installation process of the operating system should be relatively straightforward. Follow the instructions provided by the server provider. You can also refer to the Ubuntu installation guide or the installation guide of your alternative operating system.

2.2 Creating an administrator user account

It is considered good practice to use a normal user with *sudo* rights instead of using the root account for day-to-day work.

Create the new user and add it to the *sudo* group.

```
$ sudo adduser <user_name>
```

Add the user to the groups "*sudo*" and "*adm*".

```
$ sudo usermod -a -G sudo,adm <user_name>
```

2.3 Installing a generic Linux kernel

The server operating system might have a custom Linux kernel installed by default. For example, OVH hosting is using a custom GrSecurity kernel to improve the security of the system. Using the custom kernel might make the installing of AppArmor more difficult as the kernel might not have the necessary modules. It is advised to install the generic Linux kernel provided by the distribution instead.

You can check the kernel in use with the following command.

```
$ uname -r
```

If the operating system is not running a generic kernel, download and install one.

```
$ sudo apt-get install linux-kernel-generic
```

Backup (move) the custom kernel to the home folder of the current user.

```
$ sudo mv /boot/<custom_kernel_name> ~
```

Update the GRUB bootloader configuration file so that the generic kernel is chosen at boot time.

```
$ sudo update-grub
```

The system has to be restarted for the changes to take effect. The following command restarts the computer.

```
$ sudo /etc/init.d/shutdown -r now
```

3 Remote administration

For remote administration it is recommended to use the SSH protocol and software. The advantage of SSH compared to other protocols is security. For example, all SSH traffic is encrypted. SSH server software should be installed by default on the server. See this list for available SSH client software. On Windows a good choice is the Putty SSH client. On OS X and GNU/Linux systems you can use the native command-line SSH clients.

On your client system (the computer you use to access the server) use the following command to log into the server (or use the graphical interface of your SSH client). The server will ask for a password, which has probably been provided to you by the server provider.

```
$ ssh root@<server_address>
```

3.1 SSH Public Key authentication

For security reasons it is advised to use public key authentication instead of using passwords to log in to the server. Before you configure the SSH server, make sure that you have created, or that already possess, an RSA key pair. Also, make sure that you have created the administrator user account and

that you are able to access the server with that account via SSH. Make sure you can obtain root privileges with that user account. Otherwise, you might lock yourself out of the system.

3.2 Creating RSA keys

To access the server without a password, you need to create an RSA key pair on the client machine that you use to access the server. It is crucial that these keys are backed up and their confidentiality is not lost.

RSA keys can be created in a Linux with the following command.

```
$ ssh-keygen -t rsa <key_name>
```

Then enter a passphrase to encrypt the private key file. **Do not forget this password** or you will get locked out of the server.

Use the following command to enable logging in to the server with the newly created key. This adds your public key into /home/<user_name>/.ssh/authorized_keys.

```
$ ssh-copy-id -i <key_name> <user_name>@<server_address>
```

3.3 Configuring the SSH server

For security reasons it is recommended to disable password authentication for SSH. It can be disabled by editing the SSH server configuration file.

```
$ sudo vim /etc/ssh/sshd_config
        -PasswordAuthentication yes
        +PasswordAuthentication no
```

Also verify that UsePrivilegeSeparation is turned on. This improves the security of the SSH process.

```
        -UsePrivilegeSeparation no
        +UsePrivilegeSeparation yes
```

Disable root login.

```
        -PermitRootLogin yes
        +PermitRootLogin no
```

Note: Only do this if you have created the administrator user account (see section 2.2).

Only allow the administrator user account to log in.

```
        +AllowUsers <user_name>
```

To enable the new configuration you need to restart the SSH process.

```
$ sudo service ssh restart
```

At this point make sure you are able to log in with <user_name>. Also ensure you are able to obtain root privileges with the following command.

```
$ sudo -i
```

3.4 Brute-force protection for the SSH server

To protect the SSH server against SSH brute-force attacks, you can use the Denyhosts package. Denyhosts blacklists IP addresses that have too many consecutive failed login attempts. Using Denyhosts is not crucial since disabling password authentication thwarts these kinds of attacks quite effectively. However, it does add another layer of security.

```
$ sudo apt-get install denyhosts
```

4 Updating the system

As soon as the server operating system has been setup, enter the following commands to update all the packages in the system.

First resynchronize the package index.

```
$ sudo apt-get update
```

Then use apt-get with dist-upgrade parameter to upgrade to the newest version of the Linux kernel as well. Note that upgrading the kernel requires restarting of the system.

```
$  sudo apt-get dist-upgrade
```

4.1 Automating the update process

It is highly recommended to enable automatic updates so that security updates are installed without delay.

First download and install the package the handles the automatic updates, *unattended-upgrades*.

```
$ sudo apt-get install unattended-upgrades
```

Then configure the package with the following command. When prompted, choose yes.

```
$ sudo dpkg-reconfigure unattended-upgrades
```

The system can be updated manually with the following commands.

```
$ sudo apt-get update && sudo apt-get upgrade
```

5 Configuring packet filtering with UFW

Configure the firewall to accept SSH connections (tcp port 22) from any IP address.

```
$ sudo ufw allow from any to any port 22 proto tcp
```

Enter the following command to see the firewall rules and verify that the entry is correct.

```
$ sudo ufw status
```

There should be the following line:

```
"22/tcp  ALLOW          Anywhere"
```

Now set the firewall to block anything that does not meet the defined rules.

```
$ sudo ufw default deny
```

Now enable ufw to start the firewall.

```
$ sudo ufw enable
```

At this point try logging in with another terminal session. This is crucial because you might be locked out of the system if the configuration is wrong. If you are unable to log in, review the firewall configuration.

Now allow traffic to tcp ports 80 (http) and 443 (https) to enable browsers to access the web server.

```
$ sudo ufw allow from any to any port 80 proto tcp
$ sudo ufw allow from any to any port 443 proto tcp
```

6 Installing the Apache web server and modules

Install Apache web server.

```
$ sudo apt-get install apache2
```

Install the PHP module for PHP support.

```
$ sudo apt-get install libapache2-mod-php5
```

Install the mod-security web application firewall module.

```
$ sudo apt-get install libapache2-modsecurity
```

Install the mod-evasive module for (HTTP) brute-force protection.

```
$ sudo apt-get install libapache2-mod-evasive
```

Install the mod-apparmor module to enable the confining of Apache sub-processes.

```
$ sudo apt-get install libapache2-mod-apparmor
```

6.1 Configuring the Apache server

Whenever you modify the Apache configuration file, you should verify that there are no syntax errors in the configuration file before you restart the Apache process. Otherwise Apache will fail to launch and this will cause downtime for the intranet system. Use the following command to check the syntax of the configuration file.

```
$ sudo apachectl configtest
```

Note that changes in the configuration require the Apache process to be reloaded. This can be done with the following command.

```
$ sudo service apache2 force-reload
```

It is recommended that the server does not broadcast unnecessary information about the system. Add the following lines to the bottom of the Apache configuration file.

```
$ sudo vim /etc/apache2/apache2.conf
        + ServerTokens Prod
        + ServerSignature Off
        + TraceEnable Off
        + Header unset ETag
        + FileETag None
```

6.2 Configuring PHP

It is important to install and enable the Apache PHP module before moving the .php files into the root directory of the web server. Otherwise the PHP code is not handled correctly and the source code of the intranet system will be exposed.

Enable the PHP module.

```
$ sudo a2enmod php5
```

Modifications in the configuration of PHP are required for the upload feature of the intranet system to function. It is also advised to disable the printing of detailed error messages and version information.

Edit the PHP configuration.

```
$ sudo vim /etc/php5/apache2/php.ini
```

Make sure file uploading is enabled as it is required by the upload feature.

```
- file_uploads = Off
+ file_uploads = On
```

Also allow the uploading of files larger than the default of 2 megabytes.

```
- upload_max_filesize = 2M
+ upload_max_filesize = 64M
```

Disable the output of errors and version information.

```
- expose_php = On
+ expose_php = Off
- display_errors = On
+ display_errors = Off
- track_errors = On
+ track_errors = Off
- html_errors = On
+ html_errors = Off
```

Install php5-suhosin to add another layer of security for PHP.

```
$ sudo apt-get install php5-suhosin
```

6.3 Configuring SSL encryption

Create a directory for the SSL certificate files in the Apache folder.

```
$ mkdir /etc/apache2/ssl
```

Move the SSL certificate files (from the current directory).

```
$ sudo mv apache.key gandi.crt GandiStandardSSLCA.pem /etc/apache2/ssl/
```

Configure Apache to use these certificate files.

```
$ sudo vim /etc/apache2/sites-enabled/default-ssl
    - SSLCertificateFile /etc/ssl/certs/ssl-cert-snakeoil.pem
    + SSLCertificateFile /etc/apache2/ssl/gandi.crt
    - SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key
    + SSLCertificateKeyFile /etc/apache2/ssl/apache.key
    + SSLCACertificateFile /etc/apache2/ssl/GandiStandardSSLCA.pem
```

Enable the SSL module to enable SSL encryption and restart Apache.

```
$ sudo a2enmod ssl && sudo service apaceh2 restart
```

6.4 Forcing encrypted (HTTPS) connections

Enable the rewrite module to enable the forcing of encrypted connections by rewriting HTTP headers.

```
$ sudo a2enmod rewrite
```

Copy the default SSL configuration file.

```
$ sudo cp /etc/apache2/sites-available/default-ssl /etc/apache2/sites-enabled/
```

Delete (move) the default HTTP configuration file.

```
$ sudo mv /etc/apache2/sites-enabled/000-default /etc/apache2/000-default.bak
```

Edit the Apache configuration file and add the following to the beginning of the file.

```
$ sudo vim /etc/apache2/sites-enabled/default-ssl
        + <VirtualHost *:80>
            + RewriteEngine on
             + RewriteCond %{HTTP_HOST} ^www\.(.*)$ [NC]
             + RewriteRule ^(.*)$ http://%1/$1 [R=301,L]
             + RewriteRule .* https://%{HTTP_HOST} [R,L]
             + ServerAdmin root@localhost
        + </VirtualHost>
```

Now all non-encrypted connections (port 80) are converted into encrypted connections (port 443).

6.5 HTTP Strict Transport Security

The HTTP Strict Transport Security protocol (HSTS) instructs the client browsers to always use encryption when communicating with a web server. This protects against SSL stripping attacks. The *headers* module is required to implement the HSTS protocol.

Enable the *headers* module.

```
$ sudo a2enmod headers
```

Add the following line under "<VirtualHost _default_:443>" in the ssl-default file.

```
$ sudo vim /etc/apache2/sites-enabled/default-ssl
        + Header add Strict-Transport-Security "max-age=15768000;includeSubDomains"
```

6.6 Apache Access Control

There are certain files and folders in the web server root directory that should not be accessible via the web server, such as the *.git* folder. Also the listing of folder contents should be disabled (*-Indexes*). Add the following lines under "<VirtualHost _default_:443>":

```
$ sudo vim /etc/apache2/sites-available/default-ssl
        +<Directory /var/www/>
                +Options -Indexes -FollowSymLinks -MultiViews
                +AllowOverride None
                +Order allow,deny
                +allow from all
                +<Files generator.php>
                +deny from all
                +</Files>
        +</Directory>
        +<Files /var/www/generator.php>
                +Order allow,deny
                +deny from all
        +</Files>
        +<Directory /var/www/files/>
                +Order deny,allow
                +deny from all
        +</Directory>
        +<Directory /var/www/.git/>
                +Order deny,allow
                +deny from all
        +</Directory>
        +<Directory /var/www/.settings>
                +Order deny,allow
                +deny from all
        +</Directory>
```

Disable access to user directories since this feature is not needed.

```
$ sudo vim /etc/apache2/apache2.conf
        +UserDir disabled # There is no need for Apache to use user directories
```

6.6.1 File access permissions

Use chmod to set file access rights of the files and folders in the web server root directory. Directories should be 750 (drwxr-x---) and files should be 640 (-rw-r-----).

```
$ chmod 750 <folder>
$ chmod 640 <file>
```

Change the owner of the web application files of the intranet system. Owner should be the <user_name> and group should be www-data.

```
$ chown <username>:www-data <file_name>
```

You can use a wildcards to speed up this process. The following command changes the owner and group of every php file in the current folder.

```
$ chown <username>:www-data ./*.php
```

The access permissions for the *.git* folder should be different. For security reasons the web server process should not be able to access this folder.

```
$ chown -R <username>:<username> .git
$ chmod -r 700 .git
```

The "files" folder should be owned by the *www-data* user, so that the web server process is able to add uploaded files into the folder.

```
$ sudo chown www-data /var/www/files
```

6.7 Apache Security modules

The security of the Apache web server can be improved with several modules. This section covers usage of AppArmor, ModSecurity and Evasive modules.

The AppArmor module makes it possible to confine the web application from the system, which improves the security.

```
$ sudo a2enmod apparmor
```

To enable ModSecurity you need to rename the default configuration file.

```
$ cd /etc/modsecurity/
$ sudo cp modsecurity.conf-recommended modsecurity.conf
```

Then edit the configuration file.

```
$ sudo vim modsecurity.conf
        - SecRuleEngine DetectionOnly
        + SecRuleEngine On
```

Then enable the module.

```
$ sudo a2enmod mod-security
```

Mod-evasive provides protection against HTTP brute-force and denial-of-service attacks.

Create and edit the mod-evasive configuration file.

```
$ sudo vim /etc/apache2/mods-enabled/mod-evasive.conf
        +<IfModule mod_evasive20.c>
            +DOSHashTableSize 3097
            +DOSPageCount 5
            +DOSSiteCount 50
            +DOSPageInterval 1.0
            +DOSSiteInterval 1.0
            +DOSBlockingPeriod 5.0
            +DOSLogDir /var/lock/mod_evasive
            +DOSEmailNotify root@localhost
        +</IfModule>
```

Enable mod-evasive.

```
$ sudo a2enmod mod-evasive
```

7 Installing and configuring MySQL

At the time of writing, it seems that MariaDB might replace MySQL as the de facto open source database solution in the future. As MariaDB is API compatible with MySQL this should not cause problems. If you wish to use MariaDB instead of MySQL refer to the MariaDB documentation.

First install the MySQL server software on the new server.

```
$ sudo apt-get install mysql-server
```

The installer will ask for a root user password. Choose a strong, preferably randomly generated password. Memorise the password or store it in an encrypted password database, such as KeePass.

7.1 Exporting the database from the old system

Export the database from the **old** system with the following command.

```
$ mysqldump -u <database_user> -p<password> --single-transaction <database_name> > full-database.sql
```

Note: There is no space between the password and the '-p' parameter.

Transfer the database file to the new system. To accomplish this, command-line tools such as scp or GUI solutions such as FileZilla can be used.

Transfer the database file to the target server using *scp* with the following command.

```
$ scp full-database.sql <user_name>@<new_server_address>:/home/<user>/
```

Next you need to create the database in the **new** system, so you can import the data and structure of the old database.

```
$ mysql -u root -p<mysql_root_password>
```

Note: After this command everything will be entered into the mysql command-line interface. Enter only the text that is after "mysql>".

Create the database. Use the same name as the old one.

```
mysql> CREATE DATABASE <database_name>;
```

Create the user and allow it to connect to the database server. The password should be the same as the old one.

```
mysql> GRANT USAGE ON *.* to <database_user>@localhost identified by '<database_user_password>';
```

Now give the newly created user all privileges to the database.

```
mysql> GRANT ALL PRIVILEGES ON <database_name>.* TO <database_user>@localhost ;
```

7.2 Importing the database intranet database

Import the database structure and data using the mysql command.

```
$ mysql -u root -p<password> <database_name> < full-database.sql
```

Note: There is no space between the password and the -p attribute

8 Configuring AppArmor Mandatory Access Control

If you are running an operating system other than Ubuntu you might want to skip this section and refer to the documentation of your distribution. For example, Red Hat based distributions usually have SELinux Mandatory Access Control installed by default.

In Ubuntu AppArmor is enabled by default but few profiles are in enforce mode. To control AppArmor two packages are needed.

```
$ sudo apt-get install apparmor-profiles apparmor-utils
```

8.1 Confining the Apache and MySQL processes

Set Apache and MySQL processes into enforce mode.

```
$ sudo aa-enforce /etc/apparmor.d/usr.lib.apache2.mpm-prefork.apache2
$ sudo aa-enforce /usr/sbin/mysqld
```

Note: AppArmor Apache module must be enabled before Apache can be set into enforce mode.

Now define an Apache sub-profile for the web application itself. This profile defines all the files the web application can access in the filesystem.

```
$ sudo vim /etc/apparmor.d/apache2/spurdo
    +^spurdo {
        +#include <abstractions/apache2-common>
        +#include <abstractions/base>
        +#include <abstractions/php5>
        +#include <abstractions/mysql>
        +/var/log/apache2/access.log w,
        +/var/log/apache2/error.log w,
        +/var/log/apache2/ssl_access.log w,
        +/var/www/ r,
        +/var/www/** r,
        +/var/www/files/* w,
        +/tmp/** rw,
        +/var/log/apache2/modsec_audit.log w,
    +}
```

Configure Apache to use this AppArmor profile for the default virtual host.

```
$ sudo vim /etc/apache2/sites-enabled/default-ssl
```

Add the following line below "<VirtualHost _default_:443>" and save the file.

```
    +AADefaultHatName spurdo
```

Run the following commands for the changes to take effect.

```
$ sudo apparmor_parser -r /etc/apparmor.d/usr.lib.apache2.mpm-prefork.apache2
$ sudo service apache2 restart
```

8.2 Confining other processes with AppArmor

For clarity, disable the AppArmor profiles for programs that are not installed so that they are not displayed when using the *aa-status* command.

```
$ cd /etc/apparmor.d/
$  sudo mv usr.lib.dovecot.* usr.sbin.dovecot usr.sbin.nmbd usr.sbin.smbd usr.sbin.dnsmasq
usr.sbin.named usr.sbin.identd usr.sbin.mdnsd usr.sbin.nscd disable/
```

Set other processes into enforce mode as well.

```
$ sudo aa-enforce ping freshclam ntpd traceroute tcpdump
```

Do not enforce the following profiles on a production system as it might lock you out of the system.

```
$ sudo aa-enforce dhclient /usr/sbin/sshd
```

Note: Some of the profiles might need to be moved from the "extras" folder for them to work.

```
$ sudo cp /usr/share/doc/apparmor-profiles/extras/<profile_name> /usr/apparmor.d/
```

If AppArmor causes an application to malfunction, you can temporarily disable a profile by setting it into complain mode.

```
$ sudo aa-complain <profile_name>
```

You can also use the *aa-logprof* command to reconfigure a profile. *aa-logprof* scans the system log files for AppArmor messages and enables you to modify AppArmor profiles based on these messages.

```
$ sudo aa-logprof
```

Alternatively, you can set an enforced profile temporarily into complain mode and then modify its profile based on the process' activities.

```
$ sudo aa-genprof <path_to_executable>
```

At least the profile for *freshclam* requires modification for it to function properly. Set *freshclam* into complain mode and run the *freshclam* command in another terminal session.

```
$ sudo aa-genprof
$ sudo freshclam
```

For more detailed instructions on how to modify AppArmor profiles refer to the AppArmor Administration Guide.

9 Revision control with Git

Git is used for revision control in the development of the intranet system. For this reason it should be installed on the server. The web server root folder contains the *.git* folder, which holds all the revision control data.

Install the Git revision control system.

```
$ sudo apt-get install git
```

10 Installing the web application

Copy all the files into the root directory of your web server (usually /var/www/) or use Git to clone the repository from the old server.

```
$ cd /var/www/ # (or the web server root folder)
$ git clone ssh://user@old.server.example.com/var/www/ .
```

Also test that uploaded files are not world-accessible.

```
$ wget new.server.com/files/<legitimate_file_name>
```

If you are able to download a file, you need to review the access control list of Apache. Repeat this process until you get a 403 HTTP error (forbidden).

11 Backing up the system

Backing up the intranet system and especially its database is a very important precaution. Failing to do so might cause the loss of crucial data due to incidents such as hard drive failure.

11.1 Database backup scripts

First create the local backup scripts on the target server. Create the backup directory you're your home folder.

```
$ mkdir ~/backup
```

Create the database backup script file. The script takes one parameter, "daily" or "hourly" and conducts either an hourly or a daily backup based on the input.

```
$ vim database-backup.bash
+ #!/bin/bash
+ # Copyright by Veli-Matti Kivinen 2013
+ if [ -z $1 ]
+ then
+       echo "Error  no parameter given."
+       exit 2
+ elif [ $1 = "hourly" ]
+ then
+       mysqldump -u <database_user> -p<database_password> <database_name> >
~/backup/database/hourly/$(date +%H).sql
+ elif [ $1 = "daily" ]
+ then
+       mysqldump -u <database_user> -p<database_password> <database_name>  >
~/backup/database/daily/$(date +%C%y-%m-%d).sql
+ else
+      echo -e 'Invalid parameter.\nPossible paramaters are "hourly" and "daily"'
+       exit 2
+ fi
```

Note: Remember to fill in the database name, user and password.

Create two cronjobs so that the script will run automatically every hour and do an additional backup once a day.

```
$ crontab -e
+ 0 * * * * bash ~/database-backup.bash hourly
+ 30 12 * * * bash ~/database-backup.bash daily
```

11.2 Backing up configuration files

You also should backup at least the Apache configuration folder, AppArmor profiles and crontabs. The following script can be used to handle this.

Create the script file.

```
$ vim ~/local-backup.bash
+ #!/bin/bash
+ # Copyright by Veli-Matti Kivinen 2013
+ # backup apache folder
+ rsync -ar /etc/apache2/ ~/backup/apache/$(date +%C%y-%m-%d)/ >>
~/backup/local-backup.log 2>&1
+ # backup apparmor profiles
+ rsync -ar /etc/apparmor.d/  ~/backup/apparmor/$(date +%C%y-%m-%d)/ >>
/backup/local-backup.log 2>&1
+ # backup crontabs
+ rsync -ar /var/spool/cron/crontabs/    ~/backup/crontabs/$(date +%C%y-%m-%d)/
>> ~/backup/egrs-local-backup.log 2>&1
```

Again, create a cronjob to be run once a day.

```
$ sudo crontab -e
+ 0 13 * * * bash ~/local-backup.bash
```

11.3 Off-site backups

It is highly recommended to do off-site backups as well. The target server might lose all of its data due to forgetting to pay the server rental bill, for example.

This setup essentially requires at least one additional server that is located in another location as the target server. For example, you can use a Raspberry Pi mini-computer as a cheap backup server. Also, you might want to consider using the Amazon EC2 free tier for this setup.

12 Protecting the system against malware

12.1 Scanning uploaded files

Install ClamAV anti-malware software and unofficial signatures to improve malware detection rate.

```
$ sudo apt-get install clamav clamav-unofficial-sigs
```

Create two cronjobs; one for updating the malware signatures and the other for scanning the upload folder of the intranet system. If malware is found it is moved to the quarantine folder.

```
$ mkdir ~/quarantine
$ sudo crontab -e
        + 20 20 * * * freshclam
        + 10 10 * * * nice -n 19 clamscan --no-summary -r --infected --move=/home/<user>/
        /var/www/files
```

Note: You might need to edit freshclam's AppArmor profile for it to work. Use aa-logprof if the freshclam command fails.

The definitions will be updated at 8:20pm and the scan will occur at 10:10am. If the scanner finds malware, the root user will be notified by mail. The *nice* command is used to lower the priority of the clamscan process so that the scanning does not affect the performance of the web application.

12.3 Rootkit hunter

It is recommended to scan the system for rootkits and other malware with a tool such as Rootkit Hunter. Install Rootkit Hunter and configure it to run automatically daily.

```
$ sudo apt-get install rkhunter
$ sudo dpkg-reconfigure rkhunter
```

After the last command a series of questions will be asked. Answer "yes" to all questions (run daily, auto-update rkhunter's database and auto-update rkhunter's file properties database). A summary for the daily scans will be sent to the root user.

For more information refer to the rkhunter manual.

```
$ man rkhunter
```

12.3 chkrootkit

chkrootkit can also be used to scan for malware. Again, configure the application to run daily with the dpkg-reconfigure command.

```
$ sudo apt-get install chkrootkit
$ sudo dpkg-reconfigure chkrootkit
```

When prompted, choose "yes", "ok" and "no".

For more information refer to the official manual.

```
$ man chkrootkit
```

12.4 Debsums

Debsums is used to check for changes in the files of installed packages. It can play a role in intrusion detection by detecting modified binary and configuration files.

Install debsums.

```
$ sudo apt-get install debums
```

Run Debsums checking all configuration (-a) files and only print out errors (-s).

```
$ sudo debsums -as
```

For more information refer to the manual.

```
$ man debums
```

12.5 Lynis

Lynis is a system auditing tool. It detects insecure configurations and provides instructions on how to improve the security of the system.

Install Lynis.

```
$ sudo apt-get install lynis
```

Run Lynis without waiting for user input (--quick).

```
$ sudo lynis --quick --checkall
```

For more information refer to the manual. Or the official website.

```
$ man Lynis
```

13 Testing the firewall with Nmap

It is recommended to test your firewall with a portscanner such as Nmap. The port scan has to be conducted from a remote client machine. There are graphical frontends available, such as Zenmap.

Install Nmap and run a port scan.

```
$ sudo apt-get install nmap
$ sudo nmap <server_address>
```

Only ports 22, 80 and 443 should be open.

A more comprehensive scan can be conducted with parameters such as the following (as de-fined by Zenmap's "intensive scan".

```
$ sudo nmap -T4 -A -v -PE -PS22,25,80 -PA21,23,80,3389 <server_address>
```

For more information refer to the Nmap reference guide.