Carl Kvickström

Pilvipalvelut ja yksityisen pilvipalvelun toteutus

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Tietotekniikka

Insinöörityö

10.5.2015

Metropolia

Tekijä(t) Otsikko Sivumäärä Aika	Carl Kvickström Pilvipalvelut ja yksityisen pilvipalvelun toteutus 29 sivua + 1 liite 10.5.2015
Tutkinto	Insinööri (AMK)
Koulutusohjelma	Tietotekniikka
Suuntautumisvaihtoehto	Tietoverkot
Ohjaaja(t)	Lehtori Tapio Wikström

Pilvipalveluiden käyttö on yleistynyt huimasti viime vuosien aikana ja niitä sovelletaan koko ajan yhä monipuolisemmin eri käyttötarkoituksiin. Koska pilvipalveluita on tarjolla hyvin monipuolisesti ja niiden käyttö ei ensisilmäyksellä välttämättä eroa normaalisti tuotetuista palveluista, saattaa herätä kysymyksiä jotka koskevat palveluiden rakennetta ja toimintaa.

Tässä insinöörityössä perehdyttiin pilvipalveluiden rakenteisiin ja toimintaan sekä selvitettiin niistä saatavat hyödyt ja haitat. Työn tavoitteena oli myös selventää käyttäjän ja palveluntarjoajan välistä suhdetta sekä tietoturvan kasvanutta merkitystä pilvipalveluissa.

Työn lopputuloksena toteutettiin yksityiseen käyttöön tarkoitettu pilvipalvelualusta, johon asennettiin ownCloud-tallennuspalvelu ja Wordpress-julkaisualusta. Tämän käyttöönoton avulla saatiin työhön käytännön näkökulma, joka auttoi ymmärtämään paremmin ne asiat, joista palveluntarjoajan ja käyttäjän on hyvä olla tietoinen.

Pilvipalvelut olivat työn lopuksi käytettävissä pilvipalveluille tyypilliseen tapaan verkkoselaimella. Suojauksesta huolehti TLS-salausprotokolla, ja varmuuskopiointi automatisoitiin suoritettavaksi päivittäin muiden palveluiden taustalle.

Pilvipalveluiden käyttö on yleistymässä kovaa vauhtia haasteista huolimatta. Niiden tarjoamat edut ja edullinen hinnoittelumalli takaavat sen, että niitä tullaan tarjoamaan jatkossa yhä enemmän.

Avainsanat	Pilvipalvelut, tietoturva, käyttöönotto

Author(s) Title Number of Pages Date	Carl Kvickström Cloud Services and Implementation of Private Cloud 29 pages + 1 appendice 10 May 2015
Degree	Bachelor of Engineering
Degree Programme	Information Technology
Specialisation option	Data Networks
Instructor(s)	Tapio Wikström, Senior Lecturer

The use of cloud services has increased dramatically in the last few years and they are being applied more and more for different purposes. There is a huge variety of different cloud services on offer, whose use might not differ much from normally manufactured services. This can rise questions regarding the structure and operation of cloud services.

This thesis researched the structure and operation of cloud services and explained the advantages and disadvantages of this type of service model. The goal was also to clarify the connection between the service provider and the user without forgetting the importance of a good security.

The result of this thesis was a cloud service platform designed for private use, which was installed ownCloud storage service and WordPress publishing platform. With help of this deployment process were given a practical point of view, which helped to better understand the things the service provider and the user must be aware of.

At the end of this work the cloud services were accessible by a web-browser. The security was carried out using a TLS encryption protocol and the backup was automated to run daily in the background of other services.

The use of cloud services is increasing rapidly, despite the challenges. Their benefits and affordable pricing model will ensure that they will be offered in the future more and more as a solution to the increased need of services.

Keywords	Cloud services, security, deployment

Sisällys

Liitteet

Liite 1. Apache Name-based Virtual Hosts

Lyhenteet ja asiasanat

API	*Application Programmin Interface*. Ohjelmointirajapinta, jonka avulla ohjelmat voivat tehdä pyyntöjä ja vaihtaa tietoja keskenään.
ASP	*Application Service Provider*. Jostain hankittujen resurssien avulla tuotettu palvelu.
AWS	*Amazon Web Services*. Amazon.com tuottama ja tarjoama pilvipalvelu alusta.
Amazon S3	*Amazon Simple Storage Service*. Pilvitallennusalusta.
EC2	*Amazon Elastic Compute Cloud*. Joustava pilvipalveluinfrastruktuuri.
GPIO	*General-purpose input/output*. Yleiskäyttöinen portti, jota käytetään mikro-ohjaimissa ja mikroprosessoreissa.
IaaS	*Infrastructure as a Service*. Ulkoistetut palvelimet ja palvelinsalit.
Java	Sun Microsystemsin kehittämä ohjelmistoalusta, johon sisältyy oliopohjainen ohjelmointikieli.
NIST	*US National institute of standards and technology*. Yhdysvaltalainen virasto, jonka tehtävänä on edistää mm. standardeja.
PHP	*PHP: Hypertext Preprocessor*. Ohjelmointikieli, jota käytetään erityisesti dynaamisten web-sivustojen luonnissa.
PaaS	*Platform as a Service*. Pilvipalveluna hankittu ulkoistettu palvelualusta.
Python	Guido van Rossumin kehittämä monipuolinen ohjelmointikieli.
REST	*Representional State Transfer*. HTTP-protokollaan perustuva arkkitehtuurimalli, jolla toteutetaan ohjelmointirajapintoja.
SaaS	*Software as a Service*. Palveluna hankittu ohjelmisto.

TLS	*Transport Layer Security*. Salausprotokolla, jolla suojataan internet-sovellusten tietoliikenne IP-verkkojen yli.
WWW	*World Wide Web*. Internetverkossa toimiva hajautettu hypertekstijärjestelmä.
XaaS	*Anything as a Service*. Yleisnimitys kaikille mahdollisille tarjottaville palveluille.
SSL	*Secure Socket Layer*. Salausprotokolla, jonka TLS on myöhemmin korvannut

1 Johdanto

Yrityksille ja käyttäjille on ollut saatavilla pilvipalveluita hyvin laajamittaisesti jo useiden vuosien ajan. Pilvipalveluita voidaan kuvailla joukoksi yleiskäyttöisiä palveluita, jotka tarjoavat tietokonekapasiteettia sekä sovelluksia, jotka skaalautuvat käyttäjän tarpeiden mukaan. Nykyajan standardit asettavat koko ajan kovempia vaatimuksia palveluiden laadun suhteen, unohtamatta tietoturvaan kohdistuvia paineita.

Pilvipalvelut ovat ratkaisu moniin tietoteknisiin ongelmiin ja niiden suosio on jatkuvasti kasvanut yksityisellä sekä julkisella sektorilla. Yhä useammat yritykset siirtyvät käyttämään pilvipalveluita niiden houkuttelevan hinnoittelun ja ominaisuuksien takia. Pilvipalvelut tarjoavat laajan määrän vaihtoehtoisia ratkaisumalleja niin normaaleille käyttäjille kuin yrityksille, mutta niistä ei aina välttämättä ole kilpailijaa henkilökohtaisesti tuotetuille ratkaisuille. Pilvipalveluiden hyvät ja huonot puolet sekä tarpeellisuus tulee siis arvioida käyttäjäkohtaisesti.

Tässä insinöörityössä tullaan käymään läpi syitä pilvipalveluiden kasvaneeseen suosioon, pohditaan niiden tuomia etuja ja haittoja sekä perehdytään palveluiden erilaisiin rakenteisiin. Koska aina ei kannata tukeutua valmiiksi tuotettuun palveluun, on tässä työssä esitelty yksityiskohtaisesti yksityisen pilvipalvelun rakentaminen sekä käyttöönotto. Näin saadaan mukaan sisällytettyä käytännön näkökulma, joka tukee työn teknistä ja teoreettista osuutta.

2 Pilvipalvelut

2.1 Määritelmä

Pilvipalvelut (Cloud services) -termillä tarkoitetaan internetissä sijaitsevia hajautettuja ja ulkoistettuja palveluita, joita tarjotaan asiakkaille verkkovälitteisesti. Tämä mahdollistaa pääsyn vapaasti skaalautuviin ja konfiguroitaviin tietotekniikkaresursseihin, jotka voidaan kytkeä päälle tai pois hyvin nopeasti ja helposti, kuten US National institute of standards and technology (NIST) mainitsee julkaisemassaan määritelmässään [1]. Käsite on vielä nykyäänkin hyvin uusi ja siitä on käytössä monia rinnakkaisia termejä, esimerkiksi pilvilaskenta (Cloud computing).

Sanaa pilvi käytetään kuvatessa internetiä johtuen yleisesti käytetystä piirrosmallista, jolla viitataan monimutkaiseen infrastruktuuriin, jonka yksityiskohtia palvelun käyttäjät eivät voi nähdä tai täysin hallita. Tämä tarkoittaa sitä, että käyttäjä ei pysty tietämään halutessaan resurssien sijainnista, toiminnasta ja ylläpidosta kovin paljoa. Tätä rakennetta havainnollistetaan kuvassa 1, josta voidaan myös nähdä palveluiden käytön olevan päätelaitteista riippumatonta.

Kuva 1. Pilvipalveluiden toimintaperiaatteen havainnollistaminen. [2.]

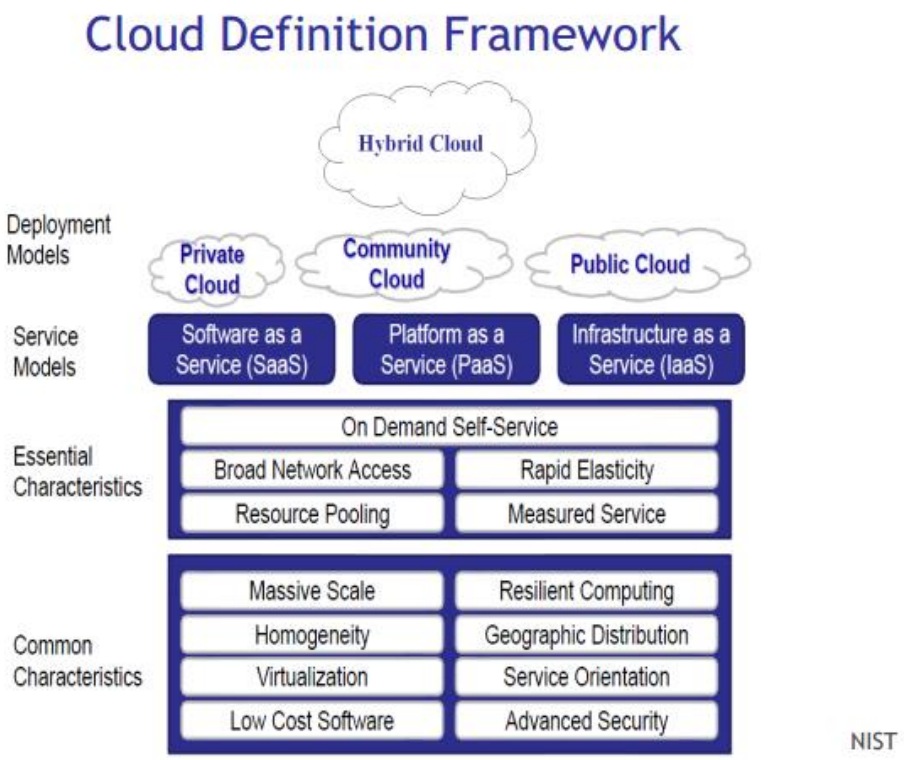
Pilvipalvelut voidaan myös luokitella ryhmittäin. SPI-pilviluokituksessa (Software Platform Infrastructure) pilvipalvelut jaetaan kolmeen ryhmään seuraavasti: sovellus palveluna (SaaS), sovellusalusta palveluna (PaaS) ja infrastruktuuri palveluna (IaaS). Yleisni-

mityksenä käytetään lyhennettä XaaS, joka tarkoittaa mitä tahansa pilvipalveluna tuotettua palvelua [3]. Pilvipalvelumallit tullaan käsittelemään tässä työssä tarkemmin vielä erikseen.

2.2 Tärkeimmät ominaisuudet

Koska pilvipalveluiden määritteleminen yhdellä tietyllä tavalla osoittautui todella hankalaksi, niin palvelut haluttiin määritellä myös niiden ominaisuuksien mukaan. NIST listasi viisi tärkeintä pilvipalveluiden ominaispiirrettä, jotka ovat

- itsepalvelu (On-Demand Self Service)
- päätelaitteesta riippumaton pääsy (Broad Network Access)
- jaetut resurssit (Recourse Pooling)
- joustavuus käytön tarpeiden mukaan (Rapid Elasticity)
- käytön mittaaminen (Measured Service).



Kuva 2. NIST-määritelmä pilvipalvelun rakenteesta. [4.]

Näiden lisäksi on olemassa vielä kahdeksan yleistä ominaisuutta (Common Characteristics), jotka löytyvät kuvan 2 alimmista riveistä. Nämä ominaisuudet kuvaavat pilvipalveluiden tarjoamia etuja. Pilvipalveluiden edullisuus perustuu käytön määrän mittaamiseen, eli käyttäjä maksaa vain siitä määrästä resursseja, mitä käyttää. Käyttöönotto on nopeaa, ja muutoksia saadaan tehtyä nopeasti. Jaetut resurssit tarkoittavat sitä, että käyttäjäyhteisöllä ei ole tarvetta ylläpitää omia IT-infrastruktuureja. Näin saadaan aikaan energiatehokkaampi ja yhdenmukaisempi palvelu, koska jokaisen ei tarvitse erikseen kiinnittää huomiota järjestelmän kokoonpanoon ja ylläpitoon.

2.3 Palveluiden tarjoaminen käyttäjille

Lähtökohtaisesti pilvipalveluiden tarjoamisen voidaan katsoa olevan hyvin suoraviivaista, koska kaikkia tekijöitä yhdistää sama yhteinen media, internet. Tärkeämpiä kysymyksiä ovat siis, minkälaista palvelua lähdetään hakemaan ja keneltä palvelu halutaan ostaa.

Amazon Web Services (AWS) tarjoaa laajan kokoelman palveluita, esimerkiksi virtuaalipalvelimia, tietokantoja ja talletusalustoja, joista keskeisimpiä ovat Amazon EC2 ja Amazon S3. Kaikki palvelut ovat käytettävissä verkkoselaimessa suoritettavassa käyttöliittymässä, ja niiden käyttöönotto on mahdollista muutamissa sekunneissa. Tunnusten luominen on ilmaista, ja AWS tarjoaa koekäyttömahdollisuuden tuotteisiinsa ennen ostopäätöstä. [5.]

Google Cloud Platform tarjoaa pitkälti samoja palveluita kuin AWS, mutta keskittyy huomattavasti enemmän ennen kaikkea erilaisiin www-kehitystyökaluihin ja virtuaaliympäristöihin, joissa voidaan suorittaa esimerkiksi Python- ja Java-ohjelmia. Jokainen palvelu pitää sisällään selainkäyttöliittymän, komentorivityökalun sekä API- ja REST-tuen [6]. Googlen kehitysympäristö soveltuu ohjelmien kirjoittamisesta aina internetsivustojen luomiseen saakka. Esimerkiksi Googlen tuoteperheeseen kuuluvat YouTube ja Google Search on toteutettu täysin käyttäen näitä työkaluja.

2.4 Palvelumallit

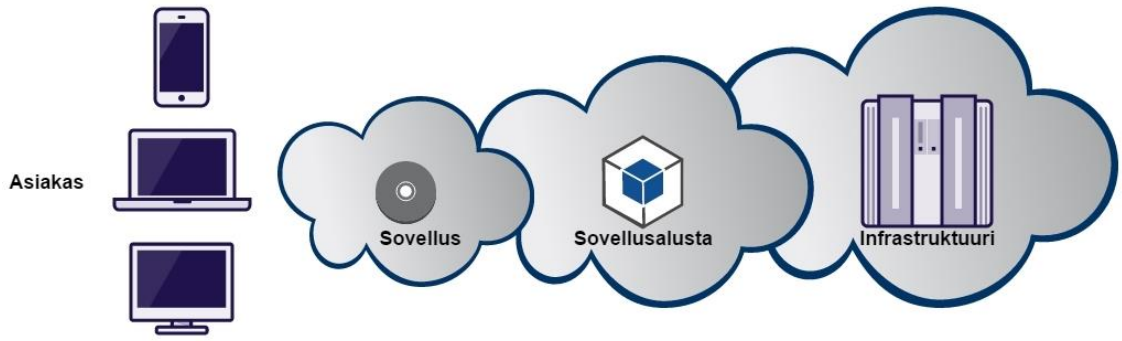
Pilvipalveluarkkitehtuuri jaetaan kolmeen eri palvelukerrokseen (Service Models): verkkosovellukset palveluna (SaaS), sovellusalusta palveluna (PaaS) ja infrastruktuuri palveluna (IaaS). Yleisesti pilvipalveluita voidaan esittää myös lyhenteellä XaaS (X as a

service) tai EaaS (Everything as a Service) [6.]. Taas voidaan siis nähdä, kuten aikaisemmin määrittelyssä huomasimme, termien rinnakkaisuus ja vakiintumattomuus. Omiksi palveluiksi voidaan myös luokitella muita palveluita, esimerkiksi tietoturva palveluna, tallennustila palveluna ja tietokannat palveluna. [7.]

2.4.1 Sovellukset palveluna

SaaS, eli ohjelmistona hankittu palvelu, antaa asiakkaalle käyttöön sovellukset, joiden asentamisesta, päivittämisestä ja ylläpidosta vastaa palveluntarjoaja. Käyttäjät tarvitsevat ainoastaan verkkoyhteyden sekä selaimen voidakseen käyttää palveluita. Maksuperusteena ei käytetä lisenssimaksua, vaan asiakas maksaa sovelluksista esimerkiksi aikaperusteisesti tai tietyn käyttäjäkohtaisen maksun. Tämä sovellusarkkitehtuuri mahdollistaa saman palvelun käytön laajan asiakaskunnan kesken, ja resurssit ovat samalla tehokkaasti käytössä. [1,s. 2.]

SaaS voidaan katsoa olevan käytännössä sama asia kuin ASP (Application Service Provider), joka on vanha termi kuvaamaan palveluiden tuottamista jostain ostetulla alustalla. Ohjelmistopalveluiden kehityksen ja virtualisoinnin ansiosta SaaS on käsitteenä korvannut vanhan ASP:n ja sitä voitaisiinkin pitää tämän kehittyneempänä versiona.

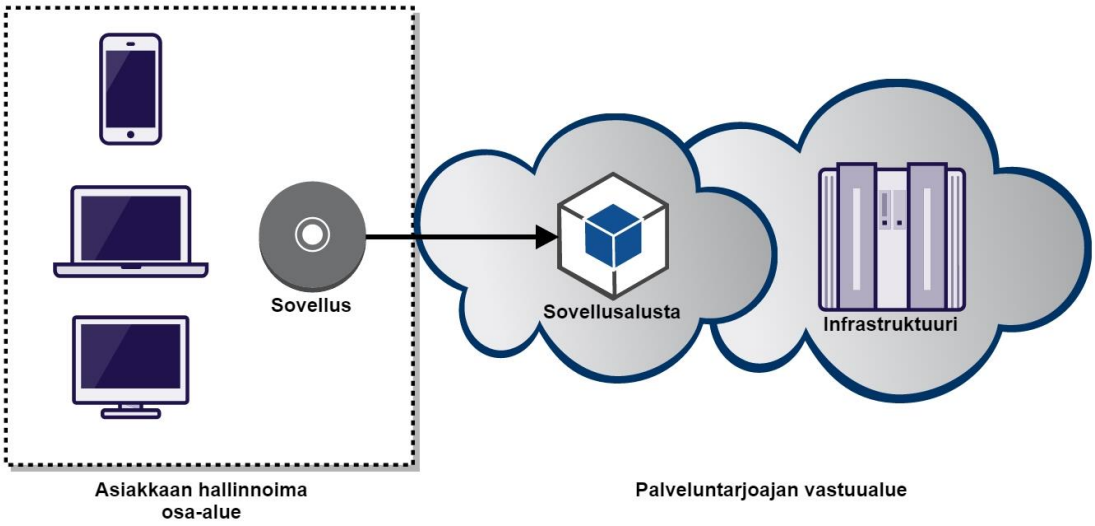


Kuva 3. SaaS-palvelumallissa asiakas ostaa sovelluksen käyttöönsä.

2.4.2 Sovellusalusta palveluna

PaaS tarkoittaa ulkoistettua palvelualustaa. Asiakas ostaa alustan palveluna, ylläpitää itse haluamiaan sovelluksia ja voi halutessaan kehittää myös omia sovelluksia. Asiakas ei pysty hallitsemaan tai valvomaan taustalla olevaa pilvipalvelua (verkko, tietokannat tai

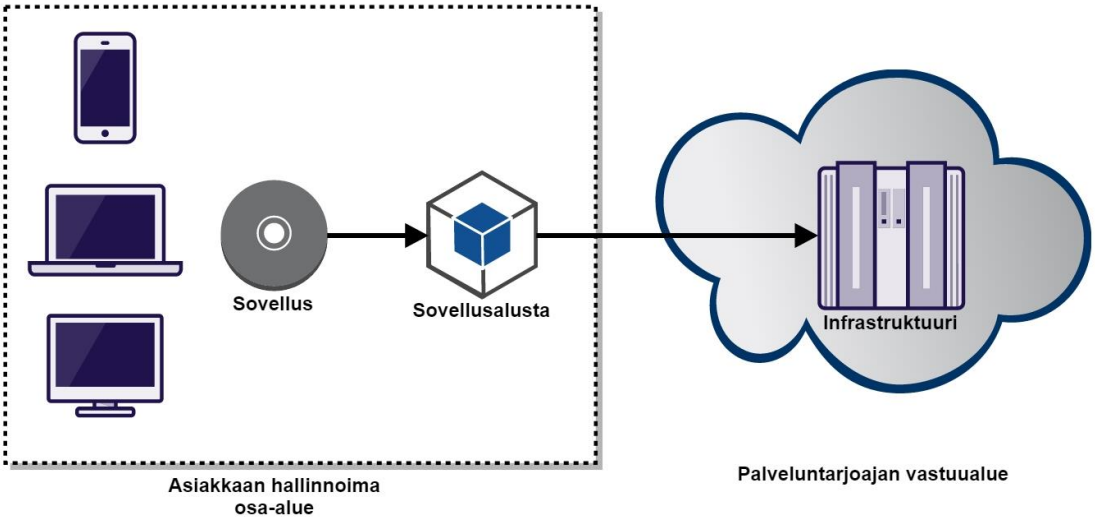
käyttöjärjestelmä). PaaS tarjoaa hyvät puitteet ohjelmistokehitykseen skaalautuvuutensa ansiosta, ja ohjelmiston käyttäjämäärän kasvaessa alustaa voidaan tarvittaessa laajentaa. [1, s. 2.]



Kuva 4. PaaS-palvelumallissa asiakas hallinnoi sovelluksia.

2.4.3 Infrastruktuuri palveluna

IaaS-palvelumallissa asiakas ostaa palveluna infrastruktuurin resurssit käyttöönsä. Tämä tarkoittaa virtualisoituja laitteistoja, jonka resursseista asiakas päättää itse. Laskentakapasiteetti, muisti sekä tallennustila varataan virtuaalikoneella, jota voidaan mukauttaa asiakkaan tarpeisiin myöhemmin uudestaan. Käyttäjä voi itse päättää käyttöjärjestelmästä ja sovelluksista sekä niiden ylläpidosta, mutta pilvi-infrastruktuuria ylläpitää ainoastaan palveluntarjoaja. [1,s. 3.]



Kuva 5. IaaS-palvelumallissa palveluntarjoaja vastaa vain resursseista.

2.5 Käyttöönottomallit

Pilvipalveluille on olemassa neljä erilaista käyttömallia (Deployment Models): julkinen, yksityinen ja yhteisöllinen pilvi sekä hybridipilvi. Niiden määrittelyssä on otettu huomioon kenellä, tai keillä on pääsy palveluun ja kuinka suuri vastuu on pilvipalveluntarjoajalla.

Julkinen

Julkinen pilvi (Public Cloud) on kaikkien käyttäjien ulottuvissa oleva avoin pilvi-infrastruktuuri, jonka omistajuus on kolmannen osapuolen pilvipalvelutarjoajalla. Tarjolla olevat palvelut voivat olla ilmaisia tai maksullisia. Lisäksi niitä voidaan käyttää laskutusta käytettyjen resurssien perusteella. Käyttäjille tämä käyttömalli tarjoaa tehokkaan sekä joustavan tavan hyödyntää niitä resursseja, joita tarvitaan: ajasta ja paikasta riippumatta. Hyötyäkseen julkisesta pilvestä käyttäjän täytyy kuitenkin myöntyä rajallisiin säätö- ja hallintatyökaluihin sekä luottaa palveluntarjoajan tietoturvan olevan asianmukaista. [1, s. 3.]

Yksityinen

Yksityinen pilvi (Private Cloud) on yksinomaan yrityksen tai organisaation käyttöön varattu pilvipalvelu, joka tunnetaan myös nimellä sisäinen pilvi. Palvelu voi olla kokonaan yrityksen hallinnoima tai se on ostettu kolmannelta osapuolelta. Tyypillisesti kuitenkin

yritys ylläpitää omaa yksityistä pilveä, johon yhdistetään suojatulla yhteydellä muista toimipisteistä ja tietoa jaetaan vain yrityksen sisällä [1, s. 3]. Näin saavutetaan hyvin valvottu ja kontrolloitu pilvi sekä huomattavasti parempi tietoturva. Koska yritys tai organisaatio varaa tässä mallissa kaikki resurssit käyttöönsä, on sitä kritisoitu siitä, ettei siinä toteudu ulkoistuksen kautta saatavat taloudelliset edut.

Yhteisöllinen

Yhteisöllinen pilvi (Community Cloud) on useamman kuin yhden yrityksen tai organisaation kanssa jaettu infrastruktuuri, jossa yhdistyvät yhteiset intressit kuten paranneltu tietoturva tai arkaluontoisten dokumenttien käsittely. Palvelu voi olla yritysten tai kolmannen osapuolen hallinnoima sekä myös näiden kombinaatio. Resurssien käyttö ja kustannukset on rajoitettu ainoastaan määritetylle yhteisölle, joten osa kustannus säästöistä toteutuu. [1, s. 3.]

Hybridi

Hybridipilvi on kahden tai useamman edellä mainittujen käyttöönottomallien yhdistelmä. Yritys voi esimerkiksi pitää ei-arkaluontoiset tiedostot julkisessa pilvessä, mutta henkilökohtaiset asiakastiedot tallennettaisiin yksityiseen pilveen, tai tarvittaessa jopa omaan tietokeskukseen tietoturvallisuussyistä. Tällä menettelyllä pystytään määrittelemään useita pilvipalveluita, jotka on kytketty toisiinsa. Se mahdollistaa tiedon helpon siirtämisen ja käyttöönoton muualla, esimerkiksi palvelun kuormituksen jaon. Hybridi käyttöönottomalli pyrkii yhdistelemään muiden mallien parhaat puolet, kuten julkisen pilven skaalautuvuuden, ja yksityisen pilven kontrollin sekä tietoturvan [1, s. 3]. Muun muassa Microsoft Azure ja Force.com hyödyntävät tätä mallia omissa palveluissaan.

2.6 Pilvipalveluiden tietoturva

Pilvipalveluiden tietoturvaan vaikuttaa monta tekijää, ja useammat niistä eivät välttämättä näy suoraan loppukäyttäjälle. Teknisen toteutuksen lisäksi palvelua hallinnoiva henkilöstö on tärkeässä asemassa, sekä laitteistojen ja fyysisen ympäristön huolto ja toimittajaketju.

Kun pohditaan tietoturvaa, on syytä kiinnittää huomiota palvelun toteutuksen laatuun ja palveluntarjoajan toimintaan kokonaisuutena. Luotettava pilvipalveluntarjoaja ymmärtää

käyttäjien tarpeet ja halun varmistua toiminnan turvallisuudesta. Se tekee toiminnasta mahdollisimman avointa ja läpinäkyvää. Käyttäjille voidaan tarjota kattava dokumentointi palvelun käytännön toimista sekä sertifiointeja ja auditointeja kolmannen osapuolen toimesta. Kilpailusyistä palveluntarjoaja ei voi paljastaa kaikkia palvelun yksityiskohtia, ja se saattaisi myös johtaa tietoturvariskeihin.

Käyttäjille näkyviltä palvelun osilta voidaan myös tehdä johtopäätöksiä. Ohjelmiston turvallisen käytön takaamiseksi käyttäjänhallinnan on oltava kunnossa. Tämä tarkoittaa asiakkaiden rekisteröitymistä palveluun ja tunnistautumista sekä aktiivista tarkkailua käyttöoikeuksissa. Jos palveluntarjoajalla on tarjolla myös muita tuotteita, niiden tutkiminen ja toimintakulttuurin selvittäminen voi antaa käsityksen toiminnan luotettavuudesta, ja jotain voidaan myös päätellä yrityksen yleisestä maineesta. Hyvä tietoturva vaatii jatkuvaa tarkkailua ja ylläpitoa sekä sujuvaa palvelun päivittämistä. Käyttäjän on siis hyvä selvittää palvelun päivityskäytännöt, vaikkakin on hyvin todennäköistä, että isoimpien palveluntarjoajien päivityskäytännöt ovat hyvin pitkälti automatisoitu ja resursoitu. Asiakkaan on kuitenkin hyvä varmistua, että päivitysmenettely sopii omiin tarpeisiin.

Tyypillinen pilvipalvelu käyttää hyväkseen selainpohjaista käyttöliittymää, jonka avulla asiakas hallinnoi palveluaan ja tallentaa siihen tietoja. Verkkoyhteyttä hyödyntävässä ohjelmistossa täytyy kuitenkin varmistaa, että käytettävät yhteydet on salattu. Näin voidaan ainakin teoriassa varmistaa palvelun käytön turvallisuus. Esimerkiksi selaimessa osoiterivillä näkyvä etuliite `https://` tai lukon kuva kertoo, että käytössä on turvallinen protokolla.

Fyysinen turvallisuus kattaa kaiken aina sähkön jakelun keskeytymisestä aina verkkoliikenteen katkeamiseen saakka. Epätodennäköisissäkin tilanteissa palveluntarjoaja voi taata palvelun jatkuvuuden varaamalla useamman tietoliikenneyhteyden ja kahdentamalla palvelun toiminnan. Valvomalla, kenellä on pääsy palvelun fyysisiin tiloihin minimoidaan turhat vahingot.

Infrastruktuuripalvelu tasolla asiakkaalla on suuri vastuu myös omasta sovellusohjelmastaan. Kokonaisturvallisuus koostuu asiakkaan sovelluksen ja palveluntarjoajan komponenttien yhteistoiminnasta. Näin ollen huonosti suunniteltu sovelluksen tietoturva ei parane laisinkaan siirrettynä pilvipalveluun.

Pilvipalveluiden etähallinta mahdollistaa uudenlaisia hyökkäysmalleja, ja esimerkiksi vääriin käsiin joutuneet tunnukset saattavat johtaa tietojen kopioimiseen tai tuhoutumiseen. Tämä vastaisi perinteisessä tapauksessa suoraa pääsyä palvelintilaan. Toinen mahdollinen hyökkäys palveluun saattaisi muodostua käytettyjen ohjelmointirajapintojen (API) välityksellä. Jos hyökkääjä saisi käsiinsä hallintaan tarvittavat avaimet, ne mahdollistaisivat uusien sovellusohjelmien ajamisen, tietojen kopioinnin ja jopa palvelimien hyödyntämisen muihin laajempimittaisiin hyökkäyksiin.

Tietoturvallisuuteen vaikuttaa suuresti myös itse käyttöympäristö. Käyttöympäristöön sisältyvät niin päätelaitteet kuin käyttäjät sekä erilaiset järjestelmät, joita hyödynnetään pilvipalveluiden kanssa. Perinteinen turvallisuus on siis aivan yhtä ajankohtainen pilvipalveluita käytettäessä kuin mitä tahansa palvelua.

3 Henkilökohtaisen pilvipalvelun suunnittelu

Pilvipalvelun suunnittelussa kiinnitettiin lähtökohtaisesti erityisesti huomiota sen käyttöönoton edullisuuteen ja käytännöllisyyteen. Koska palveluiden tarkoituksena on olla käytettävissä ajasta tai paikasta riippumatta, missä ja milloin tahansa, käytettävän laitteiston koko ja virrankulutus pyrittiin minimoimaan täysin. Palveluiksi valittiin Wordpress ja ownCloud, mikä johtuu tämäntyyppisten palveluiden kasvaneesta suosiosta nykypäivänä.

Tässä osassa työtä tullaan käsittelemään henkilökohtaisen pilvipalvelun käyttöönoton kannalta oleellisimmat laitteet, ohjelmistot sekä työkalut. Täten tästä syystä on hyvä huomioida, että palvelun toimivuuden kannalta välttämättömät tekijät, kuten internet-yhteys, lähiverkon rakenne ja laitteisto eivät ole lähemmässä tarkastelussa. Näihin seikkoihin tullaan kiinnittämään huomiota ainoastaan sen ollessa työn kannalta välttämätöntä ja haluttuun lopputulokseen olisi muuten mahdotonta päästä.

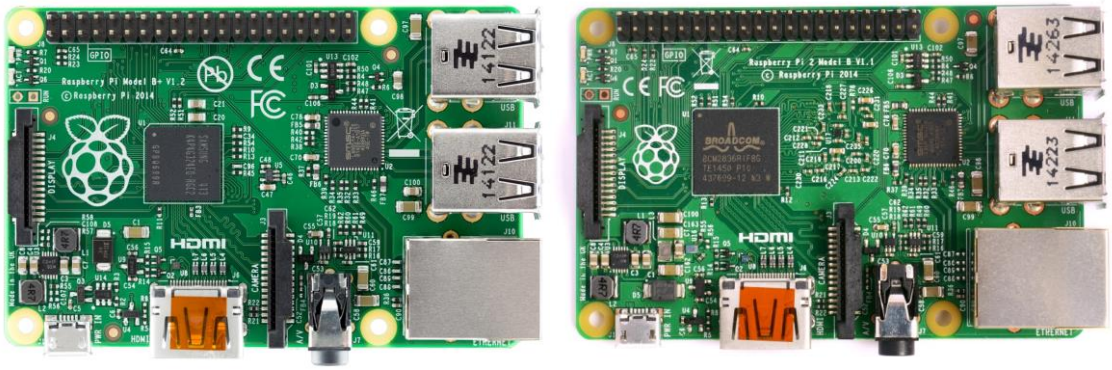
3.1 Fyysiset laitteet

Raspberry Pi

Raspberry Pi on luottokortin kokoinen yhden piirilevyn tietokone, jonka kehitti brittiläinen Raspberry Pi Foundation, ja se julkaistiin helmikuussa 2012. Kolme vuotta myöhemmin helmikuussa 2015 säätiö julkaisi Raspberry Pi 2:n, joka kulki edeltäjänsä jalanjäljissä

tarjoten hieman tehokkaamman ja parannellun suorittimen sekä kaksinkertaistetun keskusmuistin. Ensimmäisestä mallista ehdittiin tehdä neljä erilaista versiota (Model A, A+, B ja B+), jotka erosivat toisistaan ainoastaan liitäntöjen määrissä ja muistin koossa. Säätiön tavoitteena on kannustaa tietotekniikan opetukseen kouluissa. Sitä tukemassa ovat Cambridgen yliopisto ja piirivalmistaja Broadcom.

Laitteiston kokoonpano perustuu molemmissa malleissa Broadcomin valmistamaan järjestelmäpiiriin, joka sisältää suorittimen, muistin ja integroidun grafiikkapiirin. Tietokoneeseen voidaan kytkeä HDMI-liitäntäinen monitori. Syöttölaitteina voi käyttää USB-liitäntäisiä näppäimistöjä ja hiiriä. Käyttövirtansa tietokone saa mikro-USB-liitännästä ja virtalähteenä voi toimia esimerkiksi kännykän laturi. Suurimmat teknilliset erot ensimmäisen ja viimeisimmän julkaistun mallin on listattu taulukkoon numero yksi.



Kuva 4. Vasemmalla Raspberry Pi A, oikealla Raspberry Pi 2 B

Taulukko 1. Raspberry Pi:n tekniset tiedot

	Raspberry Pi Model A	Raspberry Pi 2 Model B
Prosessori:	700MHz ARM11	A 900MHz ARM Cortex-A7
Grafiikkapiiri:	Dual Core VideoCore IV® 250MHz, OpenGL ES 2.0, 1080p h.264	
Muisti:	256MB SDRAM 400MHz	1GB SDRAM
Verkkosovitin:	Ei	10/100 (USB-Ethernet silta)
Tehonkulutus:	300mA (1.5W)	800mA (4.0W)
Virtalähde:	5V MicroUSB	
Massamuisti:	SD/MMC/SDIO	MicroSD
Muut liitännät:	8kpl GPIO-liittimiä	17kpl GPIO-liittimiä

3.2 Käyttöjärjestelmä

Rasbian

Rasbian on Debian Linux-jakelupakettiin pohjautuva ilmainen käyttöjärjestelmä, joka on optimoitu erityisesti Raspberry Pi:n laitteistolle. Se pitää sisällään yli 35 000 valmiiksi käännettyä ja optimoitua ohjelmistoa yksinkertaisessa helposti käyttöön otettavassa muodossa. Ensimmäinen valmis versio julkaistiin kesäkuussa 2012, ja se on edelleen aktiivisessa kehityksessä fanien toimesta. Rasbian ei ole sidoksissa Raspberry Pi Foundationin kanssa, vaan taustalla toimii aktiivinen yhteisö, joka pitää säätiön tavoitteita ihailtavana.

3.3 Pilvipalvelut

3.3.1 ownCloud

OwnCloud on avoimen lähdekoodin ohjelmisto, joka tarjoaa käyttäjälle oman henkilökohtaisen tallennustilan ja tiedostojen synkronoinnin palveluna. Tästä syystä kuka tahansa voi asentaa sen omalle henkilökohtaiselle palvelimelleen ilman rajoituksia tallennustilan, käyttäjien lukumäärän tai liikuteltavan datan määrän suhteen. OwnCloud on ohjelmoitu PHP- ja JAVA-kielellä ja se on suunniteltu toimimaan useiden tietokannan hallintajärjestelmien kanssa (mm. SQLite, MariaDB, MySQL, Oracle Database). Käyttäjän näkökulmasta se muistuttaa hyvin paljon suosittua Dropbox-palvelua, mutta se kätkee sisälleen huomattavasti monipuolisemmat työkalut. [8.]

Tiedostojen synkronointia varten täytyy käyttäjän asentaa pääteohjelma, joka on saatavissa Windows-, FreeBSD- tai Linux-järjestelmille. Mobiililaitteille kuten iOS ja Android on saatavissa omat ownCloud-asiakasohjelmat. Tiedostojen lataaminen ja hallinta on myös mahdollista selainkäyttöliittymän välityksellä ilman lisäohjelmia. Suojauksesta palvelinpuolella huolehtii TLS-salausprotokolla.

3.3.2 WordPress

WordPress on vuonna 2003 julkaistu avoimeen lähdekoodiin perustuva sisällönhallintaohjelmisto, joka alkujaan keskittyi erityisesti blogien luomiseen ja ylläpitoon. Ohjelmointikielenä toimii PHP, ja tietojen tallentamisessa hyödynnetään MySQL-tietokantaa, jonka

täytyy olla asennettuna palvelimelle. Ohjelmistoversio 3.0:n jälkeen käyttäjät ovat voineet luoda useita sivustoja hyödyntäen ainoastaan yhtä asennuspakettia ja tietokantaa.

Lisäosiensa ja aktiivisen kehityksen myötä siitä on tullut webin suosituin sisällönhallintaohjelmisto. Internetin 10 miljoonasta suosituimmasta sivustosta 23,3 prosenttia käytti sivujensa luomiseen WordPress-ohjelmistoa [9]. WordPressin virallinen sivusto tarjoaa mahdollisuuden oman blogin luomiseen, mutta sen käyttöä on rajoitettu vain tiettyihin teemoihin ja lisäosiin.

3.4 Viimeistely

TLS / SSL

TLS (Transport Layer Security) ja SSL (Secure Socet Layer) ovat molemmat salausprotokollia, joilla voidaan suojata verkon tietoliikenne. Tavallisin käyttökohde on WWW-sivustojen suojaaminen HTTP-protokollalla, jossa TLS-versio 1.2 [10] on korvannut vanhemmat SSL-versiot.

SSL:n toiminta perustuu varmenteisiin, joiden avulla sivusto todistaa olevansa hyvämaineinen ja luotettava. Varmenteita myöntäviä yrityksiä on lukuisia ja heidän tehtävänään on taata varmenteen hakijan identiteetti. Selainvalmistajat puolestaan listaavat luotettavia varmenteen myöntäjiä, ja nämä voivat valtuuttaa myöntämisoikeuksia edelleen.

Suojatun SSL-yhteyden luomiseen tarvitaan kolme avainta: julkinen, yksityinen ja istuntoavain. Kaikki tieto, mikä on salattu julkisella avaimella, voidaan lukea ainoastaan yksityisellä avaimella ja toisin päin. Koska julkisella ja yksityisellä avaimella salaaminen ja purkaminen vie huomattavasti enemmän resursseja, käytetään niitä ainoastaan istunnon alussa SSL-käyttelyvaiheessa (Handshake) symmetrisen istuntoavaimen luomiseen. Kun luotettava yhteys on luotu, käytetään liikenteen salaamiseen ainoastaan istuntoavainta.

4 Henkilökohtaisen pilvipalvelun käyttöönotto

4.1 Rasbianin asennus

Rasbian on ladattavissa .zip-pakattuna ilmaiseksi Raspberry Pi Foundationin virallisilta sivuilta osoitteesta `https://www.raspberrypi.org/downloads/`. Halutessaan lataus voidaan suorittaa torrent-verkon välityksellä. Tarvittavat linkit löytyvät samalta sivulta.

Debianin asennus tapahtuu kirjoittamalla ladattu levykuva microSD-muistikortille, ja tämä on mahdollista toteuttaa Linux-, Mac- tai Windows-käyttöjärjestelmillä. Riippuen käytössä olevasta järjestelmästä vaiheet voivat olla toisistaan hyvin poikkeavat ja tässä työssä käsitellään ainoastaan asennus-Linux-järjestelmää käyttäen.

Ensimmäiseksi microSD-muistikortti kytketään tietokoneeseen ja listataan löydetyt tiedostojärjestelmät komennolla:

```
$ df -h
Filesystem 1K-blocks Used Available Use% Mounted on
/dev/sda5 75684968 8647364  63169956  13% /
*
*
/dev/sdb1 57288   14736     42552  26%
/media/nc10/boot
/dev/sdb2 7534284 3112340   4065628  44%
/media/nc10/f24a4949-f4b2-4cad-a780-a138695079ec
```

Todetaan, että kytketty 8GB microSD-kortti pitää sisällään kaksi osiota nimeltään `boot` ja `f24a4949..`, jotka irrotetaan järjestelmästä seuraavaksi komennolla

```
$ umount /dev/sdb1 && umount /dev/sdb2
```

Ladataan rasbianin uusin versio virallisilta sivuilta ja puretaan se aktiivisena olevaan hakemistoon:

```
$ wget http://downloads.raspberrypi.org/raspbian_latest
$ unzip raspbian_latest
```

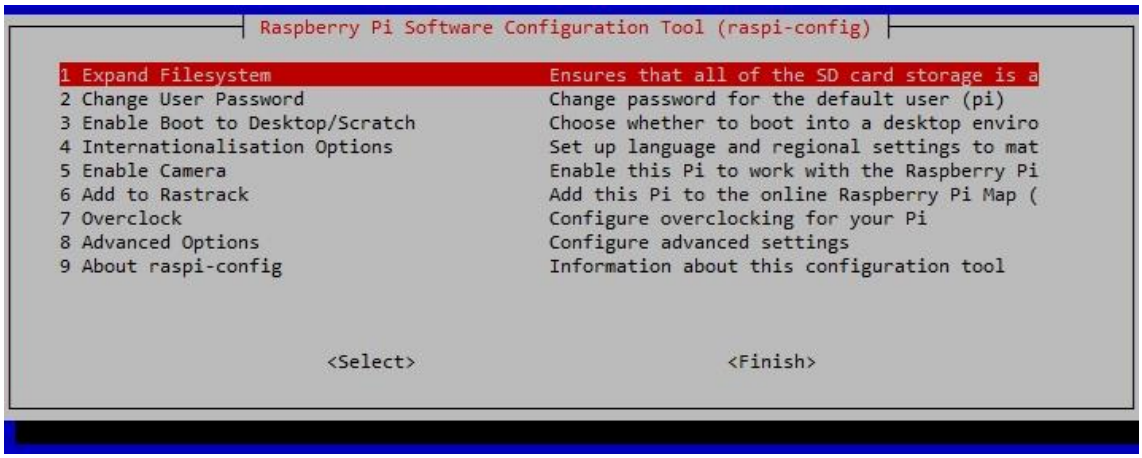
```
Archive: raspbian_latest
inflating: 2015-02-16-raspbian-wheezy.img
```

Seuraavaksi kirjoitetaan hakemistoon purettu levykuva muistikortille ja tyhjennetään jälkikäteen kirjoitusvälimuisti komennoilla:

```
$ sudo dcfldd bs=4M if=2015-02-16-raspbian-wheezy.img
of=/dev/sdb
768 blocks (3072Mb) written.
781+1 records in
781+1 records out
$ sync
```

Kirjoittaminen vaatii root oikeudet. Käyttäjän täytyy olla kirjautuneena root-pääkäyttäjänä tai käyttää `sudo`-etuliitettä. Joissakin tapauksessa kirjoituslohkon koko `bs=4M` saattaa aiheuttaa ongelmia ja silloin kokoa kannattaa pienentää, mutta prosessi vie silloin hieman pidempään. Parametri `if` viittaa luettavaan tiedostoon ja `of` puolestaan kirjoitettavaan kohteeseen. On hyvä huomioida, että kohteeseen viitataan `/dev/sdb`, eikä suinkaan johonkin sen sisältämään osioon (`sdb1, sdb2`). Näin varmistetaan, että ohjelma luo kortille tarvittavat osiot itse.

Kun microSD-muistikortti on asennettu paikalleen muiden oheislaitteiden kanssa, voidaan laite käynnistää ensimmäisen kerran. Tämä tapahtuu automaattisesti kytkemällä micro-USB-laturi laitteeseen, koska Raspberry Pi ei sisällä omaa kytkintä tähän tarkoitukseen.



Kuva 5. `Raspi-config`-ikkuna käynnistyksen yhteydessä

Ensimmäisenä eteen aukeaa `raspi-config`-sovellus (kuva 5), jonka avulla määritellään muutama tärkeä asetus. On suositeltavaa vaihtaa oletuskäyttäjän (pi) salasana ja laitteen isäntänimi sekä kytkeä päälle SSH-palvelin myöhempää käyttöä varten. Tehtyjen muutoksien jälkeen laite käynnistetään uudelleen valitsemalla `<Finish>` ja vastaamalla `Yes`, kun ohjelma kysyy uudelleenkäynnistystä.

Järjestelmään kirjauduttua muokataan verkkoyhteyksien asetuksia kirjoittamalla muutama rivi `interfaces`-tiedostoon:

```
$ sudo nano /etc/network/interfaces
```

Muutetaan laitteen `eth0`-osoite dynaamisesta staattiseksi ja määritellään ip-osoite korvaamalla tiedoston sisältö seuraavilla riveillä:

```
auto lo
iface lo inet loopback
iface eth0 inet static
address 192.168.10.50
netmask 255.255.255.0
network 192.168.10.0
gateway 192.168.10.1
```

Tehtyjen muutoksien jälkeen tekstieditori suljetaan näppäinyhdistelmällä Ctrl+X, käynnistetään laitteet uudelleen ja tarkistetaan verkkokortin asetukset komennoilla:

```
$ sudo services networking restart
$ ifconfig
eth0      Link encap:Ethernet HWaddr **:**:**:**:**:**
inet addr:192.168.10.50 Bcast:192.168.10.255
Mask:255.255.255.0
UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
RX packets:17382 errors:0 dropped:6 overruns:0 frame:0
TX   packets:22434   errors:0   dropped:0   overruns:0
carrier:0
collisions:0 txqueuelen:1000
RX bytes:5430932 (5.1 MiB)  TX bytes:12245361 (11.6 MiB)
```

4.2 Apachen asennus ja konfigurointi

Molemmat myöhemmin työssä asennettavat palvelut käyttävät hyväkseen selainpohjaista käyttöliittymää, jotka toimiakseen tarvitsevat HTTP-palvelinohjelman. Apache HTTP Server on avoimeen lähdekoodiin perustuva ohjelma, joka on saatavilla Linuxin lisäksi myös muille yleisimmille käyttöjärjestelmille [11]. Asennus Debianissa tapahtuu ajamalla terminaalissa komento:

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

Oletushakemistona sivustoille Apache käyttää `/var/www/`-polkua, jota mekin käytämme myös tässä työssä. Koska Raspberry Pi tulee toimimaan palvelimena useammalle kuin yhdelle sivustolle, täytyy oletushakemistoon tehdä pieniä muutoksia. Ensimmäisenä luodaan jokaiselle palvelulle oma hakemisto.

```
$ cd /var/www/
$ sudo mkdir -p vhost/wordpress
$ sudo mkdir vhost/owncloud
```

Vakiona Apachen oletushakemiston `/var/www/` omistusoikeudet kuuluvat root-pääkäyttäjälle. Apache puolestaan käyttää omaa käyttäjätunnusta ja ryhmää nimeltä `www-data`, jolle annamme omistusoikeudet tiedostojen muokkaamista ja ylläpitoa varten.

```
$ sudo chown -R www-data:www-data /var/www
```

Seuraavaksi annetaan `www-data`-ryhmälle oikeus kirjoittaa hakemistoon ja lisätään vielä käyttäjä `pi` kyseiseen ryhmään.

```
$ sudo chmod 775 /var/www
$ sudo usermod -a -G www-data pi
```

4.2.1 SSL-avaimien luonti

SSL-avaimien luominen aloitetaan asentamalla OpenSSL-ohjelmisto, aktivoimalla Apachen SSL-moduuli ja luomalla avaimille omat hakemistot komennoilla:

```
$ sudo apt-get install openssl
$ sudo a2enmod ssl
$ sudo mkdir -p /etc/apache2/ssl/wordpress-
inssi.ddns.net
$ sudo mkdir /etc/apache2/ssl/owncloud-inssi.ddns.net
```

Seuraavaksi luodaan itse allekirjoitettu SSL-sertifikaatti, määritellään sen voimassaoloaika ja generoidaan sertifikaatin allekirjoituspyyntö CSR (Certificate Signing Request) molemmille sivuille.

```
$ sudo openssl req -x509 -nodes -days 365 -newkey
rsa:2048 -keyout /etc/apache2/ssl/wordpress-
inssi.ddns.net/apache.key -ot
/etc/apache2/ssl/wordpress-inssi.ddns.net/apache.crt

$ sudo openssl req -x509 -nodes -days 365 -newkey
rsa:2048 -keyout /etc/apache2/ssl/owncloud-
inssi.ddns.net/apache.key -ot
/etc/apache2/ssl/owncloud-inssi.ddns.net/apache.crt
```

Komentoja suoritettaessa pyydetään täyttämään erilaisia tietoja, jotka yhdistetään sertifikaattiin. Kaikkia kenttiä ei tarvitse täyttä, mutta ehdottomasti tärkein kohta on `Common name`, jonka täytyy olla yhdenmukainen sivuston osoitteen kanssa. Alla on esimerkkinä ownCloud-palvelulle luotu sertifikaatti.

```
Common Name (e.g. server FQDN or YOUR name)
[]:owncloud-inssi.ddns.net
```

Näin saatiin luotua SSL-sertifikaatit molemmille sivustoille, joita tullaan käyttämään myöhemmin, kun otamme sivustot käyttöön.

4.2.2 Apache Virtual Host

Apache on hyvin monipuolinen työkalu, ja usean sivun isännöinti voidaan toteuttaa monella tavalla. Koska itselläni on käytössä yksi julkinen IP-osoite, jota haluan käyttää sivujeni tarjoamiseen, käytetään tässä tapauksessa nimipohjaista virtuaalista hostausta

(Name-based Virtual Hosting). Tämä vaatii `/etc/apache2/sites-availabe/default-ssl`-tiedoston laajempaa muokkausta. Kokonaisuudessaan koko konfiguraatio löytyy liitteet osiosta. Mainitaan tähän kuitenkin muutamia tärkeitä tiedoston kohtia:

- `VirtualHost` käskee Apachea vastaamaan pyyntöihin portissa 443
- `ServerName` sivuston nimi
- `ServerAlias` lisänimiä jota Apache vertaa päätunnisteeseen
- päättäessään tarjottavaa sivustoa
- `DocumentRoot` polku hakemistoon joka sisältää sivuston tiedostot
- `ErrorLog` virheloki ja sen tallennuspolku
- `SSLCertificateFile` määritetään sertifikaatin polku
- `SSLCertificateKeyFile` määritetään yksityisen avaimen polku.

Kun tiedosto on muokattu ja tallennettu, tarvitsee vielä luoda symbolinen linkki, joka ottaa sivuston käyttöön. Tämä tapahtuu ajamalla `a2ensite`-skripti halutulle sivustolle (tässä tapauksessa `default-ssl`), joka sisältää määritellyt `VirtualHost`-lohkot ja käynnistämällä apache uudelleen.

```
$ sudo a2ensite default-ssl
Enabling site default-ssl.
To activate the new configuration, you need to run:
service apache2 reload

$ sudo service apache2 reload
[ ok ] Reloading web server config: apache2.
```

Nyt meillä on luotuna kaksi sivustoa `https://wordpress-inssi.ddns.net` ja `https://owncloud-inssi.ddns.net` omilla hakemistoilla ja lokeilla. Sivustoihin tullaan jatkossa viittaamaan kyseisillä osoitteilla suoraan, koska käytössä on no-ip.comin tarjoama ilmainen, dynaaminen nimipalvelu [12].

4.3 WordPressin asennus ja konfigurointi

Wordpress on kirjoitettu PHP-ohjelmointikielellä, joten ensimmäisenä asennetaan tarvittavat PHP- ja Apache PHP5 -moduulit.

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

Seuraavaksi asennetaan MySQL-palvelin ja moduuli, joka mahdollistaa yhteyksien luomisen tietokantoihin suoraan PHP-skripteistä. Asennuksen aikana pyydetään luomaan root-salasana MySQL-palvelulle. Tämä tulee muistaa tulevia asennusvaiheita silmällä pitäen.

```
$ sudo apt-get install mysql-server php5-mysql -y
```

WordPressin asennuspaketti on saatavilla `wordpress.org` virallisilla sivuilla ja sen noutaminen onnistuu `wget`-komennolla. Uusin versio on aina saatavissa osoitteissa `wordpress.org/latest.tar.gz` ja `wordpress.org/latest.zip`.

Ensimmäiseksi siirrytään jo aikaisemmin luotuun, WordPressille varattuun hakemistoon ja ladataan paketti.

```
$ cd /var/www/vhost/wordpress
$ wget http://wordpress.org/latest.tar.gz
```

Puretaan tarball, siirretään tiedostot oikeaan hakemistoon ja poistetaan jäljelle jäänyt tyhjä hakemisto.

```
$ tar xzf latest.tar.gz
$ mv wordpress/* .
$ rm -rf wordpress latest.tar.gz
```

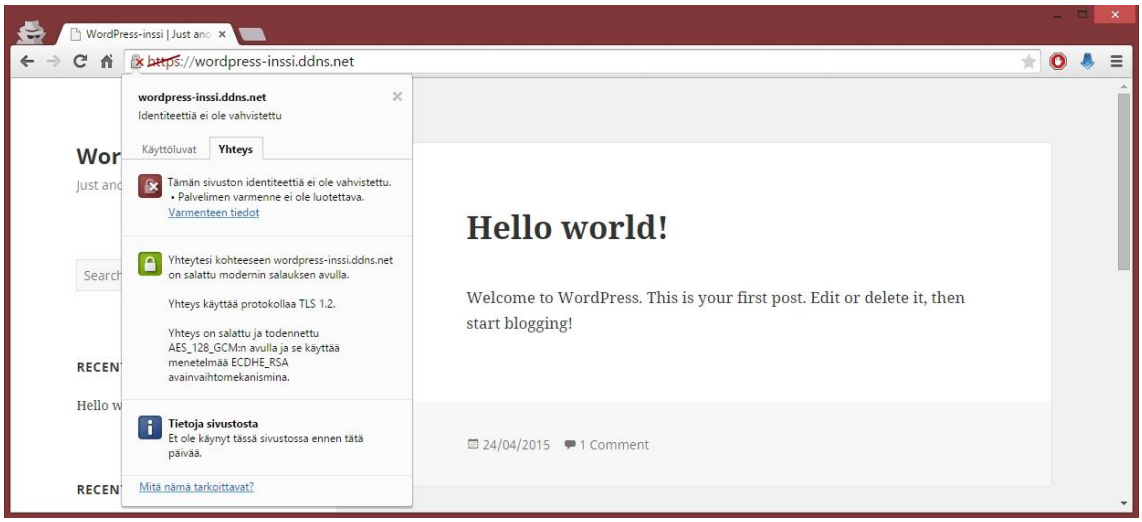
Toimiakseen WordPress tarvitsee oman tietokannan. Kirjaudutaan mysql-sovellukseen root käyttäjätunnuksella ja luodaan uusi tietokanta komennoilla:

```
$ mysql -u root -p
mysql> create database wordpress;
Query OK, 1 row affected (0.00 sec)
```

Nyt voimme siirtyä selaimella osoitteeseen `https://wordpress-inssi.ddns.net/`, jossa pyydetään täyttämään äskettäin luodun tietokannan tiedot.

```
Database Name:      wordpress
User name:          root
Password:           <MySQL root salasana>
Table Prefix:       wp_
```

Kun tarvittavat tiedot on syötetty, voidaan suorittaa asennus loppuun painamalla `Run the install` -painiketta. Seuraavaksi kysytään sivuston nimeä ja luodaan käyttäjätunnukset kirjautumista ja myöhempää sivuston muokkaamista varten. Nyt meillä on käytössä täysin toimiva asennus WordPress-alustasta, joka on saatavissa osoitteessa `https://wordpress-inssi.ddns.net/`. Sivustolla vierailtaessa tosin aukeaa ensimmäiseksi varoitussivu ”This Connection is untrusted” johtuen luotettavan sertifikaattiallekirjoituksen puuttumisesta.



Kuva 6. Onnistuneen asennuksen tuloksena luotu WordPress-sivusto, joka käyttää HTTPS-protokollaa.

4.4 ownCloudin asennus

OwnCloudin asennus aloitetaan lataamalla paketit, jotka ovat palvelun toimivuuden kannalta välttämättömiä.

```
$ sudo apt-get install apache2 php5 php5-gd php- xml-
parser php5-intl
$ sudo apt-get install php5-sqlite php5-mysql
smbclient curl libcurl3 php5-curl
```

Seuraavaksi siirrytään palvelulle varattuun hakemistoon, ladataan ownCloudin asennuspaketti virallisilta sivuilta `wget`-komennolla ja puretaan tar-arkisto.

```
$ cd /var/www/vhost/owncloud
$ wget https://download.owncloud.org/community/
owcloud-  8.0.2.tar.bz2
$ tar -xjf owncloud-8.0.2.tar.bz2
```

Siirretään tiedostot oikeaan hakemistoon, poistetaan jäljelle jäänyt tyhjä kansio ja asennuspaketti.

```
$ mv owncloud/* .
$ rm -rf owncloud owncloud-8.0.2.tar.bz2
```

Muutetaan `/etc/apache/sites-enbled/default-ssl`-tiedoston rivi `AllowOverride None` muotoon `AllowOverride All`, joka antaa oikeuden ohittaa apachen oletusasetukset käyttäen hyväksi `.htaccess`-tiedostoa. Tämä on tarpeellista, jotta ownCloud pystyy tekemään tarvittavia muutoksia, esimerkiksi kasvattamaan palvelimelle ladattavien tiedostojen suurinta sallittua kokoa.

```
DocumentRoot /var/www/vhost/owncloud
<Directory />
Options FollowSymLinks
AllowOverride All
</Directory>
```

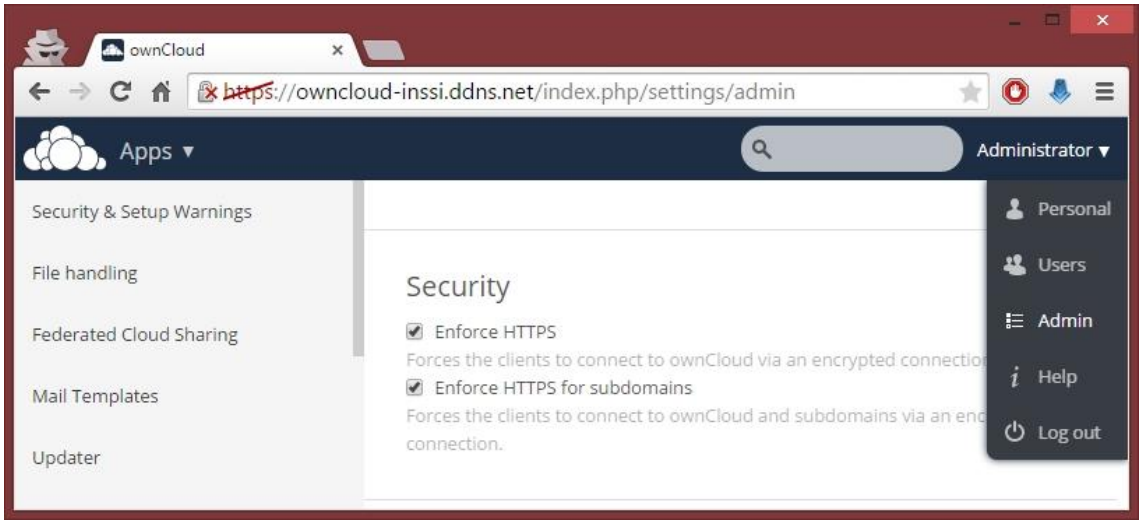
Toimiakseen ownCloud tarvitsee vielä oman tietokannan. Kirjaudutaan mysql-sovellukseen root-käyttäjätunnuksella ja luodaan uusi tietokanta komennoilla:

```
$ mysql -u root -p
mysql> create database owncloud;
Query OK, 1 row affected (0.00 sec)
```

Siirrytään selaimella osoitteeseen `https://owncloud-inssi.ddns.net`, jossa pyydetään luomaan käyttäjätunnus, antamaan käytettävän tietokannan tiedot ja määrittelemään ladattavien tiedostojen tallennushakemisto.

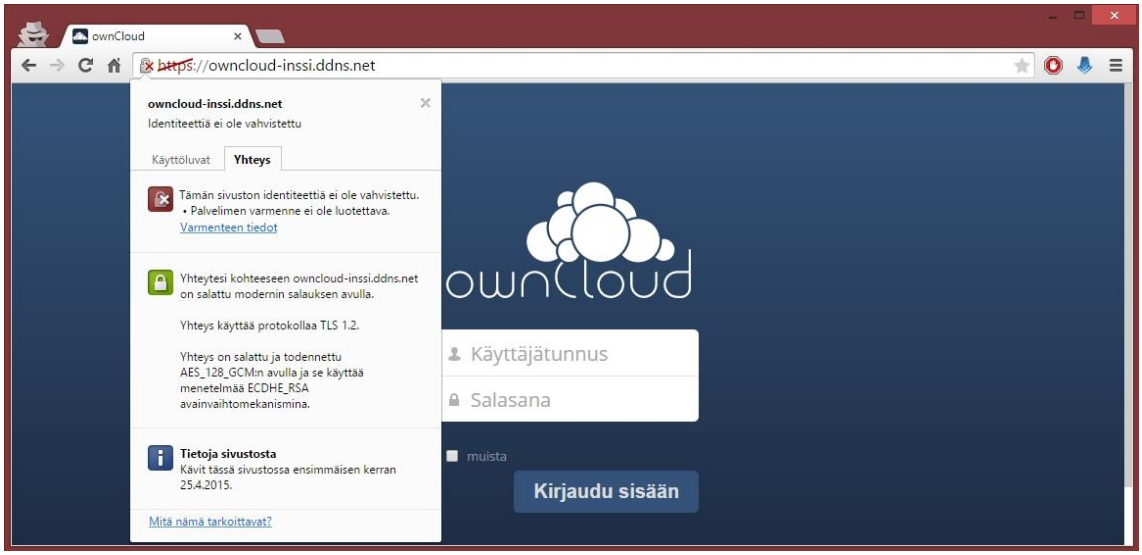
```
Username:           Administrator
Password:           <käyttäjän salasana>
Data folder:        /var/www/vhost/owncloud/data
Database user:      root
Database password:  <MySQL salasana>
Database name:      owncloud
Database host:      localhost
```

Tämän jälkeen kirjaudutaan luodulla tunnuksella sisään ja asetetaan Admin-sivulta palvelu käyttämään ainoastaan HTTPS-protokollaa.



Kuva 7. OwnCloud pakotettu HTTPS-protokollan käyttö.

Lopputuloksena saatiin luotua verkkotallennuspalvelu osoitteeseen `https://owncloud-inssi.ddns.net` suojatulla verkkoyhteydellä.



Kuva 8. OwnCloud-kirjautumissivu jossa käytössä HTTPS.

4.5 Viimeistely ja testaus

Asennusten jälkeen sivustoilla vierailtaessa saattaa esiintyä huomattavaa hidastelua, ja tästä syystä käyttömukavuus jättää vielä toivomisen varaa. Apachen palvelinohjelmistosta löytyy testiohjelma nimeltään ApacheBench (`ab`), jolla mitataan palvelimen suorituskykyä lähettämällä sille haluttu määrä pyyntöjä ja laskemalla pyyntöihin vastaamiseen käytetty aika [13]. Seuraavaksi testataan molempien sivustojen suorituskyvyt ajamalla 50 pyyntöä komennoilla:

```
$ sudo ab -n 50 https://wordpress-inssi.ddns.net/
$ sudo ab -n 50 https://owncloud-inssi.ddns.net/
```

Taulukko 2. ApacheBench tulokset.

	ownCloud	Wordpress
Complete requests	50	50
Time taken for test	146.564 seconds	196.664 seconds
Request per second	0.34 [#/sec] (mean)	0.25 [#/sec] (mean)
Time per request	2931.275 [ms] (mean)	3932.876 [ms] (mean)

Kuten taulukosta 2 voidaan nähdä, tulokset eivät ole erityisen hyvät. Tämä johtuu pääosin RaspberryPi:n rajallisista resursseista, mutta myös palveluissa käytetyn ohjelmoin-

tikielen raskaudesta. PHP on komentosarjakieli, jossa ohjelmakoodia tulkitaan vasta ohjelmaa suoritettaessa, tässä tapauksessa kun käyttäjä vierailee sivustolla. Jotta saman ohjelmakoodin tulkitsemiseen ei tarvitsisi tuhlata tarpeettomasti resursseja, voidaan ottaa käyttöön välimuisti, johon lisätään jokainen jo kertaalleen suoritettu PHP-skripti. Tämä tapahtuu varsin vaivattomasti asentamalla php-apc (The Alternative PHP Cache) -moduuli [14].

```
$ sudo apt-get install php-apc
```

Seuraavaksi otetaan moduuli käyttöön ja kasvatetaan välimuistin kokoa muokkaamalla `20-apc.ini`-tiedostoa.

```
$ sudo nano /etc/php5/conf.d/20-apc.ini
```

Muokataan tiedostoa niin, että sen sisältö näyttää seuraavalta:

```
extension = apc.so
apc.enabled = 1
apc.shm_size = 12M
```

Seuraavaksi käynnistetään apache uudelleen ja suoritetaan testit ajamalla samat komennot kuin aikaisemmin.

```
$ sudo service apache2 restart
```

Taulukko 3. ApacheBench tulokset php-apc moduulin ollessa käytössä.

	ownCloud	Wordpress
Complete requests	50	50
Time taken for test	63.625 seconds	50.938 seconds
Request per second	0.79 [#/sec] (mean)	0.98 [#/sec] (mean)
Time per request	1272.500 [ms] (mean)	1018.766 [ms] (mean)

Hyvin pienillä muutoksilla saatiin aikaan huomattava parannus, kuten taulukon 3 tuloksista voidaan nähdä. Sekunnin aikana tehtyjen pyyntöjen määrä kasvoi noin 43 prosenttia ja saman verran väheni pyynnön suorittamiseen käytetty aika.

Hyvään palveluun kuuluu tärkeänä osana varmuuskopiointi, joka on automatisoitu ja suorittaa tarvittavat toimenpiteet valittuna ajankohtana. Tässä työssä varmuuskopio otetaan koko `/var/www/`–hakemistosta, joka sisältää molempien sivustojen asentamisessa käytetyt tiedostot sekä käyttäjän lisäämät ja muokkaamat aineistot, ja tallennetaan se USB-muistitikulle. Johtuen Raspberry Pi:n rajallisista resursseista reaaliaikainen varmuuskopiointi ei tule kysymykseen, joten varmuuskopiointi tullaan suorittamaan vain kerran päivässä.

Linux sisältää juuri tähän tarkoitukseen sopivat, erittäin tehokkaat ja monipuoliset työkalut. Ensimmäiseksi alustetaan USB-väylään liitetty muisti käyttämään `ext4`–tiedostojärjestelmää.

```
$ sudo mkfs.ext4 /dev/sda1 -L untitled
```

Alustamisen jälkeen luodaan kansio, jota käytetään työn seuraavassa vaiheessa liitoskohtana tiedostojärjestelmälle.

```
$ sudo mkdir /media/backup
```

Jotta palvelin osaisi jatkossa automaattisesti liittää tiedostojärjestelmän hakemistopuuhun käynnistyksen yhteydessä, muokataan `/etc/fstab`–asetustiedostoa ja määritellään liitettävä laite sekä liitoskohde lisäämällä seuraava rivi ja tallentamalla tiedosto.

```
/dev/sda1 /media/backup ext4 defaults,noatime  0  0
```

Seuraavaksi asennetaan rsync-ohjelma, jonka avulla ylläpidetään hakemistorakenteen varmuuskopiota. Rsync kopioi oletuksena vain muuttuneet tiedostot ja isojen tiedostojen osalta vain muuttuneet osat, joten synkronointi on varsin nopeata.

```
$ sudo atp-get install rsync
```

Kun rsync-ohjelma on asennettu, täytyy vielä määrittää ajastettu komennon suorittaminen. Cron on ajastuspalvelu Unix-pohjaisille käyttöjärjestelmille, ja ajastimia muokataan crontab-ohjelmalla. Crond-taustaprosessi ajaa komennot taustalla ja tehtävä suoritetaan aina, kun aika- ja päivämääritykset täsmäävät palvelimen aikaa.

Ajastimen luominen aloitetaan käynnistämällä crontab-ohjelma root-pääkäyttäjän tunnuksella. Parametria `-e` käytetään avaamaan tämänhetkinen crontab-tiedosto helpommin käsiteltävään tekstieditoriin.

```
$ sudo crontab -e
```

Seuraavaksi lisätään tiedoston loppuun seuraava rivi ja poistutaan editorista näppäinyhdistelmällä Ctrl-X.

```
0 5 * * * rsync -av --delete /var/www/ /media/backup/
```

Tämä rivi määrittelee, että joka päivä viideltä aamuyöstä suoritetaan synkronointi `/var/www/`–hakemistosta `/media/backup/`–hakemistoon. Parametrilla `--delete` määritellään tiedosto poistettavaksi, jos on poistettu myös tiedoston alkuperäisestä sijainnista.

Lopuksi voidaan testata komennon toimivuus ajamalla se terminaalissa, ilman crontab-ohjelman aikamäärityksiä ja tarkistamalla, onko kohdehakemistoon synkronoitunut tiedostoja.

```
$ sudo rsync -av --delete /var/www/ /media/backup/

$ du -hs /media/backup/ /var/www/
353M    /media/backup/
353M    /var/www/
```

Komennon tulosteesta nähdään, että kohdehakemistoon kopioitui 353 megatavun verran tiedostoja, mikä on yhtäläinen alkuperäisen hakemiston kanssa.

5 Yhteenveto

Tässä opinnäytetyössä perehdyttiin pilvipalveluihin, niiden erilaisiin rakenteellisiin yksityiskohtiin, tietoturvaan sekä pohdittiin palveluiden tuomia etuja ja haittoja. Työ jaettiin kahteen osaan, joista jälkimmäinen sisälsi käytännön osa-alueen suunnitelmineen, jonka tehtävänä oli tutustuttaa lukija prosessiin, jonka lopputuloksena syntyi toimiva yksityiseen käyttöön tarkoitettu IaaS-mallinen pilvipalvelukokonaisuus.

Teoriaosuudessa käsiteltiin palveluiden eroavaisuudet niiden rakenteellisten erojen sekä käyttötarkoituksien mukaan. Nämä esiteltiin lähtökohtaisesti palvelun käyttäjän näkökulmasta, jotta saatiin selkeä käsitys palveluiden hankinnassa huomioon otettavista seikoista. Samalla tuotiin esille pilvipalveluiden tietoturvaan kohdistuvat paineet sekä esiteltiin luotettavan palvelun tarjoamiseen vaikuttavia tekijöitä.

Pilvipalvelun suunnitelmassa käytiin läpi käytännön osuudessa käytettävät laitteet, ohjelmistot ja muut ratkaisut kertoen samalla niiden taustoista. Suunnitelmassa huomioitiin laitteiston rajalliset resurssit ja pyrittiin pitämään tavoitteet realistisina.

Varsinaisen pilvipalvelun käyttöönoton vaiheet käytiin läpi kronologisessa järjestyksessä. Tämä alue työstä piti sisällään yksityiskohtaisesti selitetyt vaiheet asennuksista aina palveluiden konfigurointiin ja käyttöönottoon asti. Erityisesti tarkastelussa oli palveluiden toiminnan kannalta Apachen palvelinohjelman konfigurointi.

Lähteet

1 National Institute of Standards and Techonology (NIST). The NIST Definition of Cloud Computing. Verkkodokumentti. Saatavissa: http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf. Luettu 20.02.2015.

2 The Growing Popularity of Cloud Computing. Verkkodokumentti. Saatavissa: http://www.techgyd.com/the-growing-popularity-of-cloud-computing-and-digital-marketing/14913/. Luettu 15.02.2015.

3 Software, Platform, Infrastructure Model (SPI Model). Verkkodokumentti. Saatavissa: http://www.techopedia.com/definition/14019/software-platform- infrastructure-model-spi-model. Luettu 01.03.2015.

4 Cloud Definiton Framework, NIST. Verkkodokumentti. Saatavissa: http://4.bp.blogspot.com/-Dw2nunjpHR0/UtFGeIsCQYI/AAAAAAAAAD4/gfZNqjjLOUk/s1600/Cloud+definition+framework+.PNG. Luettu 03.03.2015.

5 Amazon Web Services. Verkkodokumentti. Saatavissa: http://aws.amazon.com/products/. Luettu: 04.03.2015.

6 Google Cloud Platform. Verkkodokumentti. Saatavissa: https://cloud.google.com/products/. Luettu 12.03.2015.

7 Pilvipalvelumallit. Verkkodokumentti. Saatavissa: http://fi.laovirtualisointi.wikia.com/wiki/Pilvipalvelumallit. Luettu 08.03.2015.

8 ownCloud. Verkkodokumentti. Saatavissa: https://owncloud.org/features/. Luettu 01.04.2015.

9 WordPress. Verkkodokumentti. Saatavissa: https://wordpress.org/about/. Luettu 02.04.2015.

10 TLS Standardi RFC 5246. Verkkodokumentti. Saatavissa: http://tools.ietf.org/html/rfc5246. Luettu 04.04.2015.

11 Apache HTTP Server. Verkkodokumentti. Saatavissa: http://httpd.apache.org/. Luettu 05.04.2015.

12 No-IP:Free Dynamic DNS. Verkkodokumentti. Saatavissa: http://www.noip.com/. Luettu 10.04.2015.

13 ApacheBench. Verkkodokumentti. Saatavissa: http://httpd.apache.org/docs/2.2/programs/ab.html. Luettu 30.04.2015.

14 Alternative PHP Cache. Saatavissa: http://php.net/manual/en/book.apc.php. Luettu 01.05.2015.

Apache Name-based Virtual Hosts

```
<IfModule mod_ssl.c>
NameVirtualHost *:443

#============== Suodatin ==============

 <VirtualHost *:443>
       ServerName default.only
        <Location />
                Order allow,deny
                Deny from all
        </Location>

        ErrorLog ${APACHE_LOG_DIR}/spam/error.log
        LogLevel warn
        CustomLog ${APACHE_LOG_DIR}/spam/ssl_access.log combined

        SSLEngine on
        SSLCertificateFile              /etc/ssl/certs/ssl-cert-
          snakeoil.pem
        SSLCertificateKeyFile         /etc/ssl/private/ssl-cert-
          snakeoil.key
</VirtualHost>

#============== Wordpress ==============

<VirtualHost *:443>
        ServerAdmin webmaster@localhost
                    ServerName www.wordpress-inssi.ddns.net
                    ServerAlias wordpress-inssi.ddns.net
        DocumentRoot /var/www/vhost/wordpress
        <Directory />
                Options FollowSymLinks
```

```
                AllowOverride None
        </Directory>
        <Directory /var/www/vhost/wordpress>
                Options Indexes FollowSymLinks MultiViews
                AllowOverride All
                Order allow,deny
                allow from all
        </Directory>

        ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
        <Directory "/usr/lib/cgi-bin">
                AllowOverride None
                Options +ExecCGI -MultiViews                +Sym-
LinksIfOwnerMatch
                Order allow,deny
                Allow from all
        </Directory>

        ErrorLog ${APACHE_LOG_DIR}/wordpress/error.log
        CustomLog     ${APACHE_LOG_DIR}/wordpress/ssl_access.log
          combined

        SSLEngine on
        SSLCertificateFile           /etc/apache2/ssl/wordpress-
          inssi.ddns.net/apache.crt
        SSLCertificateKeyFile        /etc/apache2/ssl/wordpress-
          inssi.ddns.net/apache.key

        <FilesMatch "\.(cgi|shtml|phtml|php)$">
                SSLOptions +StdEnvVars
        </FilesMatch>
        <Directory /usr/lib/cgi-bin>
                SSLOptions +StdEnvVars
        </Directory>
```

```
        BrowserMatch "MSIE [2-6]" \
                nokeepalive ssl-unclean-shutdown \
                downgrade-1.0 force-response-1.0
        # MSIE 7 and newer should be able to use keepalive
        BrowserMatch "MSIE [17-9]" ssl-unclean-shutdown

</VirtualHost>

#============ ownCloud ==================

<VirtualHost *:443>
        ServerAdmin webmaster@localhost
                    ServerName www.owncloud-inssi.ddns.net
                    ServerAlias owncloud-inssi.ddns.net

        DocumentRoot /var/www/vhost/owncloud
        <Directory />
                Options FollowSymLinks
                AllowOverride All
        </Directory>
        <Directory /var/www/vhost/owncloud>
                Options Indexes FollowSymLinks MultiViews
                AllowOverride All
                Order allow,deny
                allow from all
        </Directory>

        ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
        <Directory "/usr/lib/cgi-bin">
                AllowOverride None
                Options  +ExecCGI  -MultiViews  +SymLinksIfOwn-
erMatch
                Order allow,deny
                Allow from all
        </Directory>
```

```
        ErrorLog ${APACHE_LOG_DIR}/owncloud/error.log
        CustomLog       ${APACHE_LOG_DIR}/owncloud/ssl_access.log
          combined

        SSLEngine on
        SSLCertificateFile              /etc/apache2/ssl/owncloud-
          inssi.ddns.net/apache.crt
        SSLCertificateKeyFile           /etc/apache2/ssl/owncloud-
          inssi.ddns.net/apache.key

        <FilesMatch "\.(cgi|shtml|phtml|php)$">
                SSLOptions +StdEnvVars
        </FilesMatch>
        <Directory /usr/lib/cgi-bin>
                SSLOptions +StdEnvVars
        </Directory>

        BrowserMatch "MSIE [2-6]" \
                nokeepalive ssl-unclean-shutdown \
                downgrade-1.0 force-response-1.0
        # MSIE 7 and newer should be able to use keepalive
        BrowserMatch "MSIE [17-9]" ssl-unclean-shutdown

</VirtualHost>
</IfModule
```