# TURVALLINEN PALVELIN-YMPÄRISTÖ

Jonne Heiskanen

Opinnäytetyö
Toukokuu 2015
Tietojenkäsittelyn
koulutusohjelma

TAMPEREEN AMMATTIKORKEAKOULU
Tampere University of Applied Sciences

**TIIVISTELMÄ**

Tampereen ammattikorkeakoulu
Tietojenkäsittelyn koulutusohjelma

HEISKANEN, JONNE:
Turvallinen palvelinympäristö

Opinnäytetyö 40 sivua
Toukokuu 2015

---

Tämän opinnäytetyön tavoitteena oli selvittää, mitä vaatimuksia ja toimenpiteitä tietoturvallinen palvelinympäristö sisältää verkko- ja palvelintasolla. Toimeksiantajana toimi kirjoittajan nykyinen työnantaja Solita Oy.

Kerätyn tiedon pohjalta Solitalle tehtiin tarkempi ehdotus siitä, kuinka sen nykyinen palvelinympäristö saadaan turvallisemmaksi ja vastaamaan yrityksen tietoturvapolitiikkaa. Vaatimuksina oli, että nykyisen ympäristön muuttaminen tietoturvalliseksi hyödyntäisi mahdollisimman paljon automatiikkaa ja että ehdotusta voidaan hyödyntää sekä uusiin että vanhoihin palvelimiin.

Opinnäytetyön lopputuloksena syntyi ehdotus vaadittavista toimenpiteistä sekä kattava Ansible-playbook, jonka avulla yrityksen uudet ja vanhat palvelimet saadaan automatisoidusti toivottuun tilaan.

Opinnäytetyön avulla lukija saa yleiskuvan turvallisesta palvelinympäristöstä ja sen komponenteista sekä siihen liittyvistä järjestelmistä.

---

Asiasanat: linux, palvelin, turvallinen, tietoturva, Ansible

**ABSTRACT**

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Degree Programme in Business Information Systems

HEISKANEN, JONNE:
Secure Server Environment

Bachelor's thesis 40 pages
May 2015

---

This thesis was commissioned by Solita Ltd. and the purpose was to research what procedures and actions a secure server environment requires, at both network and server levels.

Based on the source material researched for this project, a guide was created for the company explaining how its current server environment could be made more robust. The company required that the transition to a secure environment must be done as automatically as possible and that it should be compatible with both new and existing servers.

A guide was created explaining the steps required for the transition, as well as a comprehensive Ansible Playbook, which makes it possible to convert the company's servers, both existing and new, automatically to the desired state.

This thesis leaves the reader with a general picture of a safe server environment, its components and the systems attached to it.

---

Key words: Linux, server, secure, information security, Ansible

**SISÄLLYS**

**LYHENTEET JA TERMIT**

| | |
|---|---|
| VLAN | Virtual Local Area Network |
| VID | VLAN identifier |
| YAML | Yet Another Markup Language |
| SSH | Secure Shell |
| HTTPS | Hypertext Transfer Protocol Secure |
| Muuttuja | Tiedon varastointipaikka |
| Tarkistussumma | Tiedoston tarkistuskoodaustapa |
| Tiiviste | Tiedoston tarkistuskoodaustapa |
| Template | Virtuaalikoneesta tehty kopio, josta voidaan tehdä uusia virtuaalikoneita. |
| DHCP | Dynamic Host Configuration Protocol |
| Playbook | Ansiblen käyttämä konfiguraatiokieli |
| SSH-avain | Autentikaatiomekanismi |
| HA | Korkea saatavuus |
| NTP | Network Time Protocol |
| UFW | Uncomplicated FireWall |
| UDP | User Datagram Protocol |
| MAC | Mandatory Access Control |
| IDS | Intrusion Detection System |
| SVN | Subversion |
| SSL | Secure Sockets Layer |
| Varmentajataho | Uskottu taho, joka myöntää sertifikaatteja salattujen yhteyksien muodostamiseen. |

# 1 JOHDANTO

Turvallisella palvelinympäristöllä tarkoitetaan ympäristöä, jonka palvelimet ovat mahdollisimman tietoturvalliset ja valvotut, ja jossa palvelimien päivitykset ovat ajan tasalla. On tärkeää, että palvelimien toiminnallisuuksia (nk. hyökkäyspinta-ala), joita voidaan käyttää palvelinta vastaan hyökätessä, minimoidaan ulkoisten ja sisäisten riskien takia, sillä pahimmassa tapauksessa yrityksen liiketoiminnan kannalta kriittistä materiaalia voi tuhoutua ja joutua vääriin käsiin. Kaikki nämä vaatimukset tulisi saavuttaa mahdollisimman automatisoidusti. Automatisoinnilla saavutetaan myös yhtenäisyys eri palvelimien konfiguraatioissa.

Solita Oy on Tampereelta lähtöisin oleva, itseään digitaaliseksi matkaoppaaksi kutsuva IT-talo, joka tarjoaa uutta liiketoimintaa ja palveluja verkossa yrityksille ja julkishallinnolle. Tällä hetkellä toimistoja on Tampereen lisäksi Helsingissä ja Oulussa. Yrityksessä työskentelee noin 330 ammattilaista eri tehtävissä.

Yrityksen sisäisestä IT-infrastruktuurista vastaa tällä hetkellä yrityksen sisäinen IT-osasto. Yksi monista IT-osaston tehtävistä on ylläpitää palvelimia, joilla pyörii yrityksen sisäisiä ja ulkoisia järjestelmiä sekä palveluita. Palvelujen ja palvelinmäärien kasvaessa voi palvelinympäristön ylläpito jäädä helposti toissijaiseksi, sillä palvelimet voivat toimia vuosia ilman kunnollista ylläpitoa, mutta turvallista ja järkevää se ei ole.

Opinnäytetyön tarkoituksena on luoda Solitalle ehdotus, kuinka sen tämän hetkinen palvelinympäristö sisältäen sekä nykyiset että tulevat palvelimet, saadaan vastaamaan yrityksen tietoturvavaatimuksia. Työssä käydään läpi eri komponentteja, joilla turvallinen palvelinympäristö voidaan luoda, sekä asioita mitä siinä pitää ottaa huomioon.

Työ rajoittuu käyttöjärjestelmien kohdalla Linux-palvelimiin, joilla käyttöjärjestelmänä on joko Debian- tai Red Hat Enterprise-pohjainen Linux-jakelu.

# 2 TURVALLINEN PALVELINYMPÄRISTÖ

## 2.1 Alkutilanne

Tämän hetkisessä tilanteessa yrityksen virtuaaliset Linux-palvelimet on luotu VMwaren Ubuntu Server- tai CentOS-templatesta, johon etukäteen ei ole tehty muita muutoksia, kuin asennettu VMwaren työkalut.

Kun uusi palvelin luodaan, sille määritellään verkkoasetukset, asennetaan päivitykset, luodaan yhdet järjestelmänvalvojatunnukset ja tehdään siitä resurssitiketti yrityksen käyttämään tehtävienhallintaohjelmistoon, johon kirjoitetaan palvelimen yleiset tiedot kuten nimi, käyttäjätunnukset, Virtual Local Area Network Identifier (VID), IP-osoite, resurssit ja mahdolliset palomuuri- ja valvontasäännöt.

Palvelimen käyttöönoton jälkeen sen ylläpidosta sekä turvallisuudesta vastaa se henkilö, jolle palvelin on osoitettu. Koska palvelimet eivät sisällä valmiina mitään valvontaa mahdollistavia työkaluja tai asetuksia, johtaa se lähes poikkeuksetta siihen, että kenelläkään ei ole tietoa mitä palvelimilla tapahtuu, kuka niille pääsee tai onko tietoturvapäivityksiä asennettu.

Nykyinen palvelinverkkojen jako on toteutettu jakamalla palvelimet kahteen eri Virtual Local Area Network:iin (VLAN), toinen IT:n tarjoamille palveluille sekä sisäisille järjestelmille ja toinen projektien käyttöön. Tämän seurauksena kaikki samassa VLAN:ssa olevat palvelimet pystyvät muodostamaan yhteyden toisiinsa.

## 2.2 Vaatimukset

Toimeksiantona oli laatia ehdotus, kuinka yrityksen palvelinympäristö saadaan turvattua käyttäen opinnäytetyön tiedonetsintävaiheessa esiin tulleita malleja ja käytäntöjä. Ehdotuksessa tulee olla listattuna parannusehdotuksia koskien verkkotekniikkaa, palveluiden näkyvyyttä, sekä yksittäisen palvelimen konfiguraatiota, joka on pystyttävä monistamaan automatisoidusti jokaisen IT-osaston hallinnoiman palvelimen käyttöön.

### 2.2.1 Palvelinkohtaiset vaatimukset

Yksittäisen palvelimen kohdalla tullaan noudattamaan kolmea pääperiaatetta: minimoidaan riskit, turvataan palvelut huolellisesti sekä pidetään ne ajan tasalla päivitysten ja muiden muuttuvien osien kohdalla.

Paras tapa vähentää palvelimiin kohdentuvia riskejä on karsia turhat toiminnot ja palvelut pois päältä, jolloin pienennetään pinta-alaa, jota kautta mahdollinen hyökkäys voi tapahtua. Käytännössä se tarkoittaa, että palvelimelta löytyy vain ne palvelut, joita tarvitaan. Kaikki ylimääräiset palvelut pitäisi poistaa tai ottaa pois käytöstä.

Toisena periaatteena on palvelimien turvaus ulkopuolisilta riskeiltä. Se mahdollistetaan käyttämällä eri suojauskerroksia, sääntöjä ja prosesseja. Pelkän palomuurin käyttö yrityksen sisäverkon ja internetin välillä ei ole riittävä suojauskerros. Sen lisäksi yksittäiset palvelimet pitäisi suojata käyttäen palvelimen omaa palomuuria. Lisäksi käyttöön pitäisi ottaa jokin tunkeilijan havaitsemisjärjestelmä eli Intrusion Detection System (IDS), jonka avulla pyritään tunnistamaan palvelimiin kohdistuvat hyökkäysyritykset.

Viimeinen periaate liittyy valppauteen. Palvelimien käyttöjärjestelmät ja ohjelmistot täytyy pitää päivitettyinä ja palvelimia täytyy valvoa vikatilanteiden ja poikkeamien varalta. (Turnbull 2005, 25-26).

Kaikki tämä täytyy pystyä tekemään keskitetysti siten, että asetukset voidaan monistaa sekä uusille että vanhoille palvelimille. Palvelinkonfiguraation täytyy myös säilyä määritellyn linjan mukaisena palvelimen eliniän, eli vastoin asetettuja ehtoja tehdyt muutokset täytyy pystyä huomaamaan ja kumoamaan. Lisäksi palvelimilta täytyy kerätä lokitietoja ja statistiikkaa, sekä niitä pitää pystyä valvomaan vikatilanteiden varalta.

### 2.2.2 Verkkokohtaiset vaatimukset

Verkkokohtaiset vaatimukset ovat yksinkertaiset palvelinympäristöä koskien. IT-osaston ylläpitämät palvelimet tai palvelut täytyy eristää niin, että jokainen yksittäinen palvelin tai palvelu sijaitsee omassa verkossaan, johon ei ole pääsyä kuin siitä vastaavalla taholla. Näiden verkkojen välinen liikenne evätään oletuksena ja poikkeukset tehdään tapauskohtaisesti. Näissä verkoissa sijaitsevat palvelut tuodaan näkyviin käyttäen edustapalvelimia, joista kerrotaan myöhemmin lisää.

# 3 KONFIGURAATIO JA MUUT JÄRJESTELMÄT

## 3.1 Jira

Jira on Atlassianin luoma tehtävien- ja projektienhallintaohjelmisto. Atlassianin mukaan sitä käytetään tehtävien organisoinnissa, työn jakamisessa sekä projektiryhmän seurannassa tuhansissa projekteissa (Atlassian 2015).

IT-osasto käyttää Jiraa sisällönhallintaan ja siellä ylläpidetään tieto kaikista IT-osaston hallinnoimista palvelimista, lisensseistä, työasemista ja muista lukuisista asioista. Ehdotuksessa Jirasta haettua tietoa palvelimista ja käyttäjistä käytetään hyväksi palvelimien asetusten ja pääsyoikeuksien asettamisessa.

## 3.2 Versionhallinta

Versionhallinta on järjestelmä, joka seuraa ja tallentaa tiedostoon tehtyjä muutoksia ja tarjoaa mahdollisuuden tarkkailla jokaista eri versiota. Versionhallintaa käytetään suurimmaksi osaksi ohjelmistokoodin tallennukseen ja hallintaan, mutta aivan samalla tavalla sitä voidaan käyttää myös konfiguraation säilytykseen. (Nemeth 2010, 398).

Solitalla käytetään tällä hetkellä pääasiassa Mercurialia, Git:iä ja Subversionia (SVN) ja jokaisessa järjestelmässä on omat hyvät ja huonot puolensa. Sopivimman työkalun valintaa ohjaa lähinnä käyttäjän oma mieltymys.

Tällä hetkellä Solita on siirtymävaiheessa, jonka aikana kaikki sen vanhat versionhallintajärjestelmät siirretään Deveon alle. Deveo on koodinhallinta ja yhteiskäyttötyökalu, joka tarjoaa saman järjestelmän alla edellä mainitut versionhallintaohjelmat (Deveo 2015).

Kaikki ehdotuksessa syntynyt koodi ja konfiguraatio säilytetään Deveossa käyttäen Git:iä.

## 3.3 Ansible

Ansible on konfiguraationhallintajärjestelmä, jonka avulla voidaan edellä mainitun lisäksi automatisoida palvelininfrastruktuuria, provisioida koneita pilveen ja orkestroida palveluita. Palvelimien hallintaan ei tarvita mitään asennettavaa ohjelmistoa tai ylimääräistä turvallisuusmekanismia, joten sen käyttöönotto on helppoa jo olemassa olevaankin ympäristöön. Hallittuja palvelimia komennetaan käyttäen merkintäkieltä nimeltä Yet Another Markup Language (YAML), jonka Ansible muuttaa kohdepalvelimella sen ymmärtämiksi normaaleiksi komennoiksi. YAML:n syntaksi on yksinkertainen ja nopeasti omaksuttavissa.

Ansible toimii ottamalla valitulle palvelimelle yhteyden käyttäen oletuksena salattua etäkäyttöohjelmistoa nimeltä Secure Shell (SSH). Kirjautumisen jälkeen koneella suoritetaan valittu moduuli ja poistetaan se. Moduuleita voidaan säilyttää millä tahansa koneella eikä Ansible vaadi toimiakseen palvelimia, tietokantoja tai taustalla pyöriviä ohjelmistoja. (Ansible 2015).

Ehdotuksessa palvelimien saattaminen automaattisesti haluttuun tilaan on toteutettu käyttäen Ansiblea.

### 3.3.1 Asennus

Ansible voidaan asentaa Unix-pohjaiselle tietokoneelle, jolle on asennettu Python 2.6. Hallituilla palvelimilla täytyy olla asennettuna Python 2.4 tai uudempi. (Ansible 2015).

Esimerkkikoodissa 1 asennetaan Ansible käyttäen Aptitudea.

```
sudo apt-get install software-properties-common
sudo apt-add-repository ppa:ansible/ansible
sudo apt-get update
sudo apt-get install ansible
```

ESIMERKKIKOODI 1. Ansiblen asennus

### 3.3.2 Käyttöönotto

Toimiakseen Ansible tarvitsee Inventaario-tiedoston hallittavista palvelimista. Oletuksena tiedosto löytyy polusta /etc/ansible/hosts. Palvelimet listataan tiedostoon allekkain. Myös erillistä inventaario-tiedostoa voidaan käyttää ja syöttää se parametrina moduulia ajettaessa.

Esimerkkikoodissa 2 näytetään esimerkki inventaario-tiedostosta.

```
[internal]
intra.yritys.fi
deveo.yritys.fi
jira.yritys.fi
[project_345]
db1.yritys.fi
front.yritys.fi
```

ESIMERKKIKOODI 2. Inventaario-tiedosto

Inventaarion avulla palvelimille ja ryhmille voidaan määritellä muuttujia ja asettaa niitä eri ryhmiin. Tällä tavoin ajettavia moduuleita voidaan kohdistaa ryhmiin ja käyttää muuttujia määrittelemään eri arvoja, joita moduuli käyttää.

Esimerkkikoodissa 3 näytetään esimerkki inventaario-tiedostosta, johon on määritelty muuttujia yksittäisille palvelimille sekä ryhmille.

```
[Tampere]
intra.yritys.fi ansible_ssh_user=intra_admin
aikataulu.yritys.fi
[Tampere:vars]
ntp_server=tre.ntp.yritys.fi
ansible_ssh_port=222
[Helsinki]
app.yritys.fi role=dhcp
db.yritys.fi
[Helsinki:vars]
ntp_server=hki.ntp.yritys.fi
ansibe_ssh_port=333
```

ESIMERKKIKOODI 3. Inventaario-tiedosto muuttujilla

### 3.3.3 Playbook

Playbookit sisältävät ajettavien moduulien koodin ja niiden avulla kohdepalvelin saadaan tahdottuun tilaan. Playbook voi olla yksittäinen pieni tehtävä tai kaiken kattava kokonaisuus, jonka sisällä yksittäiset tehtävät on jaettu rooleihin. Pieni playbook voi sisältää esimerkiksi moduulin, jolla kohdekoneelle asennetaan yksittäinen sovellus tietyllä konfiguraatiotiedostolla. Suuri playbook voi taas sisältää yrityksen kaikki palvelimet ja niiden halutut asetukset rooleilla määriteltyinä.

Esimerkkikoodissa 4 asennetaan kohdepalvelimille Audit-ohjelma.

```
---
# määritellään palvelimet, joille paketti asennetaan
- hosts: tampere
# ajetaan komennot sudona, eli pääkäyttäjän oikeuksilla
  sudo: yes
# ajettavat tehtävät
  tasks:
# asennetaan Audit, jos kyseessä RedHat-pohjainen Linux-jakelu
  - name: Install Audit (CentOS)
    yum: name=audit state=present
    when: ansible_os_family == "RedHat"
# asennetaan Audit, jos kyseessä Debian-pohjainen Linux-jakelu
  - name: Install Audit (Ubuntu)
    apt: name=auditd state=present
    when: ansible_os_family == "Debian"
# varmistetaan, että palvelu on käynnissä
  - name: Enable audited service
    service: name=auditd state=started
# muokataan Auditin konfiguraatiotiedostoa siten, että kaikki logitie-
dostot säilytetään
  - name: Keep all audit logs
    lineinfile:
            dest=/etc/audit/auditd.conf
            state=present
            regexp="^max_log_file_action"
            line="max_log_file_action = keep_logs"
            insertafter=EOF
# kopioidaan valmiiksi määritellyt säännöstöt
  - name: Copy audit rules
```

```
    copy:
      src=audit.rules.{{ansible_os_family}}
      dest=/etc/audit/rules.d/audit.rules
```

ESIMERKKIKOODI 4. Audit-ohjelman asennus

Esimerkkikoodissa 5 playbook ajetaan.

```
ansible-playbook playbook.yml –k –s –K –u user
```

ESIMERKKIKOODI 5. Playbookin ajo

Komennon purku:

ansible-playbook = playbookien ajo-ohjelma
playbook.yml = Sisältää edellisellä sivulla esitetyn koodin.
-k = Kysy SSH-salasanaa.
-s = Käytä sudoa, eli pääkäyttäjän oikeuksia.
-K = Kysy sudo-salasanaa.
-u = Käyttäjä, jona playbook ajetaan kohdekoneella.

Jotta koneiden hallinta toimisi ilman salasana-autentikaatiota ja käyttäjän interaktiota, kohdepalvelimelta tulee löytyä sen käyttäjän julkinen avain, jona playbook ajetaan. Julkisista avaimista ja niiden toiminnasta kerrotaan myöhemmin lisää.

Kuviosta 1 voidaan nähdä mitä kohdekoneelle tapahtuu:

- GATHERING FACTS: Kerätään tiedot kohdekoneesta.
- Installing Audit (CentOS): Kohta ohitetaan, koska kyseessä ei ole CentOS.
- Installing Audit (Ubuntu): Asennetaan audit.
- Enable auditd service: Käynnistetään palvelu, palautti OK, eli palvelu oli jo käynnissä.
- Keep all audit logs: Muutetaan konfiguraatiotiedostoa niin, että kaikki lokitiedostot säilytetään.
- Copy audit rules: Kopioi säännöt, joiden perusteella palvelimen toimintaa valvotaan.

```
user@ansible:~/opari/temp$ ansible-playbook main.yml -k -s -K -u user
SSH password:
sudo password [defaults to SSH password]:

PLAY [ubuntu1] *****************************************************************

GATHERING FACTS ****************************************************************
ok: [ubuntu1]

TASK: [Install Audit (CentOS)] *************************************************
skipping: [ubuntu1]

TASK: [Install Audit (Ubuntu)] *************************************************
changed: [ubuntu1]

TASK: [Enable audited service] *************************************************
ok: [ubuntu1]

TASK: [Keep all audit logs] ****************************************************
changed: [ubuntu1]

TASK: [Copy audit rules] *******************************************************
changed: [ubuntu1]

PLAY RECAP *********************************************************************
ubuntu1                    : ok=5    changed=3    unreachable=0    failed=0
```

KUVIO 1. Playbookin ajaminen

Ansible tarkastaa jokaisen playbookiin sisältyvän tehtävän kohdalla, tarvitaanko kohdepalvelimella toimenpiteitä, jotta se saadaan tehtävässä määriteltyyn tilaan. Mikäli palvelin on jo tahdotussa tilassa, ei muutoksia tarvitse tehdä ja tehtävä palauttaa arvon OK. Jos tehtävän suoritukseen on määritelty ehtoja eikä palvelin niitä täytä, palauttaa tehtävä arvon Skipping, eli kohta ohitetaan. Muutoksia vaatineet tehtävät palauttavat arvon changed. Tämän toiminnallisuuden ansiosta playbookeja voidaan ajaa ajastetusti vasten palvelimia ja varmistaa, että niiden asetukset pysyvät haluttuina. Ajastetuissa ajoissa täytyy käyttää SSH-avaimia, jotta kohdekoneelle yhdistämisessä ei tarvita ihmisen interaktiota.

# 4 TIETOVERKKO

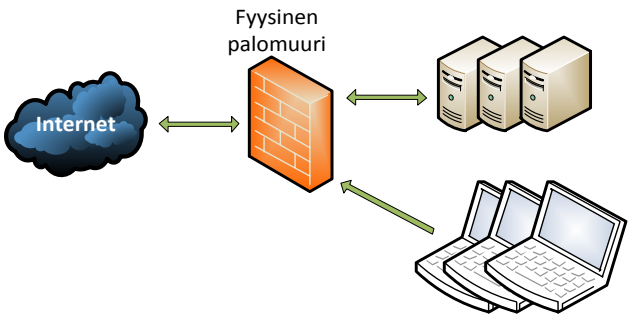
## 4.1 Segmentointi

Verkon segmentoinnilla tarkoitetaan yhden suuren verkon jakamista VLAN:eihin, eli virtuaalisiin lähiverkkoihin. Näin esimerkiksi jokaisella palvelulla tai projektilla voi olla oma suljettu verkkonsa yhden suuren verkon sisällä, jonne muista verkosta ei ole pääsyä. VLAN:it ovat loogisia verkkoja ja kaikki täsmä-, yleis- ja ryhmälähetyspaketit ovat ohjattu sekä lähetetty vain kyseisen verkon sisälle. Segmentoimalla verkko ja eristämällä jokainen IT-osaston tarjoama palvelu omaan VLAN:iin pienennetään mahdollisuutta tietomurtoihin ja sellaisen sattuessa estetään hyökkääjän pääsy muissa VLAN:eissa sijaitseviin yrityksen palveluihin tai palvelimiin. (Cisco 2014).

Ehdotuksessa kehotetaan jakamaan yrityksen verkko palvelukohtaisesti omiin VLAN:eihin. Oletuksena liikenne VLAN:ien välillä estetään ja pääsyoikeudet määritellään jokaisen verkon kohdalla vain tarvittaville henkilöille.

## 4.2 Palomuuri

Palomuuri on yksi tärkeimmistä puolustuskeinoista hyökkäyksiä vastaan ja se on yleensä ensimmäinen puolustus, johon hyökkäykset kohdistuvat. Palomuurin tehtävänä on tiputtaa tai hylätä ulospäin lähtevä tai sisään tuleva ei-toivottu liikenne. Sen avulla voidaan myös monitoroida ja kerätä epäilyttävä tai vahingollinen liikenne myöhempää tarkastelua varten. Yleensä ulommainen palomuuri sijaitsee yrityksen uloimman verkon reunalla ja sen tarkoituksena on suojata koko muuta verkkoa. Tämän lisäksi palvelimilta löytyy itsestään Netfilter, jonka tarkoituksena on suojella palvelinta itsessään. (Turnbull 2005, 79). Kuviossa 2 havainnollistetaan fyysisen palomuurin läpi kulkevaa liikennettä ja palomuurin sijaintia yrityksen sisäverkon ja internetin välissä.

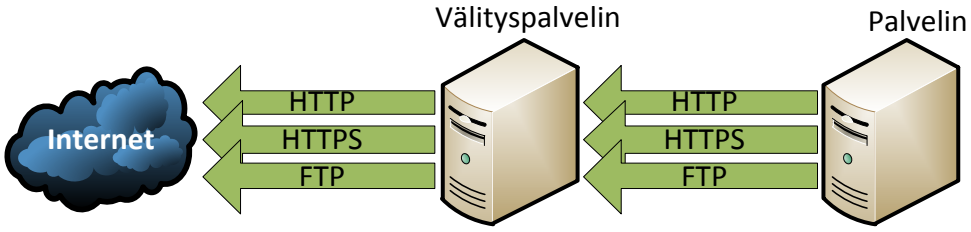



KUVIO 2. Palomuuri

Ehdotuksessa tämän hetkisiä palomuurisääntöjä kehotetaan muuttamaan tarkemmiksi nykyisestä mallista.

### 4.3 Välityspalvelin

Välityspalvelimen tarkoituksena on toimia palvelimen tai työaseman ja internetin välissä. Palvelin tekee pyynnön välityspalvelimelle, joka välittää pyynnön eteenpäin internetiin ja palauttaa tiedon takaisin palvelimelle. Välityspalvelinta käyttämällä palvelimia ei tarvitse paljastaa suoraan internetiin, jonka ansiosta tietoturva paranee. (Nemeth 2010, 974). Kuviossa 3 kuvataan kuinka verkkoliikenne kulkee palvelimelta välityspalvelimen kautta internetiin.

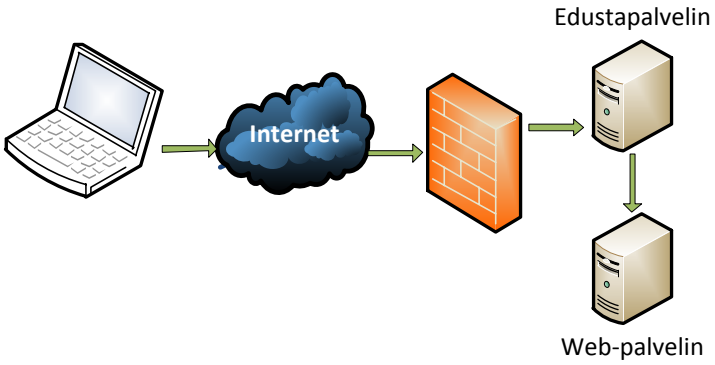



KUVIO 3. Välityspalvelin

### 4.4 Edustapalvelin

Edustapalvelin toimii päinvastoin kuin välityspalvelin eli se ohjaa ulkoapäin tulevan liikenteen toisille, yleensä eri verkossa sijaitseville palvelimille.

Edustapalvelinta käyttämällä voidaan suljetun verkon palveluita tuoda julki joko internetiin tai toiseen, yrityksen sisäiseen verkkoon avaamatta mahdollista tietoturva-aukkoa suljettuun palvelinverkkoon, missä palvelin sijaitsee. Näin ollen verkkoon on olemassa vain yksi potentiaalinen reikä ja palvelun valvonta sekä pääsynrajaaminen helpottuvat. (Nemeth 2010, 976). Kuviossa 4 havainnollistetaan kuinka verkkoliikenne ohjautuu edustapalvelimen kautta web-palvelimelle.




KUVIO 4. Edustapalvelin

Ehdotuksessa käytetään web-palvelinohjelmisto Nginx:ää, joka tarjoaa ominaisuuden toimia edustapalvelimena. Oletuksena kaikki sivut tarjoillaan salattuna HTTPS-rajapintaa käyttäen, jotta liikennettä ei voida tarkastella selkokielisenä mahdollisen hyökkääjän toimesta. Salauksen mahdollistamiseksi edustapalvelimille hankitaan Secure Sockets Layer-sertifikaatit (SSL) yrityksen käyttämältä tunnetulta varmentajataholta. SSL-sertifikaatin avulla edustapalvelimen ja siihen yhteyttä ottavan tietokoneen välille muodostetaan luotetun tahon varmistama salattu yhteys (Digicert).

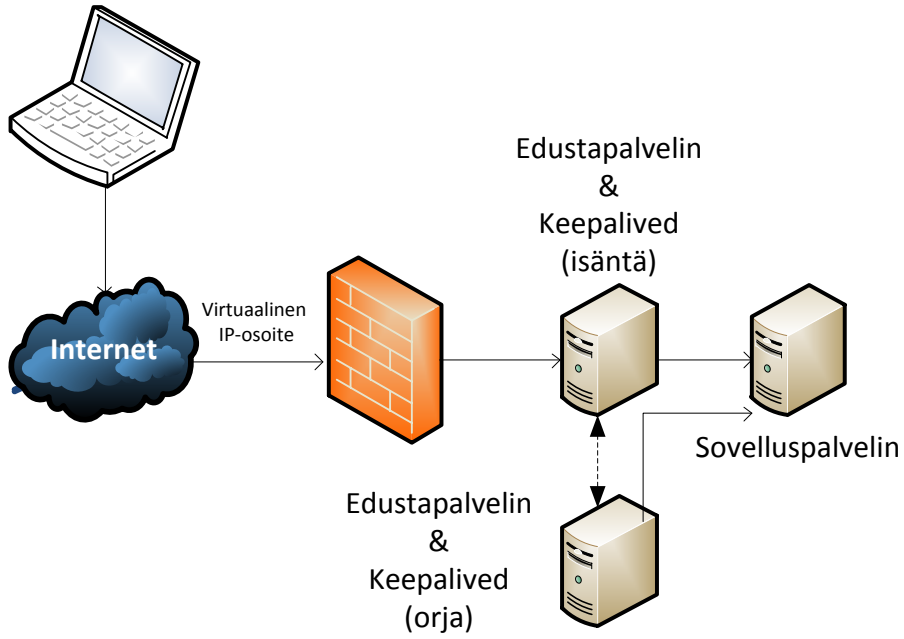
### 4.5 Korkea saatavuus eli High Availability (HA)

Tietotekniikassa korkealla saatavuudella tarkoitetaan järjestelmää, joka on suunniteltu ja hallittu siten, että sen avulla voidaan välttää palvelun oleminen alhaalla kokonaan tai mahdollisimman vähän. Korkea saatavuus ei kuitenkaan ole absoluuttinen määre, vaan se on jokaisen palvelun ylläpitäjän tai asiakkaan määriteltävissä. Yleensä palveluntarjoaja lupaa palveluntasosopimuksessa kuinka monta prosenttia palvelun olemassaoloajasta sen pitää olla saatavilla tai kuinka monta prosenttia se voi olla saavuttamattomissa. (Pearson Higher Ed).

Koska jokaisen IT-osaston tarjoaman palvelun kahdentaminen tai muutoin saattaminen vikasietoiseksi olisi toisen opinnäytetyön kokoinen kokonaisuus, ehdotuksessa päädyttiin käyttämään Keepalived nimistä reitityssovellusta edellisessä osiossa mainitun välityspalvelimien kahdentamiseen. Tällöin toinen kahdennetuista välityspalvelimista voidaan ajaa alas päivitysten ajaksi tai vikatilanteen sattuessa toinen palvelin voi ottaa roolin vastuulleen ja sen läpi tuodut palvelut jatkavat toimimistaan.

Keepalivedin toiminta perustuu instanssien väliseen verkon saatavuuteen, mutta sille voidaan myös asettaa muitakin ehtoja, kuten että palvelimella on käynnissä web-palvelimen käyttämä prosessi. Keepalived-ryhmälle määritellään virtuaalinen IP-osoite, josta palvelu vastaa. Ryhmään kuuluville instansseille määritellään joko isäntä- tai orja-rooli, joka tarkoittaa, että isäntä-roolissa oleva instanssi omistaa virtuaalisen IP-osoitteen ja vikatilanteen sattuessa sen ottaa käyttöönsä jokin määritellyistä orjista. Mikäli Keepalived-instansseja on useampia, riippuu orjan valinta sille määritellystä painoarvosta. Kun vikatilanteesta toivutaan, luovutetaan IP-osoite isäntäroolin haltijalle. Keepalived:iin on suotavaa määritellä sähköpostiasetukset, jotta vikatilanteiden sattuessa ylläpito saa niistä tiedon. (Red Hat 2015a). Kuviossa 5 havainnollistetaan Keepalived-instanssin toimintaperiaatetta.

Ehdotuksessa jokainen edustapalvelin, jonka läpi tuodaan jokin palvelu esiin, kahdennetaan ja otetaan käyttöön Keepalived. Tällöin molemmille edustapalvelimella täytyy olla ajossa täysin identtinen konfiguraatio, jotta vikatilanteessa palvelut näkyvät varmasti oikein. Tämä on toteutettu ajastetulla Ansiblen moduulilla joka tarkistaa ja korjaa mahdollisesti muuttuneen konfiguraation jokaisen edustapalvelimen osalta.




KUVIO 5. Keepalived

# 5 PALVELINKOHTAISET ASETUKSET

## 5.1 Levykuva

Koska palvelimet ovat suurimmaksi osaksi virtuaalipalvelimia, ehdotuksessa uudet instanssit tullaan asentamaan valmiista VMwaren virtuaaliympäristössä sijaitsevasta templatesta. Template on virtuaalikone, johon on asennettu pelkkä käyttöjärjestelmä ja saatavilla olevat päivitykset templaten tekopäivään saakka. Tämän jälkeen se on muutettu VMwaren templateksi, josta uudet instanssit pystytetään.

Templateja tehdään yksi kappale kummastakin käytetystä käyttöjärjestelmästä eli Ubuntu Serverin ja CentOS:in uusimmasta versiosta. Ubuntu asennetaan niin, että se ei sisällä mitään muuta ylimääräistä pakettia kuin OpenSSH-palvelimen, jotta sille voidaan ottaa yhteys käyttäen SSH-protokollaa. Myös CentOS:n template asennetaan käyttäen minimal-asennusta, jolloin se sisältää mahdollisimman vähän ylimääräisiä ohjelmia. CentOS:n minimal-asennus sisältää kuitenkin Postfixin MTA:n, Avahin ja Chronyn, jotka poistetaan asennuksesta ennen templaten tekoa.

Näillä ehdotuksilla templatet ovat mahdollisimman minimaalisia ja sisältävät jo valmiiksi mahdollisimman pienen hyökkäyspinta-alan. Kaikki tarvittavat asetukset ja ohjelmat asennetaan jälkikäteen Ansiblen-rooleilla.

## 5.2 Alustava haltuunotto

Alustavassa haltuunotossa uusi palvelin saatetaan tilaan, jossa sille ei ole pääsyä muilla kuin järjestelmänvalvojilla ja jossa sen päivitykset ovat ajan tasalla. Tämä sisältää seuraavat vaiheet:

- Lisätään koneelle uusi järjestelmänvalvojaryhmä.
- Lisätään edellä mainittuun ryhmään järjestelmänvalvojat.
- Lisätään palvelimen authorized_keys tiedostoon järjestelmänvalvojien julkiset avaimet.
- Poistetaan käytöstä autentikaatio käyttäen salasanaa, vain SSH-avaimilla tapahtuva autentikaatio sallitaan.
- Asennetaan uusimmat päivitykset ja käynnistetään palvelin uudestaan.

Tämän jälkeen palvelin on valmis ottamaan vastaan muita sille kohdistettuja rooleja.

### 5.3 Network Time Protocol (NTP)

Network Time Protocol eli NTP on protokolla, jota käytetään tietokoneiden kellojen synkronointiin. On tärkeää, että jokaiset toisiinsa yhteyksissä olevat palvelimet ovat samassa ajassa, sillä muutoin aikaleimat eivät täsmää, mikä aiheuttaa epätarkkuutta ja jopa toiminnan lakkautumisen. Erityisesti tietokantojen kanssa aikaleimat ovat erittäin tärkeitä. (NTP).

Ehdotuksessa jokaiselle palvelimelle asennetaan ntp-sovellus ja otetaan käyttöön yhtenäinen konfiguraatio, jonka mukaan palvelimen aika haetaan yrityksen sisäisiltä NTP-palvelimilta.

### 5.4 AIDE

AIDE on työkalu, jolla palvelimen tiedostojen eheyttä voidaan valvoa. Sen avulla ei pystytä estämään tiedostojen muuttumista, vaan muutokset kirjataan AIDE:n omaan tietokantaan ja niistä voidaan saada myös hälytys.

AIDE luo tietokannan sen konfiguraatiotiedostoon määritellyistä tiedostoista. Tietokantaan tallennetaan tiedoston pääsyoikeudet, inode numero, käyttäjä, ryhmä, tiedoston koko, tiedoston muokkausaika, tiedoston luontiaika, aika jolloin tiedostoa on käytetty sekä mahdolliset symboliset linkit ja niiden nimet. Tiedostot, joita on syytä valvoa, sisältävät tärkeät järjestelmän ohjelmat, kirjastot ja muut tiedostot, joiden ei oleteta muuttuvan palvelimen käytön aikana. Tiedostot, jotka muuttuvat usein, kuten lokitiedostot, väliaikaistiedostot ja käyttäjien kotihakemistot on syytä jättää ulkopuolelle.

Jokaisesta määritellystä tiedostosta generoidaan tarkistussumma tai tiiviste tietokantaan, jota hyödyntämällä mahdolliset muutokset tietomurron yhteydessä pystytään havaitsemaan.

Tietomurron sattuessa AIDE:n tietokantaa voidaan verrata aikaisempaan versioon ja siitä havaita, onko tiedostoja muokattu tai luettu hyökkääjän toimesta. Hyökkääjän on helppo peittää jälkensä ja saastuttaa palvelin omilla versioilla yleisimmistä työkaluista, mutta AIDE:n käyttämiä tarkistussummia tai tiivisteitä on lähes mahdotonta muokata huomaamattomasti. (Lehti, R.).

Saavuttaakseen täydellisen varmuuden tiedostojen eheydestä, pitäisi AIDE asentaa ja luoda ensimmäinen tietokanta ennen kuin palvelin on yhteydessä internetiin ja tallentaa tietokannasta kopio talteen muualle (nixCraft 2009).

Ehdotuksessa AIDE asennetaan jokaiselle uudelle koneelle Ansible-roolilla niin, että muutosten sattuessa niistä ilmoitetaan sähköpostilla.

### 5.5 Välityspalvelin

Jokaiselle palvelimelle, joka ei saa olla suoraan yhteydessä internetiin, otetaan käyttöön aiemmassa kappaleessa mainittu välityspalvelin, jonka läpi tarvittava liikenne internetiin kulkee. On huomioitavaa, että sisäverkon liikennettä ei kuljeteta välityspalvelimen kautta. Ehdotuksessa välityspalvelin otetaan käyttöön kohdepalvelimilla Ansible-roolilla.

### 5.6 Käyttäjätunnukset

Avaimiin perustuva SSH-kirjautuminen on yksi turvallisimmista autentikointitavoista, koska avaimia on lähes mahdotonta murtaa arvaamalla tai käyttämällä brute-force-hyökkäystä, sillä ne ovat erittäin pitkiä, eivätkä ne muodostu käyttäjän keksimistä sanoista (Morais 2015).

Brute-force-hyökkäyksellä tarkoitetaan hyökkäystaktiikkaa, jossa käyttäjätunnus ja salasana yritetään murtaa arvaamalla ne ohjelmallisesti. Ohjelma yrittää generoida käyttäjätunnuksen ja salasanan kokeilemalla jokaista mahdollista kirjainyhdistelmää. Tämä vaatii kuitenkin paljon aikaa ja resursseja ja on nykypäivänä tehoton hyökkäysmenetelmä olettaen, että salasana on tarpeeksi vahva. (Turnbull, 2005). Esimerkiksi kahdek-

sanmerkkisen salasanan, joka sisältää isoja ja pienimä kirjaimia sekä numeroita, murtamiseen kestäisi nykyaikaisella Intel i7-2600K prosessorilla yli 26 vuotta (Open Security Research).

Avaimiin perustuva autentikointi perustuu avainpariin, joka sisältää julkisen ja salaisen avaimen. Avainpari on turvallinen niin kauan, kun salainen avain pysyy vain käyttäjän tiedossa. Kohdepalvelimelle asennetaan tunnettuihin avaimiin käyttäjän avainparin julkinen avain. Yhteyttä muodostaessa palvelin salaa liikenteen käyttämällä käyttäjän julkista avainta niin, että se voidaan purkaa vain käyttäen käyttäjän salaista avainta. (IETF 2006).

Nykytilanteessa palvelimelle on määritelty ylläpidon toimesta vain yksi pääkäyttäjä ja Jiran palvelintiketiltä löytyvä salasana. Salasanan pysyessä lähes poikkeuksetta muuttumattomana, aiheutetaan tietoturvariski sen levitessä yhä useammille ihmisille.

Jotta pääsynhallintaa voidaan tiukentaa ja rajoittaa vain sen hetkisille palvelimen käyttäjille, tulee palvelimilla käyttää pääsynhallintaan julkisia avaimia. Salasanalla kirjautuminen otetaan kokonaan pois käytöstä. Jokaisen käyttäjän täytyy siis luoda itselleen avainpari, jolla autentikointi onnistuu.

Ehdotuksessa avainautentikaato toteutetaan luomalla Jiraan uusi projekti nimeltään KEYS, jonne käyttäjät voivat lisätä oman avainparinsa julkisen avaimen. Jiran palvelintiketiltä löytyy ”visibility”-kenttä, johon tällä hetkellä määritellään palvelintiketin näkevät henkilöt. Jiran tietoja hyödyntäen Ansiblella ajetaan roolia, joka pitää ajan tasalla palvelimen käyttöoikeudet niin, että palvelimelta löytyy tunnukset, jotka vastaavat Jira-tiketin visibility-kenttää ja että jokaisen käyttäjän julkinen avain lisätään palvelimelle, mikäli se Jirasta löytyy. Vastaavasti kun käyttäjä poistetaan palvelintiketiltä Jirasta, poistetaan julkinen avain palvelimen sallituista avaimista.

### 5.7 Päivitykset

Vanhentuneet ohjelmisto- ja käyttöjärjestelmäversiot ovat yksi suurimmista syistä miksi hyökkääjä pääsee murtautumaan palvelimelle. Suurin osa järjestelmistä pyörii vanhoilla versioilla, jotka ovat monta versionumeroa uusimmista jäljessä ja joissa on monesti

haavoittuvuuksia, joita hyökkääjä voi käyttää hyväkseen. On erittäin tärkeää, että palvelimet pidetään ajan tasalla niin käyttöjärjestelmän, kuin ohjelmistojen osalta. (Turnbull 2005, 56).

Ehdotuksessa päivitykset asennetaan palvelimelle määrittelemällä palvelimen Jira-tiketille viikon päivä ja kellonaika, jolloin päivitykset voidaan asentaa. Näiden muuttujien perusteella Ansible asentaa kohdepalvelimille komentosarjarivin, joka käynnistää päivitykset määriteltynä ajanhetkenä.

### 5.8 Iptables ja Uncomplicated Firewall (UFW)

Netfilter palomuuri tulee Linuxin ytimen mukana ja se mahdollistaa paketinsuodatuksen ja -muokkauksen ytimen tasolla, jolloin sitä ei kosketa niin useat rajoitukset kuin normaaleja sovelluksia. Netfilter on tilallinen palomuuri, mikä tarkoittaa, että jokaisen sen läpi kulkevan yhteyden tila pysyy tiedossa, mahdollistaen tarkemman kontrollin sen läpi kulkevasta liikenteestä.

Iptablesin toiminta perustuu tauluihin, jotka sisältävät ketjuja ja joiden sisältä yksittäiset säännöt löytyvät. Liikennettä, joka palvelimen läpi kulkee, verrataan näihin sääntöihin ja jos liikenne osuu johonkin näistä säännöistä, tehdään siihen perustuvat toimenpiteet, kuten sallitaan, tiputetaan tai estetään liikenne. (Turnbull 2005, 81).

Ubuntussa Iptablesia konfiguroidaan sen oletustyökalulla Uncomplicated Firewall (UFW). UFW on kehitetty helpottamaan Iptablesin konfigurointia käyttäjäystävällisellä tavalla (Entous 2015). CentOS:issa palomuuri konfiguroidaan suoraan Iptablesilla.

### 5.9 Lokitiedostot

Lokien kerääminen on olennainen osa turvallista palvelinympäristöä. Lähes jokainen palvelu tai sovellus luo lokimerkintöjä tapahtumistaan. Niiden avulla pystytään etsimään vikoja, tutkimaan turvallisuustapahtumia, harjoittamaan valvontaa tai seuraamaan trendejä. (Ali 2015, 225).

### 5.9.1 Logstash

Lokien keräämiseen on tarjolla monia vaihtoehtoja, mutta koska yrityksellä on jo käytössä Logstash, joka on todettu yrityksessä hyväksi työkaluksi, on se valittu ehdotuksessa lokitiedostojen loppusijoituspaikaksi.

Logstash on vapaaseen lähdekoodiin perustuva ilmainen työkalu lokien ja tapahtumien hallintaan. Sen avulla pystytään keräämään tekstuaalista tietoa mistä tahansa lähteestä ja se voidaan muokata halutulla tavalla tekstuaaliseen tai graafiseen muotoon.

Lokitiedostot täytyy kuitenkin lähettää jollain tavalla Logstashiin ja koska ohjelmistoa nimeltä Rocket-fast system for log processing (rsyslog) on käytetty yrityksessä aiemmin ja se on todettu toimivaksi, tullaan sitä vielä jatkossakin käyttämään.

Rsyslogin heikkoutena on kuitenkin sen käyttämä salaamaton protokolla User Datagram Protocol (UDP). Protokolla ei tarkista pääsevätkö paketit perille, joten tietoa voi hävitä matkalle tietomäärien kasvaessa ja salauksen puuttuessa tietoa voidaan kaapata selkokielisenä. (Turnbull 2005, 234–235).

### 5.9.2 Process accounting

Process accounting on työkalu, jolla voidaan seurata mitä komentoja palvelimella on ajettu, kuka niitä on ajanut ja miten eri prosessit käyttävät palvelimen resursseja. Valmiudet kerätä tätä tietoa löytyy lähes kaikkien modernien Linux-jakeluiden ytimestä, mutta tiedon hyödyntämiseen tarvitaan jokin kolmannen osapuolen sovellus. (Turnbull 2005, 44-45).

Komentoja:

- ac = Näyttää montako tuntia komennon ajanut käyttäjä on ollut kirjautuneena sisään.
    - optiot:
        - – d Ajat jaoteltuina päiville.
        - – p Kaikkien käyttäjien ajat.

- sa = Näyttää käyttäjän ajamat komennot sekä kertoo niiden käynnissäoloajan ja suorittimen käytön.
    - optiot:
        - -u Yksittäinen käyttäjä.
        - -m Prosessien määrä per käyttäjä.
- lastcomm = Näyttää käyttäjien ajamat komennot.
    - optiot:
        - käyttäjä = Näyttää vain määritellyn käyttäjän ajamat komennot.

Ehdotuksessa Process accounting asennetaan jokaiselle palvelimelle, jotta tämä tieto saadaan tarvittaessa näkyviin.

### 5.9.3 Audit

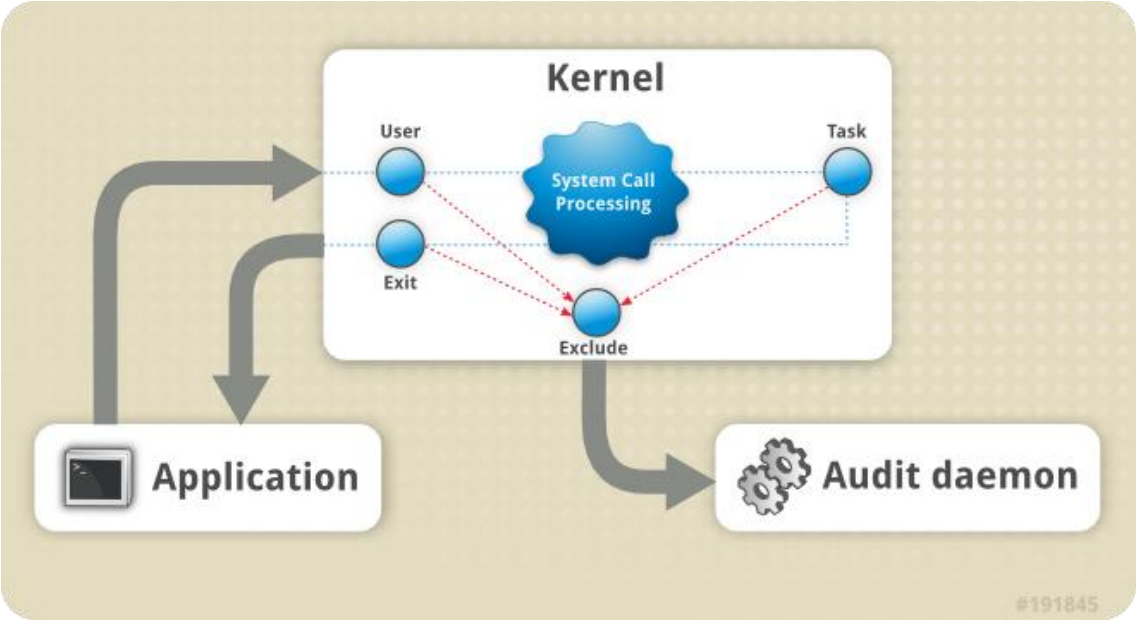
Audit on ohjelma, joka seuraa ja tallentaa sille määriteltyjä tapahtumia palvelimella. Ohjelma ei itsessään tarjoa minkäänlaista turvallisuusmekanismia, vaan se on lähinnä työkalu tapahtumienvalvontaan. Tarpeen tullessa näitä tietoja voidaan hyödyntää selvitettäessä mitä palvelimella on tehty ja kenen toimesta. Kuviossa 6 havainnollistetaan Auditin toimintaperiaatetta.

Auditilla voidaan tallentaa seuraavia tapahtumia:

- tapahtuman aika, tyyppi ja lopputulos
- tapahtuman laukaisija (käyttäjä tai prosessi)
- Auditiin kohdistuvat muutokset
- autentikaatiomekanismien käyttö
- tietokantojen muutokset

Oletuksena Audit valvoo SELinuxin tapahtumia, kirjautumisia järjestelmään, käyttäjätilien muutoksia sekä autentikointitapahtumia. (Red Hat 2015b).

Ehdotuksessa Audit asennetaan automaattisesti kaikille palvelimille muokatun konfiguraatiotiedoston kera ja lokitiedostot lähetetään yrityksen Logstash-palvelimelle.




KUVIO 6. Auditin toimintaperiaate (Red Hat 2015c)

### 5.10 Mandatory Access Control (MAC)

Pakollinen pääsynhallinta eli MAC on turvallisuusstrategia, joka rajaa yksittäisen resurssin omistajan pääsyn muihin resursseihin tiedostojärjestelmässä. Palvelimen ylläpitäjä luo säännöstöt, joita käyttöjärjestelmä valvoo. (Rouse 2013).

Pakollisen pääsynhallinnan avulla käyttäjille annetaan eri turvallisuustasoja struktuuroidusta hierarkiasta. Tällä tarkoitetaan, että käyttäjä pystyy lukemaan ja kirjoittamaan tiedostoja saman ja alemman turvallisuustason luokituksesta, mutta ei ylemmän.

Hyvin suunniteltu pakotettu pääsynhallinta on toteutettu antamalla käyttäjälle pienimmät mahdolliset oikeudet, joiden avulla tarvittavat toiminnot voidaan suorittaa. (Nemeth 2010, 922).

Eri Linux-jakeluissa on käytössä erityyppisiä pakollisia pääsynhallintamoduuleita, joista kahdesta, tämän työn aihealueeseen osuvista moduuleista kerrotaan seuraavassa kahdessa kappaleessa.

### 5.10.1 SELinux

SELinux on Linuxin ytimeen kehitetty turvallisuusmoduuli, joka mahdollistaa pakotetun pääsynhallinnan Linux-palvelimille. Sen kehittynein versio löytyy Red Hat-jakelusta ja sen ilmaisesta versiosta CentOS:sta. (Nemeth, 2010, 923)

Uusien säännöstöjen tekeminen käsin on hyvin monimutkaista, sillä siinä pitää määritellä kaikki tiedostot, kansiot ja muut kohteet, joita prosessi käyttää. Näitä kuitenkaan tarvitsee enää harvoin luoda itse, sillä yleisimmille sovelluksille löytyy internetistä valmiit säännöstöt ja paketti sisältää myös itsessään kelvolliset oletussäännöstöt. (Nemeth, 2010, 923)

Ehdotuksessa SELinux kytketään päälle jokaiselle Red Hat-pohjaiselle palvelimelle ja valvotaan, että se pysyy päällä.

### 5.10.2 AppArmor

Kuten SELinux, AppArmor on Linuxin ytimeen kehitetty turvallisuusmoduuli ja se on ollut saataville Ubuntulle versiosta 7.04 lähtien. Versiosta 7.10 eteenpäin se on myös oletuksena päällä. Paketti sisältää itsessään useimmille sovelluksille valmiit profiilit, mutta sellaisen puuttuessa niitä voi myös tehdä itse. AppArmor eroaa monista muista pakollisista pääsynhallintajärjestelmistä siinä, että se perustuu polkuihin ja se mahdollistaa pakotettujen- sekä pelkkien valvontaprofiilien käytön yhtä aikaa. (Deslauriers 2015).

Ehdotuksessa AppArmor kytketään päälle jokaiselle Debian-pohjaiselle palvelimelle Ansible-roolilla ja valvotaan, että se pysyy päällä.

## 5.11 Palvelut

Huolehtimalla että palvelimen päivitykset ovat ajan tasalla, pienennetään palvelimen hyökkäyspinta-alaa tunnettujen tietoturva-aukkojen varalta. Paras tapa suojautua tuntemattomia haavoittuvuuksia vastaan on kuitenkin poistaa tai sammuttaa palvelut, joita ei

tarvita järjestelmän toimintaan, sillä silloin niiden palveluiden mahdollisia tietoturva-aukkoja ei voida hyväksikäyttää. (Center For Internet Security, 2015).

### 5.11.1 Yleiset

Seuraavat ohjelmat on suositeltavaa poistaa tai sammuttaa käytöstä, ellei niiden käytössä olemiselle ole erityistä syytä:

- **Network Information Service (nis)**
    - Hakemistopalveluprotokolla, jota käytetään järjestelmän konfiguraatiotiedostojen levitykseen. Haavoittuvainen DOS- ja muistinylivuotohyökkäyksille, sekä käyttää heikkoja autentikointimekanismeja.
- **RSH (rsh-client, rsh-redone-client)**
    - Vanhentunut etäyhteysohjelmisto, jota käytettiin ennen SSH:ta. Sisältää useita haavoittuvuuksia. Poistettaessa otettava huomioon, että molemmat palvelin- ja asiakasohjelmisto poistetaan.
- **Talk (talk, talk-server)**
    - Ohjelma mahdollistaa käyttäjien lähettää ja vastaanottaa viestejä palvelimien välillä käyttäen terminaalisessiota. Liikenne on salaamatonta, joten se aiheuttaa tietoturvariskin.
- **Telnet-palvelin (telnetd)**
    - Ohjelma mahdollistaa yhteyksien ottamiseen palvelimeen telnet-protokollalla, joka ei ole turvallinen, eikä sen liikenne ole salattua. Voi johtaa salaamattoman liikenteen urkintaan.
- **Time (time)**
    - Palvelu joka palauttaa palvelimen kellonajan ja päivämäärän. Palvelu on tarkoitettu vain vianetsintään ja testaukseen.
- **X-ikkunoidenhallinta (server-xorg-core*)**
    - Ohjelma tarjoaa graafisen käyttöliittymän palvelimelle.
- **Avahi-daemon (avahi-daemon)**
    - Ohjelma tarjoaa verkkopalveluiden automaattisen löytämisen. Se löytää verkosta automaattisesti tulostimet, talk-palvelimet sekä muut verkkopalvelut.
- **Common Unix Print System (cups)**
    - Ohjelma tarjoaa mahdollisuuden palvelimelle tulostaa käyttäen paikallista tai verkkotulostinta. Voidaan myös sallia muiden koneiden tulostaminen käyttäen palvelimen paikallista tulostinta. Jos palvelin ei toimi erikseen tulostinpalvelimena, on suositeltavaa poistaa ohjelma.

(Center For Internet Security, 2015)

### 5.11.2 Erikoistapaukset

Seuraavat paketit on syytä poistaa, ellei palvelinta ole erikseen tehty paketin tarjoamaan rooliin:

- **DHCP-palvelin (isc-dhcp-server, isc-dhcp-server6)**
  - Palvelu jakaa muille koneille dynaamisia IP-osoitteita.
- **LDAP-palvelin/isäntä (slapd)**
  - Palvelu tarjoaa toimimisen LDAP-palvelimena tai asiakasohjelmana.
- **NFS-palvelin (nfs-kernel-server)**
  - Palvelu mahdollistaa NFS-jakojen tekemisen palvelimelle.
- **Nimipalvelin (bind9)**
  - Palvelu mahdollistaa palvelimen toimimisen nimipalvelimena, joka yhdistää IP-osoitteen nimitietueisiin.
- **FTP-palvelin (vsftpd)**
  - Palvelu mahdollistaa palvelimen toimimisen FTP-palvelimena, jolloin siihen voidaan yhdistää käyttäen FTP-protokollaa.
- **HTTP-palvelin (apache2,httpd,nginx)**
  - Palvelut mahdollistavat palvelimen toimimisen web-palvelimena, jolla tarjotaan web-sivuja.
- **Sähköpostipalvelin (dovecot)**
  - Palvelu mahdollistaa palvelimen toimimisen sähköpostipalvelimena.
- **Samba (samba)**
  - Palvelu mahdollistaa samba-jakojen teon palvelimelle, joka mahdollistaa palvelimen tiedostojen jakamisen Windows-koneille.
- **Välityspalvelin (squid3, squid)**
  - Palvelu mahdollistaa palvelimen toimimisen välityspalvelimena.

(Center For Internet Security, 2015)

# 6 VALVONTA

Valvonta on yksi tärkeimmistä palvelininfrastruktuurin hallintaelementeistä. Sen avulla ylläpito saa tiedon, kun palvelussa, palvelimella tai verkossa on ongelmia. On suositeltavaa, että valvontaan käytetty työkalu mahdollistaa ilmoittamisen potentiaalisista ongelmista jo ennen niiden syntyä. Keräämällä valvottavan kohteen historiatietoa voidaan tehdä trendianalyysiä, joka auttaa mahdollisten ongelmakohtien selvittämisessä ja kapasiteettisuunnittelussa. (Ali 2015, 167)

Ehdotuksessa yrityksen valvontaa kehotetaan parantamaan tekemällä vertailu asiakasohjelmaa käyttävistä valvontajärjestelmistä ja ottamalla niistä parhaiten sopiva käyttöön. Asiakasohjelmaan perustuvaa valvontajärjestelmää käyttämällä jokainen palvelin voidaan liittää Ansible-roolilla valvonnan piiriin automaattisesti eikä unohduksia tapahdu.

## 6.1 Valvottavat kohteet

Valvottavia kohteita mietittäessä on otettava huomioon kohteen tärkeys yrityksen toiminnalle sekä se, mitä resursseja kohteesta täytyy valvoa (Ali 2015, 167). Kuviossa 7 esitetään kuvakaappaus Check_MK-valvontajärjestelmän statistiikkasivusta joka kertoo kaikkien valvottavien palvelimien ja palveluiden sen hetkisen tilanteen.


KUVIO 7. Valvontajärjestelmän statistiikkaa (Peter, F. 2013)

## 6.2 Hälytykset

Hälytykset on syytä määritellä niin, että potentiaalisista ongelmista ilmoitetaan, ennen kuin koko palvelu tai palvelin menee toimintakyvyttömäksi. Esimerkiksi palvelimen resursseille kuten prosessorille, muistille ja levynkäytölle on syytä määritellä prosentuaaliset rajat, joiden ylittyessä saadaan hälytys. Näin ongelmiin pystytään puuttumaan, ennen kuin koko palvelu tai palvelin lakkaa toimimasta.

Yleisin tapa vastaanottaa hälytyksiä on sähköposti. Sähköpostin lisäksi tärkeille kohteille kannattaa ottaa käyttöön tekstiviestihälytykset, sillä tekstiviestillä vastuuhenkilön saa paremmin kiinni. Monet palvelut tarjoavat myös oman sovelluksen puhelimelle, johon hälytykset pusketaan.

## 6.3 Tarkastukset

Tarkastukset palvelimen tai palvelun tilasta voidaan jakaa kahteen eri kategoriaan: passiivisiin ja aktiivisiin. Aktiiviset tarkastukset tehdään valvontapalvelimen toimesta. Aktiivisten tarkastuksien hyötynä on helppo hallittavuus: kaikki tarkastukset sijaitsevat valvontapalvelimella, joten niiden päivittäminen ja hallinta on helppoa. Myös palomuurisääntöjen päivittäminen helpottuu, sillä palvelimelle täytyy luoda vain yksi sääntö, joka sallii valvontapalvelimen liikenteen.

Huonoja puolia aktiivisissa hälytyksissä ovat kuorman kehittyminen valvontapalvelimelle valvottavien kohteiden määrän kasvaessa sekä kohteiden valvominen, joihin ei voida sallia verkkoliikennettä.

Passiiviset tarkastukset toimivat toisinpäin, eli valvottava kohde lähettää tiedon valvottavasta kohteesta valvontapalvelimelle tietyin väliajoin. Passiivisten tarkastusten hyötyjä ovat skaalautuvuus sekä se, että valvontapalvelimelle ei synny niin paljon kuormaa.

Huonona puolena passiivissa tarkastuksissa on hallitsemattomuus. Tarkastuksia ei voida hallita yhtenäisesti valvontapalvelimelta. (Ali 2015, 169).

### 6.3.1 Palvelin

Yksittäisen palvelimen kohdalta on tärkeää valvoa prosessorin, muistin ja levyn käyttöä sekä verkon kuormaa. Kuormalle on määriteltävä raja-arvo, jonka ylittyessä siitä varoitetaan, jotta mahdolliseen ongelmaan voidaan puuttua ajoissa. Kerättyä historiatietoa voidaan käyttää myöhemmin hyväksi kapasiteettisuunnittelussa. (Ali 2015, 168).

### 6.3.2 Palvelu

Palvelun valvonta riippuu paljon palvelun tyypistä, mutta esimerkiksi web-palvelinta voidaan valvoa aktiivisesti ulkoa käsin niin, että valvontanoodi ottaa yhteyttä tarjottuun web-sivustoon ja tarkistaa, että se vastaa oikealla http-vastauskoodilla ja määritellyn ajan sisällä. Sisäisesti palvelua voidaan web-palvelimen osalta valvoa tarkistamalla pyöriikö web-palvelimen prosessi, paljonko web-palvelimella on yhteyksiä tai kuinka paljon web-palvelimen prosessi kuormittaa prosessoria. (Ali 2015, 168).

## 6.4 Valvontajärjestelmät

Valvontajärjestelmiä on nykypäivänä saatavilla runsaasti ja ne voidaan karkeasti jakaa kahteen ryhmään: Ensimmäiseen ryhmään kuuluvat järjestelmät, jotka ylläpito asentaa yrityksen omaan ympäristöön. Näistä suosittuja ovat esimerkiksi Nagios, Zabbix, Munin ja Cacti. Etuina itse ylläpidetyissä järjestelmissä ovat täysi hallittavuus ja se, että valvottavia kohteita ei tarvitse päästää julkiverkkoon. Huono puoli on, että järjestelmää täytyy ylläpitää ja valvoa ylläpidon toimesta.

Toiseen ryhmään kuuluvat valmiina ostetut palvelut, joiden ylläpito on palveluntarjoajan vastuulla. Etuina näissä palveluissa on oma ylläpidon puute ja varmatoimisuus itse järjestelmässä. Huonoina puolina voidaan nähdä, että järjestelmää ei välttämättä voida konfiguroida tarpeeksi omaan käyttöön sopivaksi sekä se, että aivan kaikkea ulkoverkosta käsin ei välttämättä voida monitoroida.

Yksi ratkaisu huonojen puolien eliminoimiseen on käyttää molempia järjestelmiä. Omalla järjestelmällä valvotaan sisäisiä palveluja ja palvelimia, joiden ei ole tarkoitus

näkyä ulkomaailmaan ja ulkoa ostetulla palvelulla valvotaan jo julkiverkossa näkyviä web-sivustoja ja palvelimia.

# 7 POHDINTA

Opinnäytetyön tarkoituksena oli rakentaa Solita Oy:lle ehdotus yrityksen IT-osaston palvelinympäristön saattamisesta tasolle, joka vastaa yrityksen tietoturvavaateita. Yrityksen suunnalta tulleet vaatimukset koskivat verkkotekniikkaa, palveluiden näkyvyyttä ja yksittäisten palvelimien asetuksia. Lisäksi vaatimuksena oli yksittäisten palvelimien kohdalla mahdollisimman automatisoitu siirtymä turvalliseen ympäristöön. Työn tuloksena syntynyt ehdotus sisältää toimenpiteitä kaikille vaatimuksille. Erityisesti yksittäisten palvelimien kohdalla vaatimukset toteutuivat hyvin ja lopputuloksena syntyi kattava Ansible-playbook, jolla IT-osaston palvelimet uusista vanhoihin saadaan tietoturvallisiksi automaation avulla.

Muita ehdotuksen osioita käsiteltiin pintapuolisemmin, eikä niistä ole laadittu tarkkoja räätälöityjä käyttöönottosuunnitelmia. Niiden avulla saa kuitenkin selkeän kuvan mitä ympäristölle voisi tehdä.

Tämän opinnäytetyön suurin haaste liittyi aihealueen rajaukseen. Aihealue on hyvin laaja eikä sitä ole mahdollisuutta käsitellä kokonaisuudessaan yhdessä opinnäytetyössä. Työssä keskitytäänkin tärkeimmiksi koettuihin komponentteihin, joilla yrityksen palvelinympäristö saadaan turvattua. Vaikka Ansible oli ennestään tuttu, haasteita aiheutti luotu Ansible-playbook, joka piti saada toimimaan saumattomasti sekä Red Hat-, että Debian-pohjaisilla Linux-jakeluilla.

Työssä käytettiin lähteinä alan kirjallisuutta ja internet-lähteitä. Lähteitä voidaan pitää luotettavina ja ajan tasalla olevina.

Jatkoa ajatellen, luotu ehdotus otetaan tarkempaan mietintään koko IT-osaston kesken ja suunnitellaan, miten ja miltä osin se voitaisiin toteuttaa käytännössä. Ennen varsinaista käyttöönottoa Ansiblella rakennetut konfiguraatiot tulee vielä testata ja koeajaa yrityksen ympäristössä, sillä vastoin alkuperäistä suunnitelmaa, testiympäristö rakennettiin omalle kotikoneelleni. Työn ulkopuolelle jäi joitain turvallisuusaspekteja, joiden soveltuvuutta voidaan tutkia ja ottaa käyttöön myöhemmin, mikäli niille koetaan tarvetta.

**LÄHTEET**

Atlassian. nd. Jira. Luettu 16.3.2015.
https://www.atlassian.com/software/jira

Ali, S. 2015. Practical Linux Infrastructure. California: Apress.

Ansible. nd. 2015. Installation. Luettu 19.3.2015.
http://docs.ansible.com/intro_installation.htm

Ansible. nd. How Ansible works. Luettu 17.3.2015.
http://www.ansible.com/how-ansible-works

Center For Internet Security. 2015. CIS Ubuntu 14.04 LTS Server Benchmark. Luettu 24.4.2015.
https://benchmarks.cisecurity.org/tools2/linux/CIS_Ubuntu_14.04_LTS_Server_Benchmark_v1.0.0.pdf

Cisco. 2014. Cisco Networking Academy's Introduction to VLANs. Luettu 24.4.2015.
http://www.ciscopress.com/articles/article.asp?p=2181837&seqNum=4

Deslauriers, M. 2015. AppArmor. Luettu 24.4.2015.
https://wiki.ubuntu.com/AppArmor

Digicert. What Is SSL (Secure Sockets Layer) and What Are SSL Certificates? Luettu 30.4.2015.
https://www.digicert.com/ssl.htm

Entous, D. 2015. UFW. Luettu 21.4.2015.
https://help.ubuntu.com/community/UFW

Hall, D. 2013. Ansible Configuration Management. Birmingham: Pact Publishing.

Lehti, R. nd. The AIDE manual. Luettu 19.4.2015.
http://aide.sourceforge.net/stable/manual.html

Logstash. nd. Logstash. Luettu 19.3.2015.
http://logstash.net/

Morais, J. 2015. Public and Private Keys. Luettu 20.4.2015.
https://help.ubuntu.com/community/SSH/OpenSSH/Keys

Nemeth, E. 2010. UNIX and Linux System Administration Handbook. New Jersey: Prentice Hall.

nixCraft. 2009. Debian / Ubuntu Linux Install Advanced Intrusion Detection Environment (AIDE) Software. Luettu 24.4.2015.
http://www.cyberciti.biz/faq/debian-ubuntu-linux-software-integrity-checking-with-aide/

Open Security Research. nd. Brute Force Calculator. Luettu 22.4.2015.
http://calc.opensecurityresearch.com/

Pearson Higher Ed. nd. Basic High Availability Concepts. Luettu 18.4.2015.
http://www.pearsonhighered.com/samplechapter/0130893552.pdf

Peter, F. 2013. Check_mk Custom Dashboard. Kuvakaappaus. Luettu 27.4.2015.
http://fabianpeter.de/monitoring/check_mk-custom-dashboard/

Red Hat. 2015a. Red Hat Enterprise Linux 7: Keepalived Overview. Luettu 20.4.2015.
https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/Load_Balancer_Administration/ch-keepalived-overview-VSA.html

Red Hat. 2015b. Red Hat Enterprise Linux 7: System Auditing. Luettu 24.4.2015.
https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/Security_Guide/chap-system_auditing.html

Red Hat. 2015c. Red Hat Enterprise Linux 7: System Auditing. Kuvakaappaus. Luettu 24.4.2015.
https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/Security_Guide/chap-system_auditing.html

Rouse, M. 2013. Mandatory access control (MAC). Luettu 21.4.2015.
http://searchsecurity.techtarget.com/definition/mandatory-access-control-MAC

The Internet Engineering Task Force (IETF). 2006. The Secure Shell (SSH) Authentication Protocol.
https://tools.ietf.org/html/rfc4252#section-4

The NTP Project. nd. What is NTP?. Luettu 24.2.2015.
http://www.ntp.org/ntpfaq/NTP-s-def.htm

Turnbull, J. 2005. Hardening Linux. California: Apress.