Tampereen ammattikorkeakoulu

Kotilaboratorion virtualisointi

Docker-konttitekniikalla

Matti Hyttinen

OPINNÄYTETYÖ
Toukokuu 2019

Tietotekniikka
Tietoliikennetekniikka ja tietoverkot

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tietotekniikka
Tietoliikennetekniikka ja tietoverkot

HYTTINEN, MATTI:
Kotilaboratorion virtualisointi
Docker-konttitekniikalla

Opinnäytetyö 48 sivua, joista liitteitä 12 sivua
Toukokuu 2019

Opinnäytetyön tarkoituksena oli rakentaa kotilaboratorio virtualisointitekniikalla. Alkuperäinen virtuaalikoneisiin perustuva ympäristö vaihdetaan sovelluskontteihin. Sovelluskonttien virtualisoinnissa käytetään Docker-virtualisointialustaa ja sen ympärille rakennetaan soveltuva paikallinen tietoverkko.

Tavoitteisiin sisältyi palvelimen asentaminen ja tietoverkon uudelleen rakentaminen. Palvelimelta julkiseen ulkoverkkoon näkyvät yhteydet suojataan ja paikalliseen verkkoon tarjotaan lisäpalveluita. Sovelluskontteihin tutustutaan rakentamalla avoimeen lähdekoodiin perustuvasta, mediatiedostojen suoratoistoon käytettävästä sovelluksesta päivitetty versio. Sovelluskonttien nopeaan testaamiseen viitataan rinnalle tuotavalla tietokanta ohjelmistolla, hyödyntäen julkisesti saatavilla olevia levykuvia.

Palvelimelle asennetut palvelujen hallintaa helpottavat sovellukset on testattu ja todettu toimiviksi. Verkon suunnittelun myötä paikallista lähiverkkoa voidaan laajentaa tulevaisuuden tarpeiden mukaan. Sovelluskontit tarvitsevat vielä toimenpiteitä toimiakseen halutulla tavalla, siitä huolimatta pohjalla oleva virtualisointialusta vaikuttaa käytännölliseltä.

Tulevaisuuden tavoitteisiin kuuluu paikallisten lähiverkkojen tarkempi rajaaminen ja uusien sovelluskonttien lisääminen palvelimelle. Suojattujen yhteyksien tarvitsemat sertifikaatit määritellään päivittymään automaattisesti ja mahdollisesti otetaan käyttöön VPN-tunnelointi.

Asiasanat: docker virtualisointi palvelin tietoverkko

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Information technology
Information technology and networks

HYTTINEN, MATTI:
Home lab virtualization
with Docker containers

Bachelor's thesis 48 pages, appendices 12 pages
May 2019

The purpose of this thesis was to create home laboratory by utilizing virtualization technology. Original environment will be changed from virtual machine-based technology to containerized applications. Docker container technology is utilized while creating the surrounding local network and base platform for containers.

Objectives included building the server platform and implementing a new logical network layout for local use. Connections from the server to the public network are meant to be secure and new services will be served to the local network. Containers and their utilization are introduced via streaming based application which will be updated and containerized. Flexibility of containers for testing will be referred to by introducing database features from publicly available container images.

Software for managing different services based on server have been tested and found to be in working order. Newly designed network layout meets future requirements for expansion. Running containers requires action to work in expected way but the underlying platform for them seems to be useful.

For future, local networks should be restricted further into their own zones. Different kind of new services will be added to the server with containers. Renewal of required certifications for secure connections will be automated and possible utilization VPN tunnels will be considered.

Key words: docker virtualization server network

SISÄLLYS

LYHENTEET JA TERMIT

ACL	Access List, pääsylista
ACME	Automated Certificate Management Environment, Automaattinen sertifikaattien hallinnointi
Apache	Apache HTTP Server, Avoimeen lähdekoodin perustuva HTTP-palvelinohjelmisto
BIND	Avoimeen lähdekoodiin perustuva nimipalvelinohjelmisto
cURL	Client URL, curl-ohjelmisto. Käytetään datan siirtämiseen URL-osoitteiden perusteella
DEB	Debian tiedostoformaatti
DHCP	Dynamic Host Configuration Protocol, verkkoprotokolla IP-osoitteiden jakamiseen dynaamisesti
DHE	Diffie-Hellman, avaintenvaihtoprotokolla
DNS	Domain Name System, nimipalvelujärjestelmä IP-osoitteiden muuntamiseen verkkotunnuksiksi
Docker	Sovelluskonttien luomiseen ja hallinnointiin käytettävä virtualisointialusta
Domain	Englannin kielinen termi verkkotunnukselle
EFF	Electronic Frontier Foundation, kansainvälinen tietoyhteiskunnan kansalaisoikeuksia puolustava järjestö
GPG	Gnu Privacy Guard, PGP-salausohjelmisto
GRUB	Grand Unified Bootloader, käyttöjärjestelmien käynnistyksen hallinta ohjelmisto
Hostname	Isäntänimi
HTTP	Hypertext Transfer Protocol, sovellustason protokolla tiedostojen siirtämiseen tietoverkossa
HTTPS	Hypertext Transfer Protocol Secure, suojattu sovellustason protokolla tiedostojen siirtoon tietoverkossa
Hypervisor	Virtuaalikoneiden hallinnoija
ICANN	Internet Corporation for Assigned Names and Numbers, voittoa tavoittelematon organisaatio, joka hallinnoi internetin osoitteiden tunnuksia

JSON	Javascript Object Notation, yksinkertainen tiedonvälitykseen käytettävä avoimen standardin tiedostoformaatti
Kernel	Käyttöjärjestelmän ydin
LTS	Long Term Support, pitkäaikainen tuki (käyttöjärjestelmät)
LXC, LXD	Linux Containers, Linux-käyttöjärjestelmä kontit
LVM	Logical Volume Management, Looginen tallennustilan allokointi
NAS	Network-attached storage, verkkotallennusjärjestelmä
NFS	Network File System, verkkolevyjärjestelmän avulla hakemistoja ja tiedostoja voidaan käyttää tietoverkon ylitse
OMV	Openmediavault, verkkolevypalvelimen käyttöjärjestelmä
PPA	Personal Package Archive, henkilökohtainen paketti arkisto
Reverse proxy	Englanninkielinen termi käänteiselle välityspalvelimelle
Rolling release	Ohjelmistojen jatkuva päivitysmalli
RSA	Rivest-Shamir-Adleman, julkisen avaimen salausalgoritmi
TLS	Transport Layer Security, salausprotokolla tietoliikenteen suojaamiseen (aiemmin tunnettu lyhenteellä SSL)
TCP	Transmission Control Protocol, yhteydellinen tietoliikenne protokolla. Huolehtii IP-pakettien perille pääsemisestä oikeassa järjestyksessä ja varmistaa niiden eheyden
UDP	User Datagram Protocol, yhteydetön tietoliikenneprotokolla. Laitteiden välinen yhteys ei ole välttämätön ja IP-pakettien perille pääsyä ei varmisteta
VHOST	VirtualHost, mahdollistaa useamman verkkosivun julkaisun samalla Apache-palvelinohjelmalla
VLAN	Virtual Local Area Network, virtuaalilähiverkko
VM	Virtual Machine, virtuaalikone
WLAN	Wireless Local Area Network, langaton lähiverkko

1 JOHDANTO

Kotilaboratorion virtualisoinnissa rakennetaan ympäristö sovelluskonttien testaamiseen ja palveluiden tuottamiseen tietoverkkoon. Työssä alustetaan pohja sovelluskontteihin perustuvaan virtualisointiin. Virtualisoinnin avulla voidaan hyödyntää käytettävissä olevia palvelimen resursseja monipuolisemmin ja virtualisointitekniikoita on olemassa erilaisia. Työssä otetaan tarkempaan tarkasteluun sovelluskonttien luomiseen ja hallintaan käytettävä Docker-virtualisointialusta.

Opinnäytetyössä tutustutaan sovelluskonteissa käytettävään tekniikkaan ja rakennetaan sen ympärille toimiva kokonaisuus testaamista ja kehittämistä varten. Ympäristön rakennukseen sisältyy tuntemusta useammasta tietotekniikan osa-alueesta tarvittavien perusosien rakentamiseen. Palvelimesta, tietoverkosta ja sovelluskonteista kerrotaan tapahtumien kulku ja perustellaan tehtyjä valintoja.

2 TEORIA

2.1 Virtualisointi

Virtualisoinnissa luodaan asioita näennäisesti, joka käyttäytyy luonnollisesti keinotekoisuudestaan huolimatta. Tietotekniikassa virtualisoinnilla voidaan fyysisiä resursseja hallinnoida ja käyttää useampina loogisina osina. (Hiltunen, J. 2010, 3.)

Loogisia resursseja käytettäessä taustalla olevia fyysisiä resursseja ei paljasteta suoraan virtualisoiduille alustoille. Looginen resurssien hallinnointi mahdollistaa fyysisten resurssien jakamisen pienempiin osiin tai päinvastoin, useamman fyysisen resurssin yhdistämisen ja näyttämisen yhtenä loogisena. (IBM, 2012.)

Esimerkkinä voidaan käyttää useaa rinnakkaista virtuaalipalvelinta, jotka sijaitsevat yhdellä fyysisellä palvelimella. Fyysisen palvelimen resurssit muutetaan loogisiksi osiksi ja jaetaan virtuaalipalvelimille. Loogiset resurssit näkyvät virtuaalipalvelimille käytettävissä olevina laitteina, vaikka todellisuudessa ne ovat emuloituja.

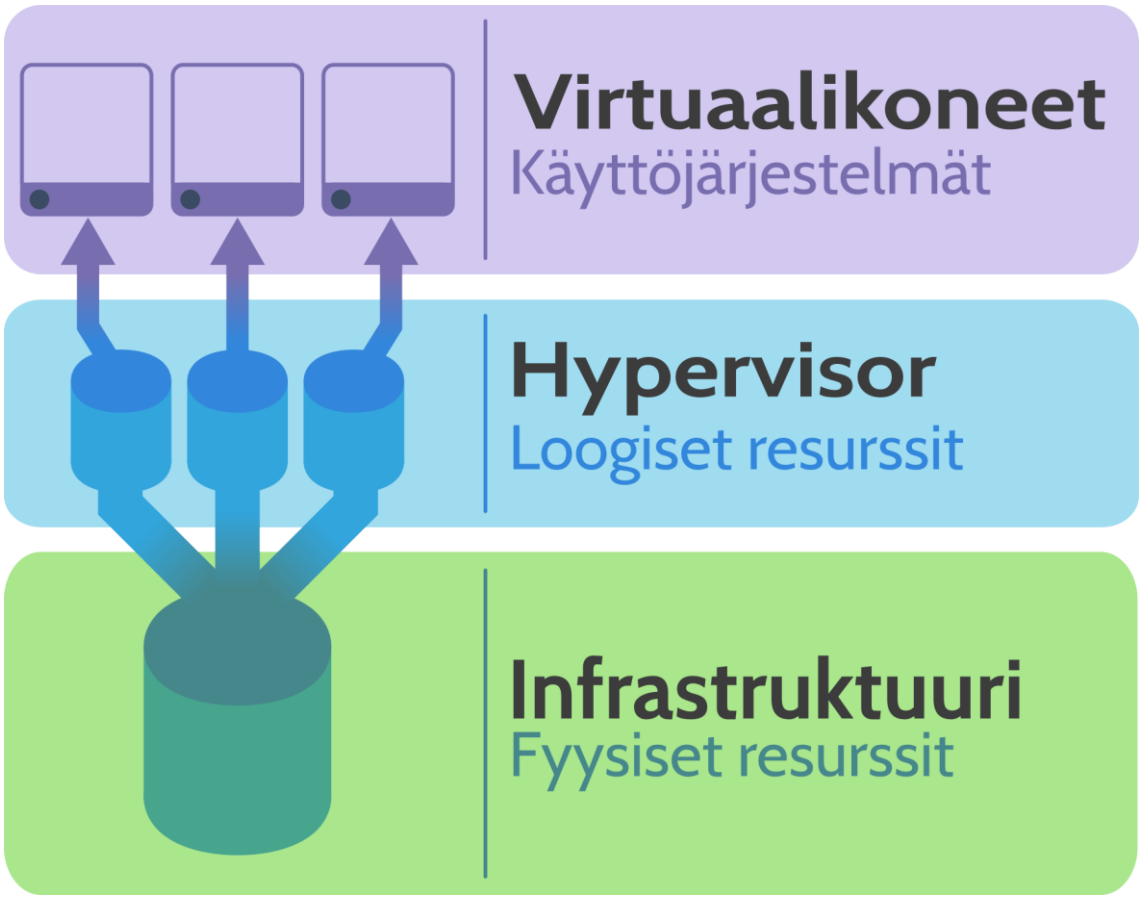
2.2 Virtuaalikoneet (VM)

Virtuaalikoneet emuloivat tietokonejärjestelmien arkkitehtuuria ja mahdollistaa fyysisen tietokoneen toiminnallisuuden loogisesti jaetuista resursseista. Virtuaalikoneiden käyttäminen saattaa sisältää virtualisointiin erikoistuneita ominaisuuksia laitteiston ja ohjelmistojen kannalta, tai olla yhdistelmä molempia. (Campbell S. & Jeronimo M. 2006.)

Hypervisor hallinnoi ja luo virtuaalikoneita. Hypervisorin tehtäviin kuuluu resurssien allokointi loogisiksi, sekä virtuaalisen alustan tarjoaminen virtuaalikoneelle. Virtuaalikoneet ovat eriytettyinä toisistaan, jolloin ne ovat tietoisia ainoastaan

hypervisorin tarjoamista resursseista. Virtuaalinen alusta sisältää emuloitavan laitteiston ns. vieraskoneelle. (Campbell S. & Jeronimo M. 2006.)

Tietokonetta, jossa hypervisor hallinnoi yhtä tai useampaa virtuaalikonetta kutsutaan yleensä isäntäkoneeksi (engl. host machine). Isäntäkoneen alaisuudessa on vieraskoneet (engl. guest machine) (Hoffman, C. 2017). Alla on esiteltynä havainnekuva virtuaalikoneiden rakenteesta.



KUVA 1. Virtuaalikoneiden virtualisointi rakenne (Docker 2019a, kuva perustuu lähteeseen)

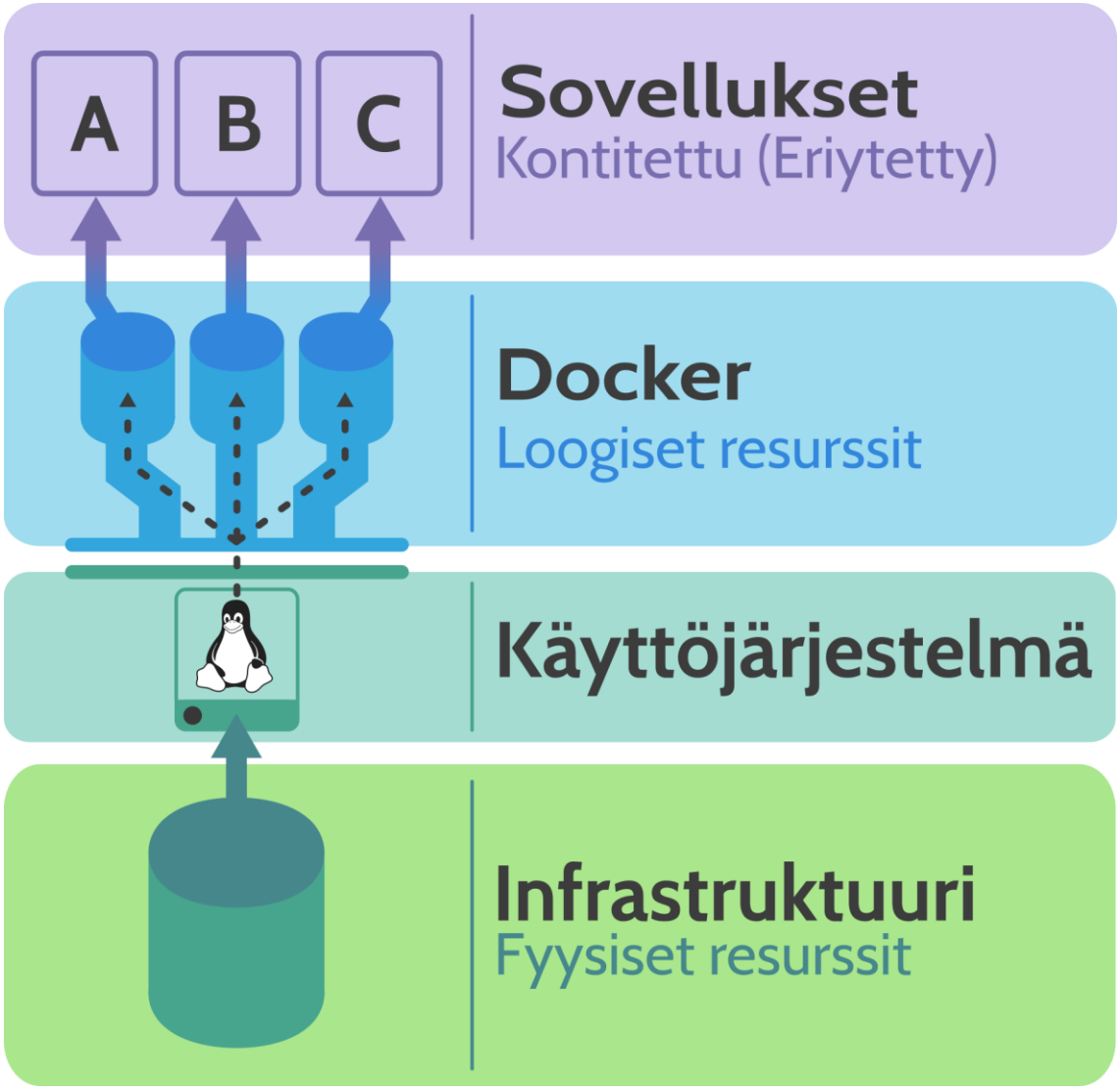
Vaikka hypervisor mahdollistaa useamman virtuaalikoneen ajamisen rajoitettuja fyysisiä resursseja hyödyntäen, niiden lopullinen suorituskyky määräytyy loogisen allokoinnin ja käytettävissä olevien laitteisto-ominaisuuksien mukaan.

Virtuaalikoneiden heikkouksiin voidaan lukea esimerkiksi niiden vaatima fyysinen tallennustila. Virtuaalikoneiden toimintaan tarvitaan alkuperäisestä käyttöjärjestelmästä kokonainen kopio, jonka päällä voidaan hyödyntää tarvittavia sovelluksia. Kokonaisen käyttöjärjestelmän ja sovellusten käynnistäminen voi olla myös hitaampaa verrattuna esimerkiksi sovellusten konttitekniikkaan. (Docker. 2019a.)

2.3 Sovellusten konttitekniikka (Docker)

Sovellusten konttitekniikassa poistetaan välistä erillinen hypervisorin virtualisointi kerros. Kontituksella virtuaaliset sovellukset hyödyntävät suoraan isäntäkoneen käyttöjärjestelmän resursseja. Hyödynnettäessä suoraan käyttöjärjestelmän käytössä olevia fyysisiä resursseja, konttien loogisten resurssien allokoinnissa ja hyödyntämisessä ei tarvitse emuloida käytettävää laitteistoa. (Badola V. 2015.)

Sovelluskontit itsessään virtualisoivat käyttöjärjestelmäänsä isäntäkoneen kanssa jaettujen fyysisten resurssien ja käyttöjärjestelmän ytimen (engl. kernel) myötä. Konteille annetaan vähimmäismäärä resursseja sovellusten suorittamiseen isäntäkoneen käyttöjärjestelmän päällä ja ne rajoitetaan loogisesti myös omiin ympäristöihin. Kontteja voidaan ajatella tavallaan riisuttuna muotona virtuaalikoneesta. Seuraavassa havainnekuvassa on esiteltynä sovelluskonttien virtualisointi rakenne. (Badola V. 2015.)



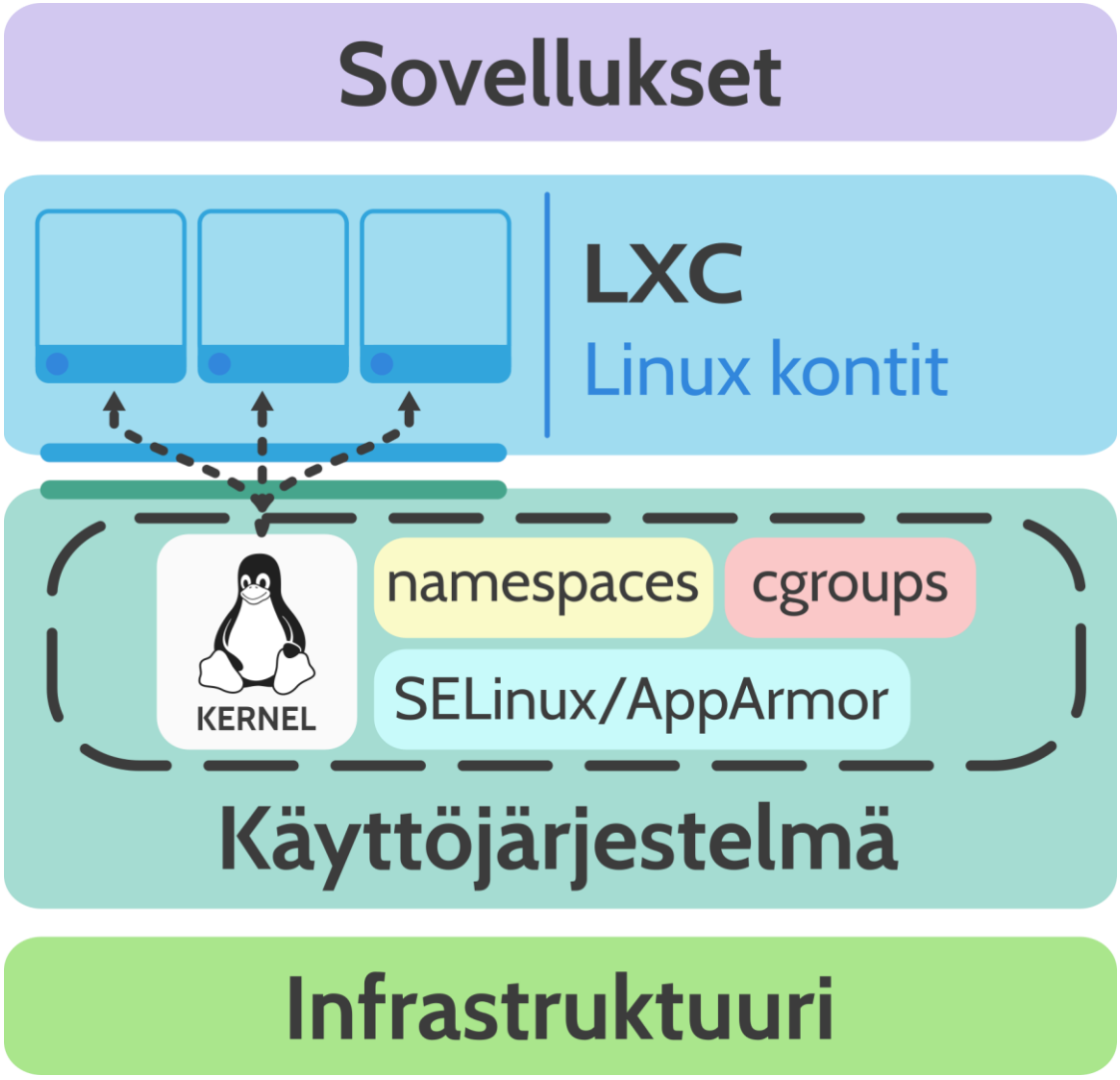
KUVA 2. Sovelluskonttien virtualisointi rakenne (Docker 2019a, kuva perustuu lähteeseen)

Käyttöjärjestelmä ei ole suoraan tietoinen olemassa olevista sovelluskonteista, sillä ne on erotettu loogisesti rajapinnassa (Badola V. 2015). Tekniikka mahdollistaa paremman suorituskyvyn verrattaessa hypervisorin päällä ajettaviin virtuaalikoneisiin, sillä käytettäviä resursseja ei tarvitse emuloida. Sovelluskontit käynnistyvät yleensä nopeammin ja ne mahdollistavat paremman kokonais- resurssien hallinnan.

2.4 Linux-käyttöjärjestelmä kontit (LXC/LXD)

Linux-käyttöjärjestelmän kontituksella voidaan hyödyntää virtuaalikoneen ominaisuuksia pienemmällä fyysisten resurssien jalanjäljellä. Siinä missä Docker keskittyy sovelluskontteihin, LXD-tekniikalla muodostetaan kontti kokonaisesta käyttöjärjestelmästä ja sitä voidaan hyödyntää normaaliin tapaan. LXD pohjautuu olemassa olevaan LXC:n tekniikkaan, mutta se ei ole suoraan uudelleenkirjoitettu versio siitä. LXD tarjoaa vaihtoehtoisen työkalun tekniikan hyödyntämiseen. (LXD Introduction, 2019.)

Käyttöjärjestelmä kontit tarjoavat täyden funktionaalisen Linux-ympäristön, jonka päälle voidaan asentaa esimerkiksi useampia sovelluksia. LXC hyödyntää Dockerin tapaan isäntäkoneen käyttöjärjestelmän ytimen ominaisuuksia konteissa, jolloin fyysiset resurssit voidaan jakaa loogisiksi. Kontit ovat eriytetty toisistaan loogisella tasolla kuten sovelluskontit. (Adeesh, F. 2017.)



KUVA 3. Linux-käyttöjärjestelmä konttien virtualisointi rakenne (Adeesh, F. 2017, kuva perustuu lähteeseen)

Yllä olevassa kuvassa on havainnoitu Linux-käyttöjärjestelmän kontituksen rakenne. Käyttöjärjestelmän ytimen hyödyntämät ominaisuudet ovat mainittuna erikseen. LXC-virtualisoinnin rakenne on yksinkertaistettuna hyvin lähellä virtuaalikoneiden tekniikkaa.

3 TAVOITTEET

3.1 Virtuaalikoneista siirtyminen sovelluskontteihin

Tavoitteisiin kuuluu kotilaboratorio tyylisen virtualisointiympäristön rakentaminen ja samalla sovelluskonttien hyödyntäminen. Sovelluskonttien luomiseen käytetään Docker-konttitekniikkaa. Alkuperäisen kotilaboratorion ympäristö koostui palvelimelle asennetusta Xen-virtualisointialustasta ja useammasta virtuaalikoneesta. Aikaisempi järjestelmä korvataan toisella tekniikalla.

Palvelimen rakennusta varten on käytettävissä vanhemman puoleinen tietokone, josta löytyy AMD Phenom II X4 810 -prosessori, neljä gigatavua keskusmuistia ja kovalevyn tallennuskapasiteettiä 320 gigatavua. Palvelimen käyttämät tiedostot tallennetaan pääasiassa lähiverkossa sijaitsevalle tiedostopalvelimelle (NAS).

Alkuperäisessä ympäristössä käytettiin kolmea Linux-virtuaalikonetta, joiden välillä jaettiin palvelimen resurssit. Kahdella yleiskäyttöisellä virtuaalikoneella oli käytettävissä yksi prosessori ydin ja gigatavu keskusmuistia. Viimeisellä median muunnokseen ja suoralähetykseen perustuvalla virtuaalikoneella oli käytettävissä kaksi ydintä ja kaksi gigatavua keskusmuistia.

Palvelimella on kokonaisresursseja käytettävissä rajallisesti ja vaihtamalla sovelluskontteihin saadaan sovelluksille tehokkaampi resurssienhallinta. Sovelluskontit saavat ainoastaan tarvitsemansa minimiresurssit ja virtuaalikoneiden virtualisoinnista aiheutuvat latenssit pienenevät. Virtuaalikone joutuu käymään lävitse käynnistysprosessin verrattuna sovelluskonttiin, joka käynnistyy suoraan omassa eristetyssä ympäristössään (Intel. 2015, 11). Alustana käytettävä palvelin on vanhemman mallinen, jolloin laitteistokomponenttien virtualisointi ominaisuuksia on vähemmän. Modernien prosessorien tukemat virtualisointi ominaisuudet parantaisivat suorituskykyä, virtuaalikoneiden päästessä lähemmäksi ns. rauta tasoa.

3.2 Docker-sovelluskontit

Käytettävät sovellukset kontitetaan omaan eristettyyn tilaan käyttäen hyödyksi Dockerin tarjoamia työkaluja. Työkaluista on saatavilla versiot graafisella käyttöympäristöllä suurimmille työpöytäkäyttöön suunnatuille käyttöjärjestelmille (Windows, macOS ja Linux). Palvelinympäristössä suurin osa konttien parissa tehdystä työstä perustuu komentorivin työkaluihin.

Sovellusten kontitukseen tarvittavat työkalut, levykuvien luomisohjeet ja dokumentaatio löytyy suoraan Dockerin kotisivuilta (Docker. 2019b, 2019c). Työkaluista asennetaan komentorivipohjaiset versiot Ubuntu-käyttöjärjestelmälle, koska palvelinta hallinnoidaan etäyhteydellä. Alustava työkalujen asentaminen palvelimelle käsitellään tarkemmin järjestelmän asennuksen ja sovelluskonttien luomisen aikana.

Sovelluskontteja voidaan käyttää myös suoraan valmiiksi julkaistujen levykuvien perusteella. Suosituimmista sovelluksista on saatavilla yleensä yhteisön tai alkuperäisen ohjelmiston tekijän luoma levykuva versio ja niitä voi selata suoraan Docker Hub -palvelusta (Docker, 2019d).

Sovelluskonttien luodut levykuvat käännetään ja asetetaan paikallis- tai etärekisteriin. Paikallisessa rekisterissä levykuvat sijaitsevat itse palvelimella. Etärekisteriä hyödynnettäessä käytetään Docker Hub -palvelua, jolloin levykuvat ovat saatavilla Internet-yhteyden välityksellä. Paikallista rekisteriä käytettäessä levykuvaa ei tarvitse lähettää aina uudestaan ja se soveltuu hyvin esimerkiksi testaamiseen. Etärekisteriä hyödynnetään usein valmiiksi testatun sovelluskontin jakamisessa yhteisölle.

4 YMPÄRISTÖN RAKENNUS

4.1 Käyttöjärjestelmän valinta

Käyttöjärjestelmäksi on valittu Ubuntu Server 18.10 -versio, joka on viimeisin saatavilla oleva normaali versio palvelimen asennus hetkellä. Saatavilla on myös pitkäaikaisen tuen piirissä oleva versio 18.04 LTS (Long Term Support), joka on vakaampi ja saa pääasiassa tietoturvaan liittyviä päivityksiä. Pitkäaikaisen tuen myötä voi myös hakea teknistä tukea suoraan Canonical-tukijajärjestöltä maksullisella sopimuksella. Normaali versioita tuetaan yhdeksän kuukauden ajan ja pitkäaikaisia versioita viisi vuotta julkaisupäivästä lähtien.

Kotilaboratorio ympäristössä ei välttämättä tarvita mahdollisimman hyvää vakautta ja siksi valintana on normaali versio. Normaali version myötä saapuvat uusimmat ominaisuudet ja käyttöjärjestelmän ytimet. Normaali versiossa ei vielä käytetä jatkuvien päivityksien tyylistä julkaisutapaa (engl. rolling release), jossa viimeisimmät alustalle tehdyt ohjelmistokäännökset julkaistaan loppukäyttäjille. Jatkuvia päivityksiä hyödyntävät Linux-käyttöjärjestelmä variantit julkaisevat yleensä myös virkistetyn version asennuslevykuvasta, joka sisältää julkaisuun mennessä tehdyt uusimmat ohjelmistokäännökset. Silloin viimeisimmän version kiinni ottamiseen ei tarvitse tehdä suurta päivitystä esimerkiksi uudelleen asennuksen myötä.

Linux-käyttöjärjestelmien valinnan tarjous on kirjava ja muita hyviä vaihtoehtoja pieneen testaus ympäristöön ovat esimerkiksi CentOS-, Fedora- tai Debian-jakeluversiot. CentOS- ja Debian-jakeluversiot perustuvat vakauteen ja eivät tarjoa suoraan ohjelmien viimeisimpiä ominaisuuksia ja versioita. Fedoran palvelimille suunnattu jakeluversio on lähempänä Ubuntun normaalia versiota, jossa tukiaika on lyhyempi ja uudemmat ominaisuudet tulevat nopeammin jakeluun.

4.2 Käyttöjärjestelmän asennus

Minimi levykuva versiolla asennetaan Ubuntu-käyttöjärjestelmä palvelimelle, joka kattaa minimivaatimukset jatko asennusta varten (Ubuntu Community, 2018). Minimalistisella lähestymistavalla palvelimen lähtökohdassa vaaditut resurssit ovat mahdollisimman pienet ja esimerkiksi työpöytä käyttöön soveltuvaa graafista käyttöympäristöä ei asenneta ollenkaan. Asennuksessa palvelimen kaikki fyysiset resurssit annetaan käyttöjärjestelmän käytettäväksi. Asennuksessa kannattaa varsinkin kiintolevyjen allokoinnissa kiinnittää huomiota asennustapaan ja valita oikea fyysinen tallennusmedia, mikäli käytettävissä on useita.

Asennuksen aikana valitaan käyttöjärjestelmän kieli, sijainti ja näppäimistön asettelu. Isäntänimen (engl. hostname) kohdalla palvelin kannattaa nimetä helposti tunnistettavaksi. Isäntänimi yksilöi palvelimen, sekä sen näkyvyyden tietoverkkoon. Isäntänimen asettamisen jälkeen valitaan peilipalvelin, jolla sijaitsee Ubuntun arkisto asennettavista paketeista. Peilipalvelin kannattaa valita asennettavan palvelimen sijainnin perusteella omasta maasta tai mahdollisimman läheltä. Peilipalvelimen sijainti vaikuttaa pakettien latausnopeuteen, pidentäen asennukseen ja päivityksiin käytettävää aikaa. HTTP Proxyn asetukset voidaan jättää oletuksille (tyhjä), mikäli käytössä ei ole välityspalvelinta.

Käyttäjätilien luomisen aikana tehdään alustavasti normaali käyttäjä. Normaali käyttäjä on oikeuksiltaan erillään pääkäyttäjästä (root). Luonnin aikana pyydetään käyttäjän nimi, käyttäjätunnus ja uniikki salasana. Luotu käyttäjä asetetaan “sudoers” ryhmään ja ryhmän jäsenet voivat tehdä järjestelmänhallintaan liittyviä toimintoja sudo-komennon kautta.

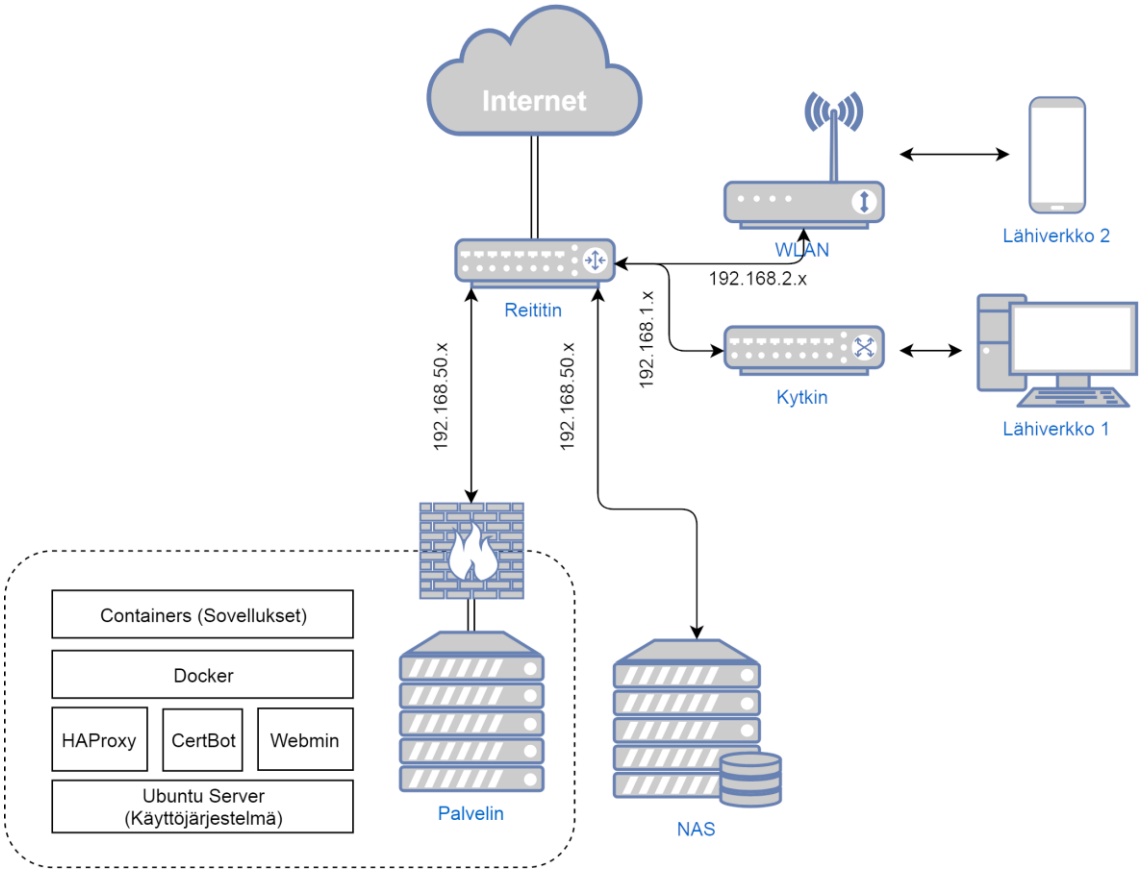
Palvelimella on käytettävissä yksittäinen tyhjä kovalevy. Levytilan allokoinnissa käytetään ohjattua menetelmää ja halutessaan sen voi myös tehdä manuaalisesti. Allokoinnissa käytetään koko levytilan allokointia ja asetetaan LVM (Logical Volume Management). LVM näyttää yksinkertaistetun loogisen näkymän fyysisestä tallennustilasta ja sitä voidaan esimerkiksi laajentaa sisältämään useampia fyysisiä kovalevyjä (Ubuntu Community, 2014).

Käyttöjärjestelmän päivitysten asennukseen voidaan vaikuttaa kolmella erilaisella menettely tavalla. Automaattiset päivitykset voidaan kytkeä kokonaan pois päältä, jolloin järjestelmänvalvojan tarvitsee manuaalisesti suorittaa päivitykset. Toisena mahdollisuutena on tietoturvapäivitysten automaattinen asentaminen ilman käyttäjän toimia, jolloin käyttöjärjestelmän tietoturva pysyy ajan tasalla. Tietoturvapäivitysten ulkopuolella olevat toimet, esimerkiksi ohjelmaversion päivitykset tarvitsevat järjestelmävalvojan suostumuksen. Viimeisenä vaihtoehtona voidaan valita Canonical:in tarjoama Landscape-palvelu. Palvelun avulla voidaan ohjata useampia järjestelmiä verkon ylitse. Pienessä käyttöympäristössä on suositeltavaa jättää automaattiset tietoturvapäivitykset päälle.

Viimeisenä asennusvaiheena valitaan erilliset asennettavat sovellukset. Minimi asennuksessa voidaan valita OpenSSH Server -paketti. Paketin asentaminen mahdollistaa SSH-etäyhteyksien käyttämisen. Ubuntussa käytetään oletuksena käynnistyksen hallintaan GRUB-ohjelmaa ja sen asentaminen on suositeltavaa. Uudelleenkäynnistyksen jälkeen käyttöjärjestelmä on valmis käytettäväksi.

4.3 Tietoverkon looginen rakenne

Tietoverkon fyysinen kokonaisrakenne on yksinkertainen ja koostuu muutamasta verkkolaitteesta. Alla olevassa kuvassa on havainnollistettu lähiverkkojen rakenne ja se sisältää myös palvelimella sijaitsevan ohjelmisto pinon.



KUVA 4. Tietoverkon looginen kokonaisrakenne

Lähiverkot on eriytetty toisistaan loogisella tasolla antamalla niille omat osoiteavaruudet. Ohjelmisto pinon sisältävä palvelin ja verkkolevypalvelin sijaitsevat virtuaalisessa lähiverkossa. Verkkoon kytkeytyville päätelaitteille, kuten tietokoneille ja älypuhelimille on omat lähiverkkonsa. Seuraavalla sivulla olevassa taulukossa on listattuna lähiverkot, maskit ja verkkoon loogisesti maksimissaan kytkettävissä olevat laitemäärät. Palvelimien virtuaaliverkolle on mainittuna myös VLAN-tunniste.

TAULUKKO 1. Lähiverkkojen osoite taulu

Nimi	Osoiteavaruus	Maski	Laitemäärä	VLAN ID
Lähiverkko 1	192.168.1.0/24	255.255.255.0	254	-
Lähiverkko 2	192.168.2.0/24	255.255.255.0	254	-
Palvelimet	192.168.50.0/28	255.255.255.128	14	50

Verkon reitittämiseen on käytössä EdgeRouter Lite -reititin, jossa on kolme fyysistä porttia verkkoliikennettä varten. Ensimmäinen reitittimen portti asetetaan ohjaamaan liikennettä kytkimeen liitettyyn lähiverkkoon 192.168.1.0/24. Kytkimeen on yhdistettynä myös palvelin ja verkkolevypalvelin virtuaalilähiverkkona (VLAN). Toiseen reitittimen porttiin kytketään langattoman lähiverkon (WLAN) tukiasema ja siihen ohjataan langattomille verkkoyhteyksille tarkoitettu liikenne lähiverkosta 192.168.2.0/24. Viimeiseen porttiin liitetään julkinen ulkoverkko.

Palvelimille on luotu virtuaalilähiverkko osoiteavaruudella 196.168.50.0/28 ja verkon tunnukseksi on asetettuna 50. Virtuaalilähiverkkoja käyttämällä tietoliikenneverkko voidaan jakaa loogisesti useampaan osaan ja käyttää olemassa olevia yhteyksiä ja laitteita, mikäli niiltä löytyy tuki ominaisuudelle. Virtuaalilähiverkko mahdollistaa myös lähiverkon muodostamisen laitteille, jotka eivät sijaitse fyysisesti samassa paikassa.

Loogiseen virtuaalilähiverkkoon kuuluva tietoliikenne ohjataan IP-paketeissa olevilla tunnuksilla oikealle päätelaitteelle. Usein fyysisissä yhteyksissä käytetään termejä liputettu (engl. tagged) ja ei-liputettu (engl. untagged) erottamaan lähiverkkojen ja virtuaalilähiverkkojen liikenne. Mikäli laitteisto tukee VLAN-ominaisuuksia, voidaan paketteja liputtaa ja lähettää virtuaalilähiverkkoihin. Mikäli laitteistolta ei löydy tukea ominaisuudelle, IP-paketeista poistetaan virtuaalisestä lähiverkosta kertova VLAN-lippu.

Virtuaalilähiverkkojen avulla voidaan parantaa verkon suorituskykyä kiireisissä verkoissa, jolloin esimerkiksi reitittimille normaalisti kuuluvaa reititys työtä eri tietoverkkojen välillä voidaan siirtää kytkimien hoidettavaksi. Virtuaalilähiverkon protokolla sijoittuu OSI-mallissa tasolle kaksi. Kytkimet, jotka ovat tietoisia VLAN-ominaisuudesta, voivat ohjata liikennettä lippujen perusteella. Virtuaalilähiverkoilla voidaan myös tarvittaessa rajoittaa paikallisten lähiverkkojen välillä kulkevaa liikennettä, joka tuo tietoturvan kannalta lisäetuuksia. (Mitchell, B. 2018.)

Kotilaboratorion käyttöympäristön perusteella ei ole tarpeen rajoittaa yhdistettävien päätelaitteiden määrää normaaleissa lähiverkoissa. Käyttöympäristön laajuudesta riippuen tietoverkoissa saatetaan joutua pilkkomaan osoiteavaruutta pienempiin lohkoihin osoitteiden riittävyyden takaamiseksi. Osoiteavaruuden rajoittaessa laitteiden määrää, suuremmissa organisaatioissa voidaan siirtyä myös laajempaan yksityiseen käyttöön tarkoitettuun osoiteavaruuteen 10.0.0.0.

4.4 Tiedostojen jakaminen verkkolevypalvelimelta

Palvelimen ja sovelluskonttien tiedostoja varten tarvitaan tallennuspaikka verkkolevypalvelimelta. Yksinkertaisuudessaan voidaan luoda NFS-protokollalla jaettu hakemisto palvelimen ja verkkotallennuspalvelimen välille, johon voidaan tarvittaessa sijoittaa käytettävää dataa. Ominaisuuden käyttämiseksi palvelimen Ubuntu-käyttöjärjestelmään asennetaan NFS-protokollan työkalut ja asetetaan se asiakkaaksi (engl. client) verkkolevypalvelimen jakamaan hakemistoon. (ks. Liite 1, NFS: NFS-työkalujen asennus.)

Asennuksen jälkeen tehdään pysyvä kiinnityspiste (engl. mount) palvelimen käyttöjärjestelmään, jolloin verkkolevypalvelimen jakoon yhdistetään automaattisesti. Jako voidaan lisätä pysyvästi muokkaamalla asetustiedostoa /etc/fstab tiedostopolussa ja lisäämällä sinne seuraava rivi kommentteineen:

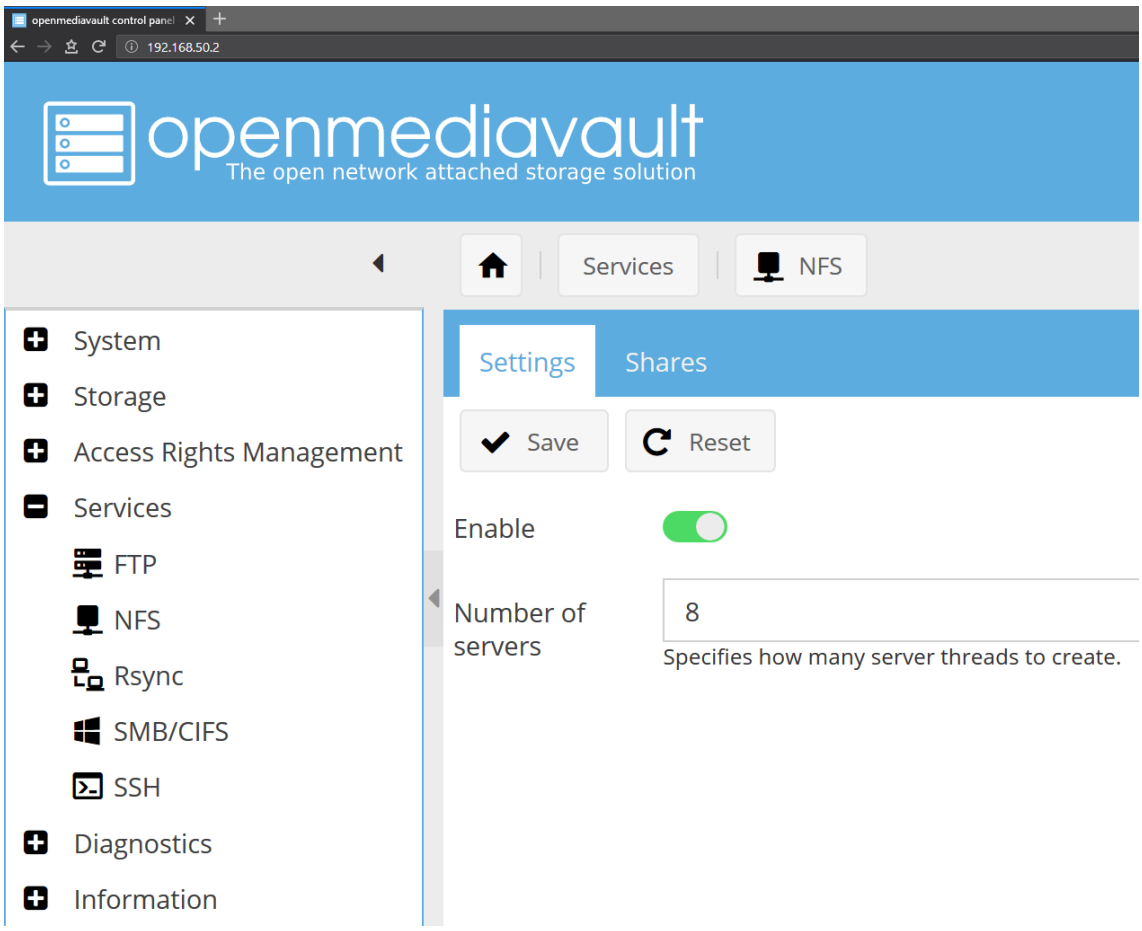
```
# OMV NFS Share
192.168.50.2:/export/<kansio>    /mnt/nfs    nfs    defaults    0 0
```

Tiedostoon määritellään yhdistämistä varten verkkolevypalvelimen IP-osoite ja jaettava hakemisto. Seuraavaksi määritellään palvelimella oleva tiedostopolku, johon jaosta käytettävät tiedostot tulevat näkyviin. Huomioitavaa on luoda tai käyttää palvelimella tyhjää kansiota, sillä jaossa oleva hakemisto lisätään näkyviin määriteltyyn kansioon ja edelliset tiedostot menetetään.

Tiedostojärjestelmän tyypiksi asetetaan ”NFS” ja lisäoptioina “defaults”, joka viittaa valmiiksi alustettuihin tiedostojärjestelmästä riippuviin asetuksiin. Optioissa voi halutessaan asettaa esimerkiksi tiedostoihin pääsyn saavat käyttäjät ja jaossa käytettävät kirjautumistunnukset.

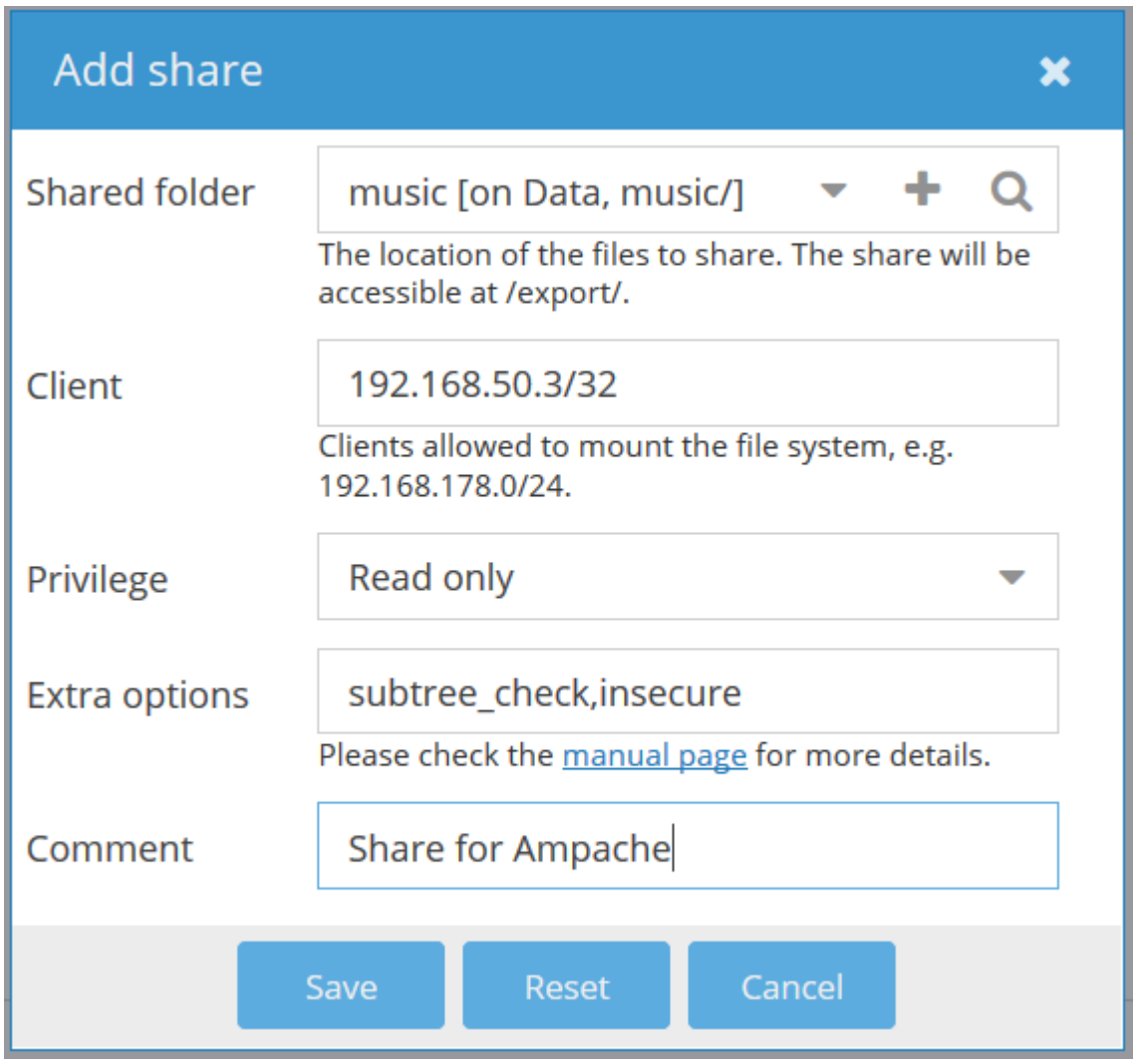
Lopussa olevat kaksi numeroa määrittelevät varmuuskopioinnin ja laitteiden virheentarkastuksen. Ensimmäinen numeroarvo määrittelee, luodaanko tiedostojärjestelmästä varmuuskopio ja jälkimmäinen numero laitteen tarkastusmenettelyn käynnistyksen yhteydessä. Asetettaessa numeroarvot nollaksi, varmuuskopiota ei luoda ja yhdistettyä laitetta ei tarkisteta erikseen.

Verkkolevypalvelimelle asennetusta openmediavault-käyttöjärjestelmästä (OMV) tarvittavien jakojen luominen on yksinkertaista. Selaimella voidaan yhdistää selainpohjaiseen hallintapaneeliin ja aktivoida NFS-protokollan palvelu jakojen hallintaa varten.



KUVA 5. Openmediavault: Hallintapaneelin NFS palveluasetukset

Samalla sivulla on myös välilehti jakojen (engl. shares) luomiseen. Pudotus valikosta valitaan jaettava kansio, johon voidaan liittyä osoittamalla /export/<kansio> polkuun. OMV luo loogisen /export/ hakemiston jakojen yhteydessä. Luotaessa uusia jakoja, kannattaa ottaa huomioon asiakaskoneiden verkon suuruus. Jako voidaan asettaa haluttaessa esimerkiksi koko lähiverkon saataville. Esimerkki kuvassa luodaan Ampache-sovellukselle tarvittava jako.



KUVA 6. Openmediavault: NFS-jaon luonti Ampache-sovellusta varten

Ampache-sovellusta varten tehtävässä jaossa määritellään yhteysosoitteeksi palvelimen IP-osoite verkkomaskilla 255.255.255.255 (/32). Luotuun jakoon voi yhdistää ainoastaan yksittäinen päätelaite määritetyllä IP-osoitteella. Menetelmää voidaan käyttää, mikäli halutaan tarkasti hallittuja jakoja. Asettamalla esimerkiksi kokonaisen lähiverkon suuruisen asiakaskoneryhmän, voidaan jakoon yhdistää yhtäaikaisesti useammasta päätelaitteesta. Ampache-sovellus lukee hakemistosta musiikkitiedostoja, jolloin jaossa oleviin tiedostoihin annetaan ainoastaan lukuoikeudet. Luotaessa jakoa esimerkiksi varmuuskopioiden tekemiseen, tarvitaan hakemistolle luku- ja kirjoitusoikeudet.

4.5 Palvelimen ulkoverkkoyhteydet

Reitittimen ja palvelimen palomuurisäännöt asetetaan oletuksena hylkäämään ulkoverkosta kohdistuvat yhteyspyynnöt kaikkiin portteihin. Palomuurista avataan ainoastaan tarvittavat portit sovelluksille, jotka vaativat kommunikointi yhteyden ulkoverkkoon. Käytössä olevien avoimien porttien rajaaminen pienentää ulkoverkosta palvelimelle kohdistuvaa hyökkäys pinta-alaa.

Ulkoverkkoon yhdistyvät web-sovellukset asetetaan palvelimelta näkymään oletuksena suojattuun yhteyteen (HTTPS) käytettävän portin 443 kautta. Palvelimelle voidaan yhdistää verkkotunnuksen (engl. domain) perusteella ja taustalla oleville web-sovelluksille tehdään yhteys säännöt käänteisen välityspalvelimen (engl. reverse proxy) asetuksissa. Hyödyntämällä käänteistä välityspalvelinta, voidaan ulkoverkkoon näkyviä web-sovelluksia ohjata yhdistämään halutun portin kautta. Käänteinen välityspalvelin tarkastaa yhteydenoton aikana käytettävän verkkotunnuksen ja vertaa sitä asetuksissa määritettyihin sääntöihin. Yhteyspyynnön täsmätessä välityspalvelimen sääntöihin, voidaan web-sovellukselle saapunut yhteys sallia ja päästää läpi.

Suojattuja yhteyksiä voidaan hyödyntää ilman auktorisoidun tahon allekirjoittamia sertifikaatteja. Let’s Encrypt on valtuutettu digitaalisten sertifikaattien luovuttaja (Linux Foundation, 2019). Let’s Encrypt mahdollistaa ilmaisten allekirjoitettujen sertifikaattien luomisen omille verkkotunnuksille. Sertifikaattien luomiseksi tarvitsee hakuprosessissa osoittaa omistavansa tai hallinnoivansa verkkotunnuksia. Käyttämällä allekirjoitettua sertifikaattia yhteyden luotetaan olevan suojattu ja esimerkiksi selaimet eivät ilmoita suojausongelmasta. Let’s Encrypt:in sertifikaatit ovat voimassa 90 päivän ajan, jonka jälkeen ne pitää uusia.

5 SOVELLUKSET

5.1 Webmin

Webmin on palvelimen hallintaan tarkoitettu verkkoselaimessa toimiva graafinen käyttöympäristö, joka mahdollistaa esimerkiksi palomuurin, porttiohjauksien ja nimipalveluiden asetuksien muokkaamisen. Käyttöliittymään voidaan lisätä uusille palveluille tuki lisämoduulien avulla. Nimipalvelimen (DNS) asetukset ja palomuurin säännöt luodaan paikalliseen verkkoon Webminin avulla. Webmin on kokonaisuudessaan hyvä lisätyökalu järjestelmän hallinnointia varten.

Webmin voidaan lisätä Ubuntu-käyttöjärjestelmään suoraan ladattavana ja asennettavana DEB-pakettina kehittäjäyhteisön kotisivuilta (Webmin, 2018). Käytettävyyden ja päivitysten helpottamiseksi voidaan Ubuntuun lisätä Webminin arkisto (engl. repository).

Komentorivillä paketinhallinnan lähteisiin lisätään uusi arkisto muokkaamalla asetustiedostoa /etc/apt/sources.list tiedostopolusta. Muokkaaminen ja asennus vaatii järjestelmänvalvojan oikeuksia (sudo-oikeudet). Tiedoston voi avata muokattavaksi oletus tekstieditorilla tai lisätä arkiston tiedot liitteissä olevalla komennolla (ks. Liite 1, Webmin: Arkiston lisäys Ubuntun paketinhallintaan).

Tiedoston loppuun lisätään uudelle riville arkiston tiedot ja mukaan voidaan liittää lisätiedoksi kommenttirivi. Liitteissä oleva komento lisää suoraan kommenttirivin ja arkiston. Kommenttirivin voi lisätä tiedostoon sijoittamalla rivin alkuun risuaidan (#). Tiedostoon lisättävä rivi (kommentteineen) on:

```
# Webmin repository
deb https://download.webmin.com/download/repository sarge contrib
```

Lisäyksen jälkeen tiedostosta voidaan poistua tallentaen tehdyt muutokset. Ennen pakettien noutamista ja asentamista lisätään kehittäjän luoma GPG-avain luotettujen listalle. Avainta lisättäessä saattaa tulla ilmoitus gnupg-paketin puuttumisesta. Paketti on tarvittaessa asennettava GPG-avaimien käsittelyä varten.

Gnupg-paketin asentaminen, avaimen noutaminen ja lisääminen voidaan tehdä liitteissä olevien komentojen avulla (ks. Liite 1, Webmin: Kehittäjän GPG-avaimen lisäys luotettujen listalle).

Onnistuneesta GPG-avaimen lisäyksestä ilmoitetaan komentorivillä “OK” viestillä. Suojattuun yhteyteen (HTTPS) perustuvaan pakettien asennukseen tarvitaan tuki paketinhallintaan. Webmin ja tuki suojatulle yhteydelle voidaan asentaa paketinhallinnan työkaluilla käyttäen liitteissä olevia komentoja (ks. Liite 1, Webmin: Arkiston päivitys ja Webminin asennus).

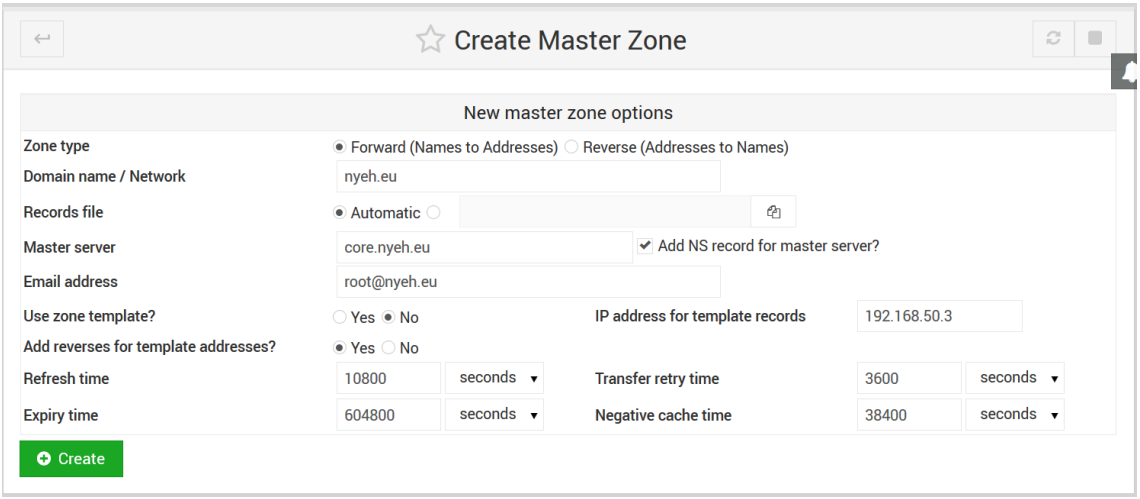
Webminin pakettiin on asetettuna riippuvuuksia (engl. dependency) muihin ohjelmiin ja työkaluihin. Asentamalla suoraan paketinhallinnasta ja sen työkaluilla, riippuvuudet ratkotaan ja asennetaan automaattisesti käyttäjän hyväksynnällä. Webminin asennusohjeissa on dokumentoituna tarvittavat riippuvuudet, mikäli asennuksen haluaa suorittaa suoraan DEB-paketista.

Asennuksen jälkeen Webminin hallintapaneeliin voi yhdistää verkkoselaimella ja hallinnoida käyttöjärjestelmää graafisella käyttöympäristöllä. Oletuksena Webmin käyttää porttia 10000 yhdistämiseen. Hallintapaneeliin voi kirjautua ainoastaan pääkäyttäjän (root) ja järjestelmänvalvojan (sudo) oikeudet omaavat käyttäjät. Esimerkiksi lähiverkosta palvelimella sijaitsevaan hallintapaneeliin voidaan yhdistää menemällä selaimella osoitteeseen https://192.168.50.3:10000/. Webmin luo suojattua yhteyttä varten itse allekirjoitetun sertifikaatin ja selain ilmoittaa epäluotettavasta suojatusta yhteydestä. Selaimeen voi lisätä poikkeuksen ilmoituksen ohessa, jolloin yhteys sallitaan.

5.1.1 Paikallinen nimipalvelin (DNS)

Webminin hallintapaneelissa voidaan luoda lähiverkkoon paikallinen nimipalvelin (DNS), joka pohjautuu BIND-ohjelmistoon. BIND voidaan asentaa Webminin hallintapaneelin käyttämättömien moduulien valikosta. Moduulin asentamiseen käytetään Ubuntun paketinhallintaa tarvittavien pakettien ja riippuvuuksien asentamiseen automaattisesti. Ohjelmasta asennetaan viimeisin vakaaksi todettu versio.

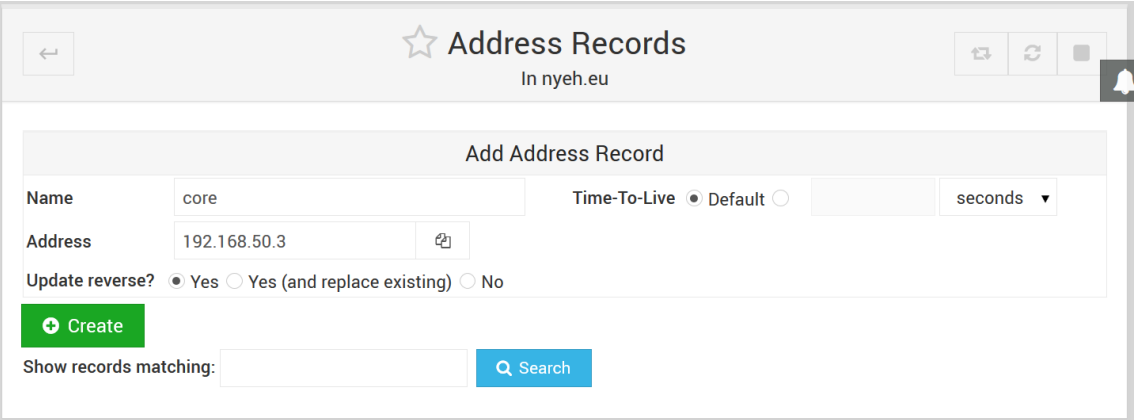
Lisämoduulin aktivoinnin jälkeen voidaan BIND-ohjelman asetus sivuilta luoda uusi vyöhyke (DNS Zone) klikkaamalla olemassa olevista vyöhykkeistä “Create Master Zone” painiketta. Käyttäjä ohjataan uudelle sivulle tarvittavien tietojen syöttämiseen vyöhykkeen luomista varten.



KUVA 7. Webmin: DNS-vyöhykkeen luonti verkkotunnukselle

Yllä olevassa kuvassa käytetään olemassa olevaa verkkotunnusta. Paikallista nimipalvelinta luotaessa kannattaa käyttää verkkotunnusta, jonka omistaa tai sen käyttäminen esimerkkinä on sallittua. Verkkotunnuksena käytetään ”nyeh.eu” ja palvelimelle myönnetään aliverkkotunnus (engl. subdomain) ”core.nyeh.eu”, jota käytetään myös DNS-hallintapalvelimena. Sähköposti osoitteeksi paikallisessa nimipalvelimessa voi laittaa kuvitteellisen, ei olemassa olevan osoitteen. IP-osoitteeksi määritellään palvelimen paikallinen osoite 192.168.50.3.

Uuden vyöhykkeen luomisen jälkeen siirrytään automaattisesti verkkotunnuksen vyöhykkeen hallintasivulle. Vyöhykkeeseen lisätään uusi osoitekirjaus (Address Record) valitsemalla vyöhykkeen hallintasivulta “Address”.

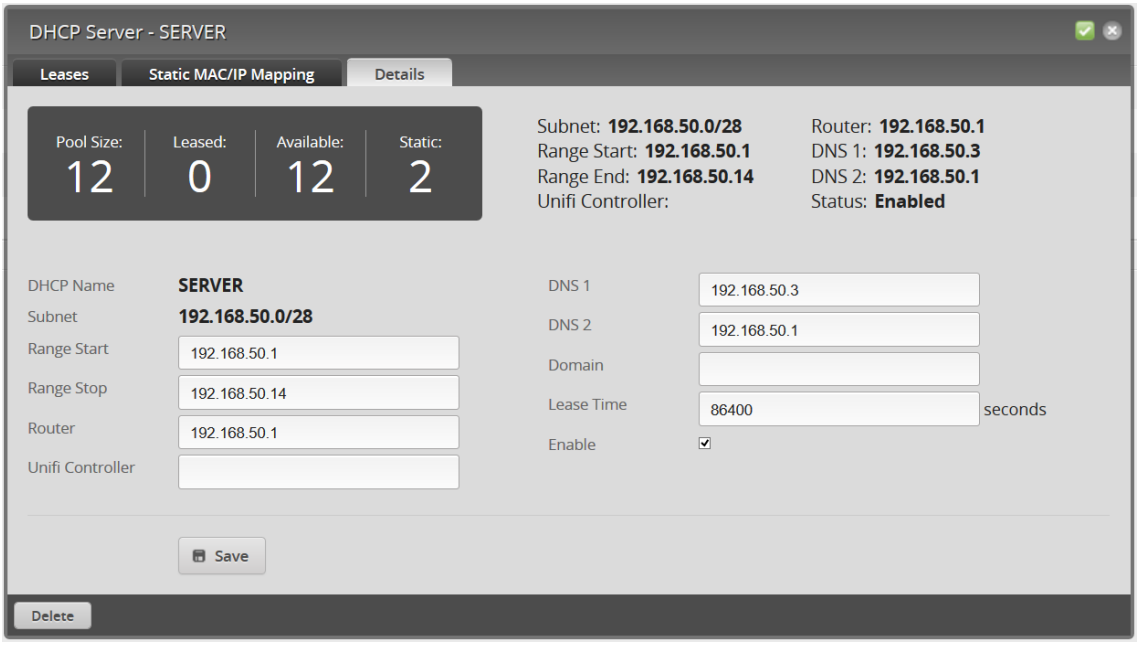


KUVA 8. Webmin: Osoitekirjauksen lisääminen vyöhykkeeseen

Ensimmäisenä kirjauksena määritellään nimipalvelun tarjoava palvelin, jonka nimi näkyy alitunnuksena. Aliverkkotunnus “core” ohjataan palvelimen IP-osoitteeseen, jolloin palvelimeen voidaan yhdistää lähiverkosta “core.nyeh.eu” verkkotunnuksella. Yksilöityjä aliverkkotunnuksia voidaan lisätä useampia esimerkiksi lähiverkon muille laitteille.

Nimipalvelimen perusasetuksilla lähiverkossa voidaan yhdistää laitteiden välillä helpommin muistettavilla verkkotunnuksilla. Reitittimen asetuksissa asetetaan ensisijaiseksi nimipalvelimeksi paikallinen palvelin, jolloin DNS-kyselyt lähetetään ensimmäisenä sille. Halutessaan voi lisätä käänteisen vyöhykkeen, jolloin IP-osoitteella tehtyyn nimikyselyyn vastataan verkkotunnuksella, mikäli vastaavuus löytyy.

Reitittimien välillä on erilaiset menettelytavat asetusten muokkaamiseen ja alla olevassa esimerkkikuvassa EdgeRouter Lite -reitittimessä asetetaan DHCP:n asetuksista ensisijaiseksi nimipalvelimeksi paikallinen DNS-palvelin ja toissijaiseksi nimipalvelimeksi reititin.

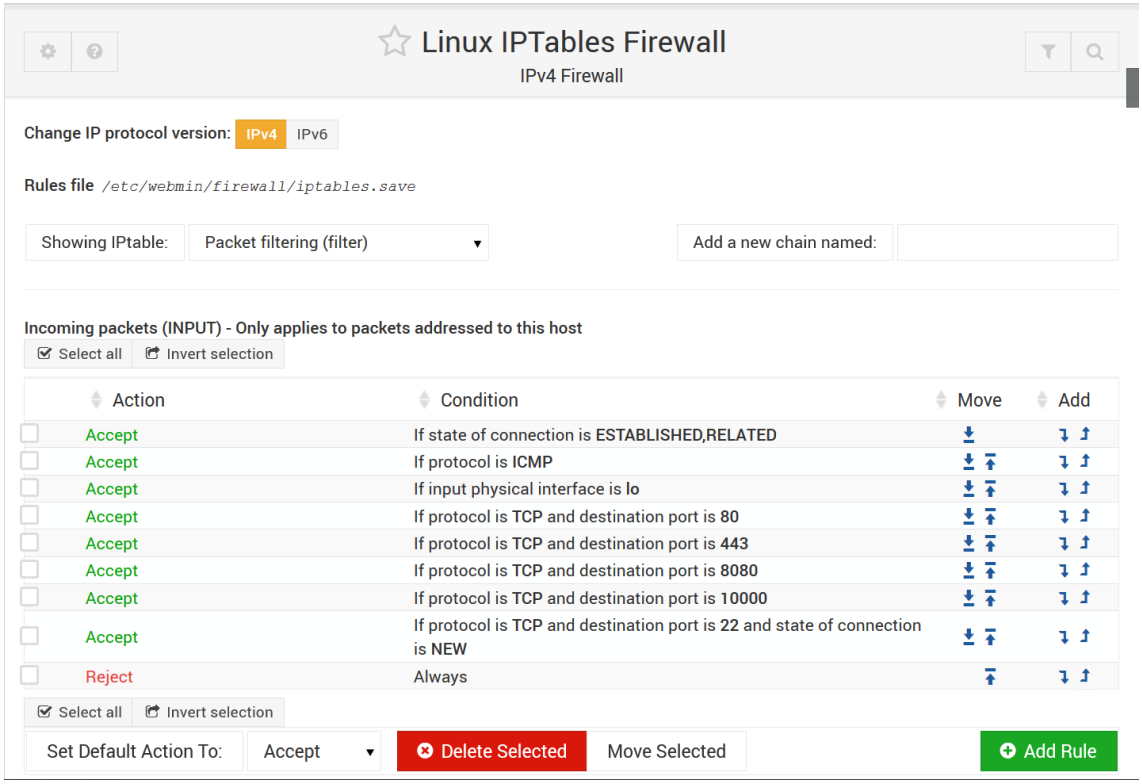


KUVA 9. EdgeRouter Lite: DHCP-asetukset palvelimien virtuaalilähiverkolle

Lisättäessä nimipalvelin ensisijaiseksi myös muihin lähiverkkoihin, voidaan niistä yhdistää verkkotunnuksen perusteella palvelimella tarjottaviin verkkosivuihin. Verkkotunnuksen vastaavuuden lisäämisen ansiosta lähiverkosta voidaan yhdistää Ampache-sovellukseen suoraan verkkotunnuksella ja selaimen HTTPS-yhteyspyynnössä näkyy verkkotunnus IP-osoitteen sijaan. Käänteisen välityspalvelimen asetuksissa määritellyt säännöt täyttyvät ja liikenne ohjautuu oikein. Lähiverkosta yhdistettäessä verkkosivut eivät näyttäydy rikkinäisinä ja liikenne on suojattu päätelaitteelle asti.

5.1.2 Palomuuri

Webmin hallintapaneelin avulla voidaan hallita palomuurisääntöjä. Graafisessa käyttöliittymässä voidaan lisätä uusia sääntöjä nopeasti valitsemalla valikoista sääntöön lisättävät asiat. Webminin palomuuri moduuli hallinnoi sääntöjä käyttäen IPTables-apuohjelmaa ja sen toiminta perustuu Linux-ytimessä olevaan pakettien suodatussääntöjen luomiseen.



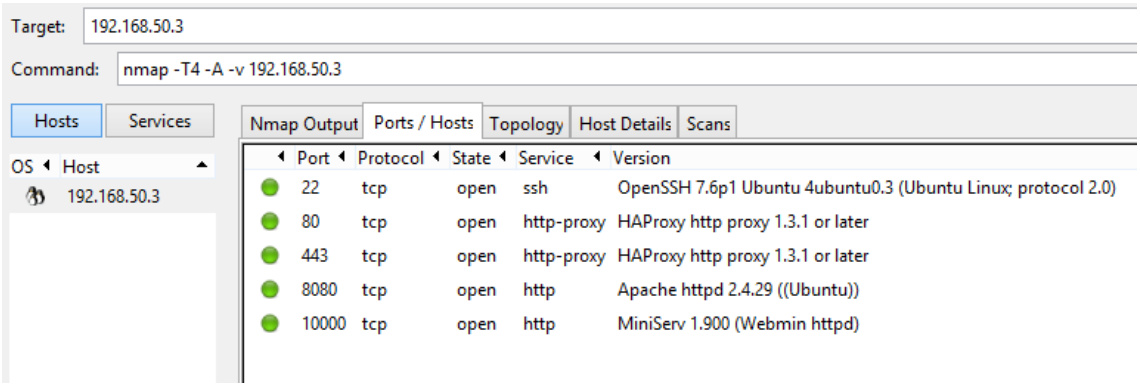
KUVA 10. Webmin: Palomuurisääntöjen hallintasivu

Yllä olevassa kuvassa näkyy perusnäkymä palomuurisääntöjen hallintaan. Kuvassa luodut säännöt ovat yksinkertaistetut ja nopeat. Palomuuriin lisätään sääntö palvelimeen kohdistuvasta HTTP- ja HTTPS-liikenteestä portteihin 80 ja 443. Ampache-sovelluskontti käyttää kontin ulkopuolisena porttina 8080, jolloin sen liikenne on sallittava paikallisesti. Webmin hallintapaneeliin yhdistetään oletuksena portin 10000 välityksellä. SSH-yhteyksiä varten päästetään sisään liikenne portista 22, mikäli yhteystyyppi on uusi. Säännöistä poikkeavat sisään tulevat yhteyspyynnöt hylätään. Mikäli palvelimelle lisätään uusia tietoverkkoon kohdistuvia palveluita, tarvitsee palomuuriin lisätä tai päivittää sääntöjä.

Palvelimella olevalla palomuurilla voidaan pienentää hyökkäyksille altistuvaa pinta-alaa. Haluttaessa liikennettä voidaan suodattaa huomattavasti tarkemmin esimerkiksi asettamalla kohdeosoitteeksi palvelimen IP-osoite ja tiukentamalla sääntöjä. Docker esimerkiksi hallinnoi yhteyksiään sovelluskontteihin IPTables-apuohjelman välityksellä ja lisää tarvittavat säännöt ohjelman toiminnan takaamiseksi. Webminin hallintapaneelissa näkyy myös Dockerin lisäämät säännöt.

5.1.3 Palomuurin testaus

Asennettua palomuuria voidaan testata kohdistamalla palvelimelle porttien skannaus. Porttien skannauksessa haistellaan palvelimella olevia avoimia portteja ja mahdollisesti pyritään kaivamaan taustalla olevat palveluprosessit. Porttien skannaamiseen voidaan käyttää esimerkiksi Zenmap-ohjelmatyökalua (Zenmap, 2019).



KUVA 11. Zenmap: Paikalliseen lähiverkkoon näkyvät palvelimen portit

Yllä olevasta kuvasta nähdään skannauksen aikana havaitut palvelimella olevat avoimet portit, jotka ovat saavutettavissa lähiverkosta. Versioissa näkyy portin takana oleva aktiivinen palveluprosessi. Skannauksesta huomioitavaa on avoimena lähiverkkoon näkyvä 8080 portti, joka on tarkoitettu Ampache-sovelluskontin ulkoiseksi portiksi. Portin pitäisi näkyä avoimena ainoastaan palvelimen sisäisenä porttina. Palomuurista ja sen asetuksista on havaittu siis vuotokohta.

Ulkoverkosta palvelimelle kohdistuva liikenne osuu alustavasti reitittimellä olevalle palomuurille ja sen portinohjauksessa tehtyihin sääntöihin. Reitittimeltä on ohjattu ainoastaan porteista 80 ja 443 tuleva liikenne palvelimelle. Myöhemmin reitittimeen lisätään ohjaus portille 22, joka on käytössä oletuksena SSH-liikenteelle ja tarvitaan etäohjauksen mahdollistamiseksi ulkoverkosta.

5.2 HAProxy

HAProxy on pääasiassa tarkoitettu palvelimiin kohdistuvan TCP/HTTP-protokollaa käyttävien yhteyksien kuorman jakamiseen (HAProxy, 2019a). HAProxy:n toimiessa kuormantasaajana ja välityspalvelimena, voidaan palvelimelle kohdistuvia kutsuja ohjata portin ja verkkotunnuksen perusteella. Ohjaukset voidaan tehdä ja kohdistaa samalle palvelimelle tai tarvittaessa useammalle eri palvelimelle.

Ubuntun paketinhallinnasta löytyy suoraan vakaaksi todettu versio HAProxy:stä. Paketin voi asentaa suoraan käyttämällä liitteissä olevaa komentoa (ks. Liite 1, HAProxy: Asennus paketinhallinnasta). Asennuksen aikana ohjelma ladataan ja asennetaan, mutta sen prosessia ei käynnistetä suoraan. HAProxy:n palvelua ei kannata käynnistää ennen kuin tarvittavat muutokset on tehty asetustiedostoon.

HAProxy:n asetustiedosto Ubuntu-käyttöjärjestelmässä löytyy oletuksena /etc/haproxy/haproxy.cfg tiedostopolusta. Hallinnointiin ja asetuksien tekemiseen on saatavilla kattava dokumentaatio HAProxy:n yhteisöltä (HAProxy, 2019b). Lähtökohdiltaan asetuksien perusteet ovat kohtuu yksinkertaiset. Välityspalvelimen rakentaminen perustuu etupään ja takapään asetteluun.

Etupäässä (engl. frontend) kuunnellaan määriteltyyn porttiin tulevia yhteyspyyntöjä ja verrataan niitä pääsylistojen (ACL) sääntöihin. Mikäli yhteyspyyntö täsmää pääsylistoissa oleviin sääntöihin, ohjataan liikenne eteenpäin taustalle (engl. backend).

Tausta(-palvelimen) asetuksissa määritellään kuormanhallinta ja lisäoptiot. Näiden lisäksi määritellään liikenteen ohjaaminen palvelimien kuorman perusteella. Palvelimiin voidaan asettaa myös terveystarkastus (engl. health check) jolloin tarkistetaan, onko palvelin aktiivisena ja pystyykö se vastaanottamaan uusia yhteyksiä.

5.2.1 Välityspalvelimen asetukset

Välityspalvelimen toiminnassa määritellään sisään tulevalle liikenteelle tehtävät toimenpiteet ja ohjataan haluttu liikenne palvelimelle. HAProxy:n asetustiedosto löytyy /etc/haproxy/haproxy.conf tiedostopolusta ja tiedostoon tehdyt asetukset löytyvät liitteistä. (ks. Liite 4, Konfiguraatiot: HAProxy.)

Globaaleissa muuttujissa (engl. global variable) voidaan määritellä yläraja samanaikaisesti vastaanotettaville yhteyksille ja asettaa korkein käytettävä bittimäärä väliaikaisten DHE-avaimien generoimiseen. HAProxy asettaa oletuksena käytettävän bittimäärän vastaamaan palvelimella käytössä oleviin RSA-avaimiin. Esimerkiksi Let’s Encrypt:in sertifikaatit luodaan oletuksena 2048-bittisellä RSA-avaimella. Muut globaalit muuttujat on jätetty oletusasetuksille.

Oletukset (engl. defaults) kohdassa määritellään välityspalvelimen tila ja tarvittavat optiot. HAProxy on asetettuna välittämään verkkoliikennettä, jolloin optiot “forwardfor” ja “http-server-close” lisätään. Optioilla suojattu yhteys voidaan terminoida välityspalvelimella ja jatko-ohjata liikenne suojaamattomana taustapalvelimelle.

Etupään “http-in” asetuksissa otetaan vastaan saapuvat yhteyspyynnöt portista 80 (HTTP). Suojaamaton yhteys ohjeistetaan kääntymään suojattuun käyttäen ohjaukseen HAProxy:n skeemaa HTTPS-liikenteelle. Palvelimella ei käytetä rakennus hetkellä sovelluksia ulkoverkosta suojaamattomalla yhteydellä, jolloin liikenne voidaan oletuksena kääntää suojatulle.

Etupään “https-in” asetuksissa otetaan vastaan saapuvat yhteyspyynnöt portista 443 (HTTPS). Suojattua yhteyttä varten sisällytetään sertifikaatit määritetystä hakemistosta. Hakemisto sisältää yksittäisiin tiedostoihin yhdistettynä verkkotunnusten sertifikaatti ketjun ja yksityisen avaimen. Suojatulle HTTPS-liikenteelle voidaan koventaa asetuksia ja kovennukseen käytetyt parametrit perustuvat lähteeseen (Zuba, 2015). Liikennettä ohjataan pääsylistojen (ACL) perusteella, jolloin sisään tulevista yhteyspyynnöistä parsitaan käytetty verkkotunnus (esimerkiksi ampache.nyeh.eu) ja ohjataan liikenne osuman perusteella taustapalvelimelle.

Taustapäässä liikenne ohjataan suojaamattomana paikallisesti porttiin 8080, jota Ampache-sovelluskontti kuuntelee ulkoisesti. Yhteyksien määrän yläraja taustapalvelimelle voidaan tarvittaessa asettaa ja tulevaisuudessa voidaan mahdollisesti testata kuormantasausta yhteyksien määrän perusteella.

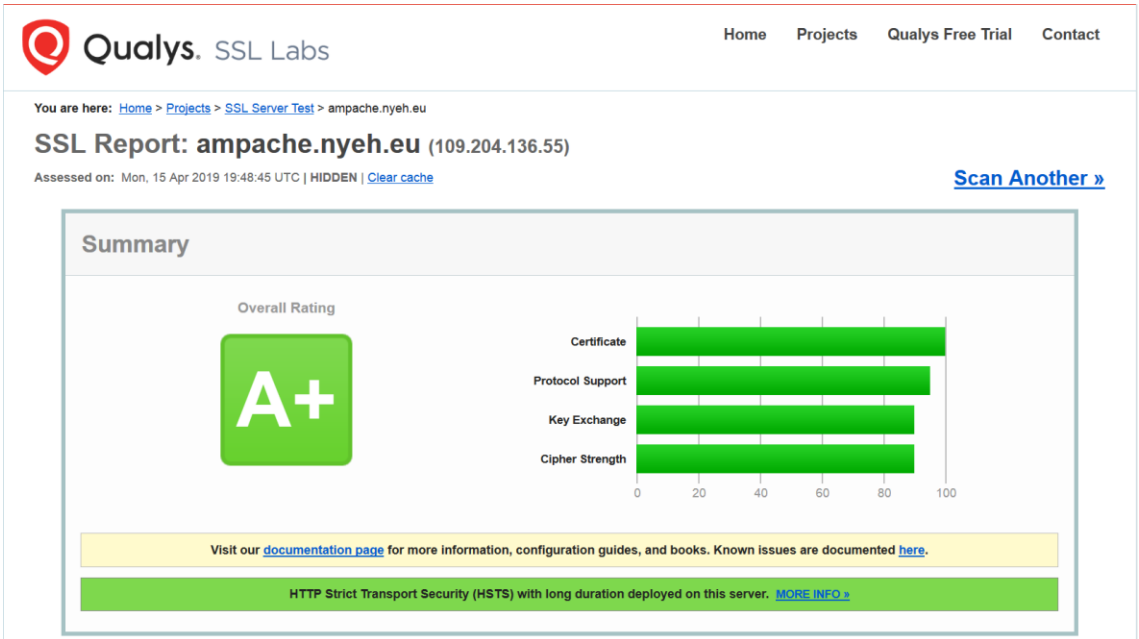
HAProxy:n muokatun asetustiedoston eheys voidaan tarkistaa ohjelman avulla käyttämällä liitteissä olevaa komentoa. Tarkistuksessa otetaan huomioon mm. syntaksivirheet ja ristiriitaiset asetukset. Tarkistus on suositeltavaa tehdä aina muutoksien jälkeen, ennen kuin HAProxy:n palveluprosessin käynnistää. (ks. Liite 1, HAProxy: Asetustiedoston tarkastus.)

Asetustiedoston muodostamisen ja tarkastamisen jälkeen voidaan käynnistää HAProxy:n palveluprosessi. Palveluprosessin voi käynnistää Ubuntu-käyttöjärjestelmässä liitteissä olevalla komennolla. (ks. Liite 1, HAProxy: Palvelu prosessin käynnistys).

5.2.2 Suojatun yhteyden testaaminen

Suojattujen yhteyksien testaamiseen voidaan käyttää ulkoista verkkopalvelua. Työkalun avulla saadaan selville palvelimelle kohdistuvan suojatun liikenteen salauksen vahvuus ja sertifikaatin kättelyssä käytettävä metodi. Esimerkiksi Qualys tarjoaa verkossa työkaluja suojatun yhteyden testaamiseen (Qualys, 2019).

Testaukseen käytettävään työkaluun syötetään haluttu verkkotunnus ja verkkotunnuksen takana olevalle palvelimelle tehdään syväanalyysi. HAProxy:n asetuksien perusteella suojatut yhteyspyynnöt kohdistuvat välityspalvelimelle ja se terminoidaan siihen. Jatkoyhteys taustapalvelimelle tehdään suojaamattomana, mutta palvelimelta ulkoverkkoon menevä yhteys on silti suojattu.



KUVA 12. Qualys: Lopputulos palvelimen suojatun yhteyden testaukselle

HAProxy:n asetuksissa tehtiin kovennuksia suojatun yhteyden muodostamiseen ja oletuksissa määriteltiin käytettäväksi uudempia salakirjoitusmenetelmiä. Testissä voidaan tarkemmin tarkastella palvelimen hyväksymiä salausmenetelmiä ja käytössä olevan sertifikaatin tietoja. Halutessaan sertifikaatin luonnissa voidaan käyttää voimakkaampia 4096-bittisiä RSA-salausavaimia. Palvelimen suojatulle yhteydelle luotu sertifikaatti on alustavasti 2048-bittinen. Sertifikaatin noutaminen käsitellään tarkemmin CertBot-ohjelman kappaleessa.

5.3 CertBot

Certbot on EFF (Electronic Frontier Foundation) järjestön kehittämä ja ylläpitämä työkalu Let’s Encrypt sertifikaattien automaattista luomista varten (EFF, 2019). Ohjelma yksinkertaistaa sertifikaatin luomisprosessia ACME-protokollan ylitse. Certbotin verkkosivuilla voi valita ohjeistuksen käytössä olevan käyttöjärjestelmän, verkkopalvelimen tai välityspalvelimen perusteella. Ubuntulla certbot-paketti voidaan asentaa suoraan EFF:n ylläpitämästä PPA-arkistosta (Personal Package Archive). Alustavasti asennetaan PPA-arkistojen hallinnointia helpottavat skriptit ja sen jälkeen lisätään arkistoon ohjaava linkki. Asennuksen ja arkiston lisäämisen komennot löytyvät liitteistä (ks. Liite 1, Certbot: Perustyökalut ja arkiston lisäys).

“Universe” on Ubuntu-yhteisön ylläpitämä avoimen lähdekoodin arkisto. Arkisto on oletuksena aktiivisena asennettaessa Ubuntu Server 18.10 ja Ubuntu Server 18.04 LTS -versioita. Arkisto saattaa olla aktivoitamattomana eri varianteissa, joten komennon ajaminen on suositeltavaa asennusohjeiden siitä mainittaessa. Arkiston lisäämisen jälkeen paketinhallinnan tietokanta voidaan päivittää uusilla tiedoilla ja asentaa certbot-ohjelmisto riippuvuuksineen liitteissä olevalla komennolla (ks. Liite 1, Certbot: Arkiston päivitys ja ohjelman asennus).

Esimerkki komennoissa olevilla ”&&” merkeillä voidaan jonottaa useampi suoritettava komento yhdellä komentosyötteellä. Usein lyhyitä komentoja tai vakituisesti käytettyjä komentoja jonotetaan peräkkäin niiden käytön nopeuttamiseksi.

Riippuen käytettävästä ohjelmistosta ja käyttöjärjestelmästä, loppuun asti automatisoitu sertifikaatin luonti ei ole välttämättä mahdollista työkalun avulla. HAProxy-ohjelmaa välityspalvelimena käytettäessä joudutaan sertifikaatti luomaan ilman automaattista lisäystä ohjelman asetuksiin.

5.3.1 Verkkotunnukset

Ennen sertifikaatin noutamista tarvitaan käyttöön verkkotunnus ja sellaisen voi halutessaan ostaa verkkotunnusten välittäjiltä. Esimerkiksi Suomen ylätason verkkotunnusta (.fi) hallinnoi Traficom (liikenne- ja viestintävirasto). Hallinnoija voi luovuttaa oikeuden rekisteröidä ja myydä verkkotunnuksia verkkotunnusten välittäjille. Verkkotunnusten välittäjiltä on saatavilla useita erilaisia ylätason verkkotunnuksia ICANN-organisaation vapauttaessa niiden varaamisen.

Käytettävissä oleva ”nyeh.eu” verkkotunnus on ostettu Joker.com verkkotunnusten välittäjältä. Verkkotunnuksesta ohjataan välittäjän tarjoamalta nimipalvelimella aliverkkotunnukseen “ampache.nyeh.eu” kohdistuva liikenne palvelimen julkiseen ulkoverkon IP-osoitteeseen.

Välittäjän tarjoamaan nimipalvelimeen luodaan uusi DYNA (Dynamic Address) kirjaus, joka mahdollistaa julkisen IP-osoitteen mukana dynaamisesti muuttuvan kirjauksen. Dynaamisesti muuttuvaa kirjausta voidaan päivittää automaattisesti

esimerkiksi asentamalla DDClient-ohjelmisto Ubuntun paketinhallinnasta (ks. Liite 1: CertBot: DDClient-ohjelman asennus).

Dynaamisen kirjauksen luomisen lisäksi verkkotunnuksen välittäjä tarjoaa käyttäjänimen ja salasanan kirjauksen päivittämiseen automaattisesti ohjelman avulla. Menettelytavat ja dynaamisen kirjauksen luominen vaihtelee palvelutarjoajien välillä. DDClient-ohjelmiston asetustiedoston luomiseen voidaan käyttää välittäjän tarjoamaa dokumentaatiota (Joker.com, 2018). Liitteisiin on lisättynä käytetty asetustiedosto ilman arkaluonteista tietoa, kuten käyttäjätunnusta ja salasanaa. Ohjelmiston asetustiedosto sijaitsee oletuksena Ubuntu-käyttöjärjestelmässä /etc/ddclient.conf tiedostopolussa. (ks. Liite 4, Konfiguraatiot: DDClient.)

5.3.2 Sertifikaatit

Sertifikaattien uusiminen CertBot-ohjelmalla on yksinkertaista. Ohjelma voidaan ajaa itsenäisessä (engl. standalone) tilassa, jolloin haetaan ainoastaan sertifikaatit verkkotunnukselle. Noudon jälkeen ne voidaan lisätä välityspalvelimen käytettäväksi. Sertifikaatti tarvitaan Ampache-sovelluskontin liikenteen suojaamiseen ja varmentamiseen. Esimerkki sertifikaatin noutamisesta ohjelman itsenäisessä tilassa löytyy liitteistä (ks. Liite 1, CertBot: Sertifikaatin nouto itsenäisesti.)

Sertifikaatin luomiseen ja noutamiseen tarvitaan yhteydenpitoon sähköpostiosoite, johon lähetetään viesti ongelmatilanteissa. Sähköpostiin saapuu myös muistutus, kun sertifikaatin viimeinen voimassaolo päivä lähestyy. Lukemalla ja hyväksymällä käyttöehdot sertifikaatit luodaan.

CertBot sijoittaa luodut sertifikaatit oletuksena /etc/letsencrypt/live/<verkkotunnus> tiedostopolkuun. Esimerkiksi Ampache-sovelluskonttia varten luodut sertifikaatit sijoittuvat aliverkkotunnuksen perusteella /etc/letsencrypt/live/ampache.nyeh.eu tiedostopolkuun.

Verkkotunnusta varten luodaan neljä tiedostoa. Tiedostossa “cert.pem” sijaitsee verkkotunnus kohtainen sertifikaatti. Tiedostossa “chain.pem” sijaitsee sertifikaatti ketju, joka sisältää auktorisoidut sertifikaattien luovuttajat aina loppukäyttäjälle asti. Tiedostossa “fullchain.pem” on yhdistelmä aiemmasta kahdesta tiedostosta. Tiedostossa “privkey.pem” sijaitsee haetun sertifikaatin yksityinen avain. Yksityistä avainta ei saa luovuttaa ulkopuoliselle taholle.

Sertifikaatti halutaan lisätä välityspalvelimena toimivan HAProxy-ohjelman käytettäväksi. Sertifikaatti ketju tarvitsee yhdistää yksityisen avaimen kanssa yhdeksi tiedostoksi. Yhdistettyjä sertifikaatti tiedostoja varten luodaan esimerkissä uusi kansio HAProxy-ohjelman juurikansioon, muokataan kansion oikeuksia ja luetaan tarvittavat tiedostot yhdeksi kokonaiseksi tiedostoksi (ks. Liite 1, CertBot: Sertifikaatin muunnos yksittäiseksi tiedostoksi). Yhdistetyn sertifikaatti tiedoston tiedostopolku voidaan lisätä HAProxy:n asetustiedostoon ja Ampache-sovelluskontille on mahdollistettu suojatun yhteyden käyttäminen.

6 SOVELLUSKONTIT

6.1 Docker Engine (CE)

Docker Engine -ohjelmasta asennetaan yhteisön ylläpitämä versio Ubuntulle. Yhteisön ylläpitämää versiota Dockerista tuetaan seitsemän kuukauden ajan. Viimeisimmät korjaukset ja päivitykset tulevat ainoastaan tuen piirissä olevalle vakaalle versiolle (Docker, 2019e). Docker Engine -ohjelman asennuksessa sovelletaan dokumentaatiosta löytyviä asennusohjeita Ubuntulle (Docker, 2019b).

Asennuksessa hyödynnetään paketinhallintaa ja alustavasti sinne lisätään Dockerin julkinen avain. Avaimen lisäämiseen käytetään cURL-ohjelmistotyökalua ja sen lisäksi asennetaan suojatun yhteyden (HTTPS) hyödyntämiseen tarvittavat paketit paketinhallinnalla. Pakettien asennukseen löytyy komennot liitteistä (ks. Liite 1, Docker Engine: Perustyökalut ja paketit).

Komennoissa kenoviiva merkitsee rivinvaihtoa komentorivillä, syötettäessä asennuskomentoa paketit ovat erotettuina toisistaan ainoastaan välilyönnillä. Pakettiin ”ca-certificates” sisältyy Mozillan selaimessa käytetyt auktorisoidut sertifikaattien allekirjoittajat (Debian, 2019). TLS-salausprotokollaan perustuvat ohjelmistot voivat varmistaa salatun yhteyden luotettavuuden vertaillessaan sertifikaateissa olevia julkisia avaimia (Kassner, 2008). Kaikkia paketteja ei välttämättä asenneta uudestaan, mikäli ne on aikaisemmin asennettu.

Seuraavalla komennolla voidaan lisätä Dockerin virallinen GPG-avain luotettujen listalle:

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

Syötetyssä curl-komennossa käytetään liipaisimia (parametrejä). Liipaisimella “f” piilotetaan etäpalvelimen palauttamat virheilmoitukset yhteyden aikana. Liipaisimella “sS” piilotetaan prosessin edistymistä näyttävä mittari, mutta näytetään virheilmoitus toimenpiteen loppuun viennin epäonnistuessa. Liipaisin “L”

uudelleen lähettää kutsun korjattuun kohteeseen, mikäli alkuperäinen etäpalvelin ilmoittaa haettavan sisällön siirtyneen. (Die, 2019.)

Dockerilta noudetun julkisen avaimen jälkeen voidaan vertailla sormenjälkeä dokumentaatioon. Dokumentaatiossa on lisättynä voimassa oleva luotettu sormenjälki, jonka viimeisen kahdeksan merkin perusteella voidaan tehdä haku. (ks. Liite 1, Docker Engine: Sormenjäljen tarkastus).

Mikäli haun perusteella saatu ulostulo ja dokumentaatio pitävät paikkaansa, voidaan asennusta jatkaa. Huomioitavaa sormenjäljen tarkastuksessa on sertifikaatin voimassa oleva luovutusaika. Tulevaisuudessa voimassa oleva sormenjälki voi muuttua ja vertauskohde pitää varmistaa dokumentaatiosta.

Docker Engine -ohjelmasta asennetaan viimeisin vakaa julkaisu versio. Dockerin arkisto voidaan lisätä Ubuntun paketinhallintaan liitteissä olevalla komennolla (ks. Liite 1, Docker Engine: Arkiston lisäys). Komennossa arkiston lisäämiseen hyödynnetään ympäristömuuttujia ja paketinhallintaan lisätään Dockerin ohjelmistoversio 64-bittisestä arkkitehtuurista.

Docker Engine -ohjelmasta on saatavilla myös versioita muille arkkitehtuureille. Muuttujalla “$(lsb_release -cs)” haetaan automaattisesti käytössä olevan käyttöjärjestelmän jakeluversion nimi. Esimerkiksi Ubuntu Server 18.10 -jakeluversion nimi on “cosmic”. Arkiston lisäämisen jälkeen Docker Engine -ohjelma ja sen työkalut voidaan asentaa. (ks. Liite 1, Docker Engine: Arkiston päivitys ja Docker-työkalujen asennus).

Pakettien asennuksen jälkeen voidaan suorittaa ohjelman testaus siihen soveltuvalla sovelluskontilla, jonka voi käynnistää esimerkki komennolla (ks. Liite 1, Docker Engine: Testaus sovelluskontti). Komennon ajaminen käynnistää Dockerin palveluprosessin ja noutaa verkkoarkistosta viimeisimmän version “hello-world” sovelluskontista ja suorittaa sen. Sovelluskontin suorittaminen tulostaa komentoriville viestin onnistuneesta asennuksesta. Docker Engine -ohjelma on nyt valmis sovelluskonttien luomiseen ja käyttämiseen.

6.2 Dockerfile: Sovelluskontin määrittely

Dockerfile on konfigurointitiedosto, joka määrittelee ympäristön sovelluskontille. Sovelluskontin sisällä olevat resurssit, kuten verkkolaitteet ja tallennusmedian ajurit virtualisoidaan ja eriytetään muusta käyttöjärjestelmästä. Siksi esimerkiksi verkkoyhteyksiä varten määritellään sovelluskontin sisäiset ja ulkoiset porttiohjaukset. Määrittelyssä ohjeistetaan myös sovelluksen toimintaan tarvittavien tiedostojen kopiointi sovelluskonttiin. (Docker, 2019c.)

Konfigurointitiedoston luomisen ja määrittelyn jälkeen voidaan sovelluskontin levykuva luoda Dockerin työkaluilla. Luomisen aikana sovellusta varten noudetaan perustaksi määritelty levykuva, esimerkiksi riisuttu versio Debian-käyttöjärjestelmästä. Perustan päälle asennetaan tarvittavat paketit ja kopioidaan sovelluksen tarvitsemat tiedostot.

Sovelluskonttien konfigurointitiedostoa voidaan tavallaan ajatella ruuan valmistusohjeena, kohta kohdalta herkulliseen lopputulokseen. Konfigurointitiedoston alustakohtaisista komennoista on lisättynä liitteisiin suppea esittely (ks. Liite 2, Dockerfile-komennot).

Konfigurointitiedoston sisällön luomiseen löytyy parhaat käytännöt Dockerin dokumentaatiosta. Konfiguroinnin aikana kannattaa lähtökohtaisesti ottaa huomioon sovelluskontin “lyhytaikaisuus”. Lyhytaikaisuudella tarkoitetaan mahdollisuutta sovelluskontin tuhoamiseen ja levykuvan uudelleen rakennukseen mahdollisimman pienellä määrällä uudelleen konfigurointia. Esimerkiksi käytettävistä sovelluksista voi tulla uusia versioita lyhyellä aikavälillä, jolloin sovelluskontteja voidaan joutua rakentamaan uudestaan useampaan otteeseen. (Docker, 2019f.)

6.3 Ampache

Ampache on avoimen lähdekoodin ohjelmisto, joka perustuu multimediasisällön suoratoistoon (engl. streaming) tietoverkoissa. Suoratoistolla esimerkiksi paikalliseen verkkotallennusjärjestelmään tallennettua mediaa voidaan toistaa päätelaitteella tietoverkon ylitse.

Ampache-sovelluksen lähdekoodi ja asennusohjeet löytyvät ohjelmakehitysprojektin versiohallintaan ja lähdekoodin julkaisuun tarkoitetulta GitHub-verkkosivulta (Ampache, 2018). Viimeisin vakaaksi määritelty versio löytyy “master” kehityshaarasta. Dockerfile-konfigurointitiedoston luomiseen voidaan hyödyntää yhteisön luomaa pohjaa, joka on viimeksi päivitetty kaksi vuotta sitten (Docker Hub, 2019a). Konfigurointitiedoston luonnissa päivitetään käytettävät ohjelmistot ja perustana toimiva levykuva uudempaan versioon, sekä muokataan asennus menettelyä. Sovelluskontin lähdekoodi ja käynnistys skripti on lisättynä liitteenä. (ks. Liite 3, Ampache: Dockerfile.)

6.3.1 Dockerfile-konfigurointitiedosto

Alkuperäisessä sovelluskontin versiossa hyödynnetään perustana Ubuntu 14.04 -käyttöjärjestelmän levykuvaa ja käytettävissä on useampia versioita. Perustana käytettävä levykuva versio voidaan valita tagin perusteella Docker Hub -palvelusta. Uudessa sovelluskontin versiossa perustaksi asetetaan viimeisin pitkäaikaisen tuen piirissä oleva Ubuntu-käyttöjärjestelmän jakeluversio käyttämällä tagia “ubuntu:latest”. (ks. Liite 3, Levykuva lähteen määrittely.)

Levykuvan jakeluversiota vaihtamalla saadaan uusimmat päivitetyt versiot sovelluksen vaatimista paketeista. Yhteisön luomassa sovelluskontin pohjassa käytettiin esimerkiksi verkkopalvelimien dynaamisten sivustojen luontiin tarkoitetusta ohjelmointikielestä vanhempaa PHP5-ohjelmistoversiota. Ubuntun uusimmassa LTS-jakeluversiossa viimeisin vakaaksi todettu ohjelmistoversio on PHP7. Myös PHP:n moduuleista löytyy uusimmat versiot.

Yhteisön luomassa sovelluskontin pohjassa asennetaan erikseen sovelluksen tarvitsemat koodekit. Koodekit tarvitaan mediatiedostojen muunnokseen suoralähetyksen aikana, esimerkiksi musiikkitiedoston bittivirran vähentämiseen hitaammille tietoverkkoyhteyksille. Halutessaan mediatiedostojen muuntamiseen tarvittava ffmpeg-sovellus ja koodekit voidaan asentaa manuaalisesti. Uudempaan Ubuntu-jakeluversion arkistoon on lisättynä suoraan ffmpeg-paketti, jolloin tarvittavat koodekit voidaan asentaa paketinhallinnasta.

Ampache-sovelluksen verkkosivun julkaisuun käytetään Apachen HTTP-palvelinohjelmaa, kuten alkuperäisessä yhteisön luomassa sovelluskontin pohjassa. Pakettien asentaminen Ubuntun arkistosta löytyy liitteen lähdekoodista (ks. Liite 3, Ampachen vaatimat paketit.)

Ampachen lähdekoodi voidaan ladata GitHub-palvelusta suoraan käyttäen git-ohjelmatyökalulla, joka asennettiin paketinhallinnasta Ampache-sovelluksen vaatimien pakettien mukana. Sovelluskonttiin ladataan viimeisin vakaa versio “master” kehityshaarasta väliaikaiseen kansioon. Työkalulla ladattu tiedosto on pakattuna ja se puretaan Apachen verkkojulkaisujen oletuskansion juureen. (ks. Liite 3, Ampachen lähdekoodin nouto; Ampachen tiedostot.)

Ampachen toiminta on riippuvainen myös useammasta erillisestä paketista, joita ei välttämättä asenneta käyttöjärjestelmän paketinhallinnasta suoraan. Composer on PHP-riippuvuuksien hallintaan luotu työkalu. Ampachen kehittäjät ovat luoneet sovellusprojektilleen riippuvuudet, jotka voidaan asentaa Composer:illa.

Konfigurointitiedoston komennoissa ladataan Composer-ohjelmatyökalu ja verrataan ladatun ohjelmatyökalun sormenjälkeä kehittäjän avaimeen. Sormenjäljen täsmätessä asennusta voidaan jatkaa normaalisti. (ks. Liite 3, Composer: Ampachen riippuvuudet.)

Composer voidaan suorittaa asentamisen ja tarkastamisen jälkeen Ampache-sovelluksen asennuskansiossa (ks. Liite 3, Composer: Suoritus). Riippuvuuksien asentamisen jälkeen komennossa annetaan omistusoikeudet Apachen ”www-data” oletuskäyttäjälle. Omistusoikeuden myöntäminen antaa verkkopalvelin ohjelmistolle oikeudet lukea ja suorittaa kansioissa olevia tiedostoja.

Sovelluskontista ulospäin näkyvät tiedostopolut voidaan myös määritellä. Esimerkiksi Ampachea varten määritellään ulospäin näkyvät tiedostopolut mediatiedostojen linkittämiseen ja asetustiedostojen säilyttämiseen. (ks. Liite 3, Ulkoiset tiedostopolut.)

Sovelluskontin käynnistämiseen hyödynnetään erillistä skriptiä (ks. Liite 3, Ampache: Sovelluskontin käynnistys skripti). Käynnistyksessä suoritetaan Apachen HTTP-palvelinohjelmisto etusijalla, estäen sovelluskontin sulkeutuminen automaattisesti skriptin suorittamisen päätyttyä. Alkuperäisessä yhteisön versiossa ajettiin etusijalla tietokanta palvelimen prosessia. Sovelluskontin päivitetyssä versiossa käytetään sovelluskontista erillään olevaa ulkopuolista tietokantaa, jolloin käynnistysprosessia muokattiin erilaiseksi. (ks. Liite 3, Ampachen käynnistys.)

6.3.2 Apachen virtuaalipalvelin (VHOST)

Ampache-sovelluksen verkkosivut julkaistaan Apachen HTTP-palvelinohjelman avulla ja sen toimintaa varten luodaan konfiguraatiotiedosto, joka määrittelee virtuaalipalvelimen (VHOST). Virtuaalipalvelimen käytettäväksi sidotaan portti 80 ja kuunneltavaksi osoitteeksi asetetaan tähtimerkki (*), jolloin määriteltyyn porttiin tulevaa HTTP-liikennettä kuunnellaan osoitteesta riippumatta. Hyväksytty isäntänimi yhteyspyynnöille määritellään “ServerName” asetuksella. Isäntänimeä verrataan HTTP-yhteyspyynnön aikana. Isäntänimeksi voidaan asettaa palvelimeen yhdistetty verkkotunnus tai IP-osoite. Asetuksella “ServerAlias” viitataan peitenimeen (engl. alias). Peitenimen avulla voidaan yhteyspyyntöjen varalta lisätä hyväksyttyihin isäntänimiin ylimääräisiä verkkotunnuksia tai IP-osoitteita. (ks. Liite 4, Konfiguraatiot: Apache VHOST.)

Virtuaalipalvelimen juurikansioksi (DocumentRoot) asetetaan Ampache-sovelluksen asennustiedostot sisältävä kansio. Hakemiston (Directory) määrityksissä voidaan asettaa sallitut direktiivit kansioille. Direktiivillä “AllowOverride All” voidaan sallia kansioihin sisältyvien “.htaccess” tiedostojen asetusmuutokset esi-

merkiksi symbolisten linkkien seuraamiseksi. Ampachessa ominaisuutta hyödynnetään esimerkiksi URL-osoitteiden siistimiseen. (ks. Liite 4, Konfiguraatiot: Apache VHOST.)

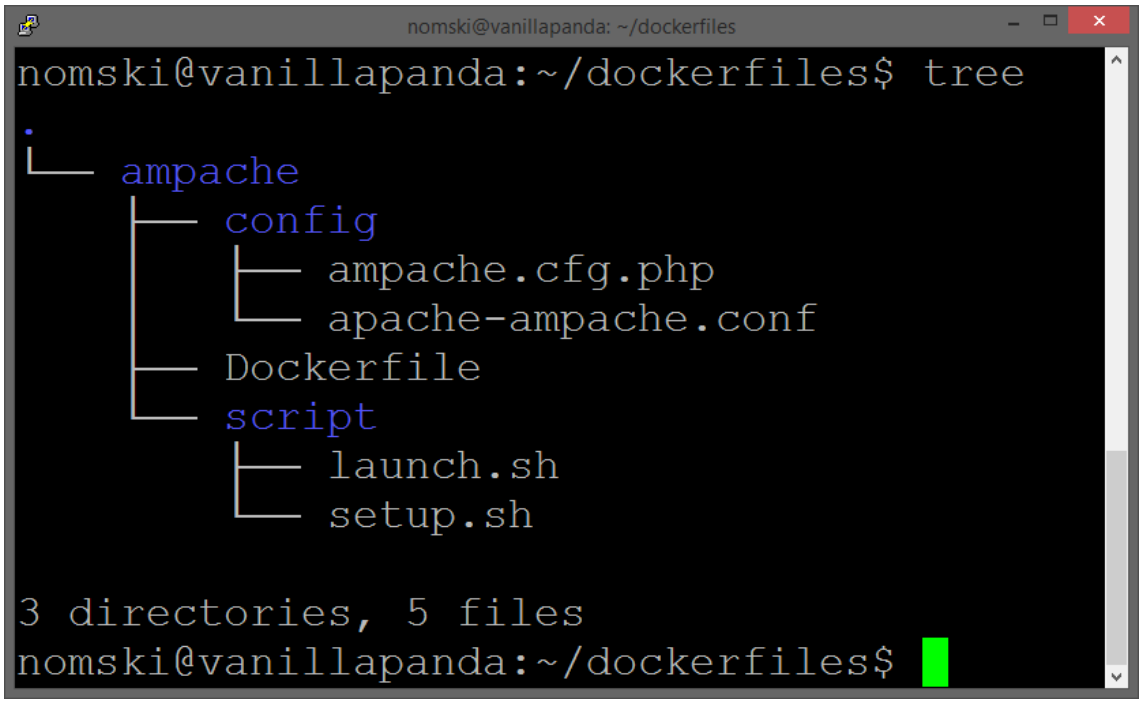
Luotu VHOST-konfigurointitiedosto lisätään Apachen saatavilla olevien verkkosivujen kansioon. Lisätystä konfiguraatiotiedostosta luodaan symbolinen linkki aktiivisten verkkosivujen kansioon. Ennen symbolisen linkin luomista poistetaan aktiivisten verkkosivujen kansiosta muut symboliset linkit, jolloin julkaistaan ainoastaan Ampache-sovelluksen verkkosivut.

Symbolisten linkkien käyttäminen on hyödyllistä, sillä tarvittaessa muokattavat konfigurointitiedostot sijaitsevat yhdessä paikassa. Symbolinen linkki viittaa aina alkuperäiseen tiedostoon tai sijaintiin, jolloin tiedostoon tehdyt muutokset näkyvät myös linkatussa sijainnissa.

Apachen URL-osoitteiden uudelleenkirjoituksen mahdollistava moduuli aktivoidaan, jolloin osoitteiden muuttaminen lennosta on tarvittaessa mahdollista (Apache, 2019). VHOST-konfigurointitiedoston lisääminen sovelluskonttiin määritellään Dockerfile-konfigurointitiedostossa. (ks. Liite 3, Apache2: konfiguraatiotiedosto.)

6.3.3 Ampache-sovelluskontin levykuva

Levykuvan luonnissa kannattaa huomioida sovelluskontin lähdekoodissa määritellyt tiedostopolut tiedostoineen, joiden oletetaan löytyvän levykuvan rakennuksen aikana. Käytettäessä työssä olevia esimerkkejä ja lähdekoodia, tiedostopuun oletetaan näyttävän seuraavalta:



KUVA 12. Ampache-sovelluskontin tiedostopuu

Ampache kansio toimii juurikansiona. Juurikansiossa sijaitsee Dockerfile-konfigurointitiedosto, sekä kansiot “config” ja “script” tiedostoineen. Kuvassa näkyvä “ampache.cfg.php” asetustiedosto ei ole välttämätön, mikäli haluaa tehdä Ampache-sovelluksen ohjatun asennuksen selaimessa. Muokatussa Dockerfile-konfigurointitiedostossa Ampache-sovelluksen asetustiedoston lisäämisen on oletuksena kommentoituna, jolloin sitä ei automaattisesti lisätä sovelluskonttiin levykuvan luonnin aikana.

Halutessaan oman muokatun asetustiedoston voi lisätä sovelluskonttiin ottamalla lähdekoodista kommentointiin viittaavan risuaita (#) merkin komennon edestä pois. Muokattavaksi soveltuva Ampache-sovelluksen asetustiedosto löytyy kehitysprojektin verkkosivulta GitHub-palvelusta (Ampache, 2018).

Tiedostojen ollessa oikeissa sijainneissaan, Dockerin työkaluilla voidaan suorittaa komento sovelluskontin levykuvan luomiseksi Ampachen juurikansiossa. Levykuvan luonnista löytyy esimerkki komento liitteistä. (ks. Liite 1, Docker Engine: Ampache levykuvan luonti.)

Esimerkissä levykuvalle luodaan nimellä “ampache-ctr” tunniste, jolloin sen tunnistaminen useamman levykuvan keskeltä on helpompaa. Komennon lopussa oleva piste viittaa komennon suoritukseen käytettävää tiedostopolkua, jossa käyttäjä sijaitsee kyseisellä hetkellä.

Ensimmäisen levykuvan luonnissa voi kulua pidemmän aikaa riippuen esimerkiksi ladattavien tiedostojen määrästä. Sovelluskontin uudelleen luomiseen käytetään rakennuksen aikana työkalun tekemiä välillisiä levykuvia. Välillistä levykuvaa voidaan hyödyntää “välimuistina” mikäli suoritettava komento tai levykuvan sisältö ei ole muuttunut.

Dockerin lataamat ja luodut levykuvat saavat erillisen yksilöivän levykuva tunnisteen, vaikka niiden luomisessa käytettäisiin yhtenevää tunnistetta. Paikallisesti tallennettujen levykuvien listan voi tarkistaa tarvittaessa komennon avulla. (ks. Liite 1, Docker Engine: Levykuvien listaus.)

Listassa voi näkyä useampia levykuvia, joilla ei ole tunnistetta tai arkistoa. Kyseiset levykuvat ovat sovelluskontin luonnissa käytettäviä välillisiä levykuvia tai samalla tunnisteella tehtyjä aikaisempia versioita. Lisäämällä komennon perään levykuvan tunnisteen, voi tarkastella lähemmin tunnisteen perusteella tehtyjä versioita.

Sovelluskontin voi käynnistää luodun levykuvan pohjalta. Aiempaan testaamiseen käytettyyn “hello-world” levykuvaan verrattuna Ampache tarvitsee useamman argumentin toimiakseen suunnitellusti. Liitteissä on esimerkki Ampache-sovelluskontin käynnistämisestä luodun levykuvan pohjalta. (ks. Liite 1, Docker Engine: Ampache-sovelluskontin käynnistys.)

Sovelluskontin käynnistyksessä asetetaan parametreissä ohjauksia sovelluskontin ulkopuolella ja sisäpuolella oleviin ominaisuuksiin. Sovelluskontin ulkopuoliset muuttujat ovat kaksoispisteen vasemmalla puolella ja sisäiset oikealla puolella. Erottelun voi huomata esimerkiksi sovelluskontin portteja määriteltäessä.

Ampache-sovelluskontin käynnistyksessä määritellään “--name” parametrissä käynnissä olevan sovelluskontin nimi. Nimen avulla voidaan tunnistaa käynnissä olevat sovelluskontit. Sovelluskontin “nimi” on erillinen levykuvan luonnissa tehtyyn tunnisteeseen verrattuna.

Sovelluskontille voidaan esitellä ulkoinen tiedostopolku “--volume” parametrillä. Ampachen tapauksessa avataan ulkoinen tiedostopolku mediatiedostoihin, jotka sijaitsevat sovelluskontin ulkopuolella.

Sovelluskontti kohtaiset porttiohjaukset voidaan määritellä “-p” parametrillä. Sovelluskontti suoritetaan taustalla irrallisena (engl. detached) prosessina käytettäessä “-d” parametriä. Etusijalla suoritettaessa sovelluskontin sisäinen komentorivi on käytettävissä käynnistämisen jälkeen.

Sovelluskontin käynnistämisen jälkeen Ampachen ohjatun asennuksen voi suorittaa menemällä selaimella palvelimen osoitteeseen. Esimerkissä sovelluskontin ulkopuolinen portti ohjataan testauksessa porttiin 8080, joka ei ole oletus portti HTTP-liikenteelle. Testiasennuksessa voidaan silloin mennä osoitteeseen http://192.168.50.3:8080/install.php joka viittaa palvelimella sijaitsevaan Ampache-sovelluskonttiin.

Ohjatussa asennuksessa syötetään tietokannan yhteystiedot ja Ampachen käyttämä käyttäjätunnus ja salasana. Tietokannan yhteyteen tarvitaan IP-osoite MariaDB-sovelluskontista. Aktiivisen sovelluskontin IP-osoitteen noutoon on liitteissä esimerkki komento (ks. Liite 1, Docker Engine: MariaDB-sovelluskontin IP-osoitteen noutaminen). Mediatiedostojen enkoodaukseen voidaan käyttää asennuksessa ehdotettua ffmpeg-sovelluksen pohjaa. Lopulta tehdään erillinen käyttäjätunnus ja salasana Ampache-sovelluksen ominaisuuksien käyttämistä varten.

Ensikertaa kirjauduttuessa sovellukseen ehdotetaan mediatiedostojen katalogin lisäämistä kokoelmiin. Katalogi voidaan lisätä sovelluskontin ulkopuolisesta tiedostopolusta asettamalla tiedostopoluksi /media. Sovelluskontin näkemät ulkopuoliset tiedostot perustuvat sovelluskontin käynnistämisen yhteydessä määriteltyihin parametreihin.

6.4 MariaDB

MariaDB on avoimeen lähdekoodiin perustuva, yhteisölähtöinen kehityshaara MySQL-tietokannan hallintajärjestelmästä. Tietokannat ovat hyödyllisiä jäsennellyn datan hallinnassa ja käyttämisessä. Useat sovellukset esimerkiksi tallentavat tietokantaan luomansa datan jäsennellysti.

Ampache-sovelluskontti luotiin ja rakennettiin yksityiskohtaisemmin ja sen rinnalle tarvitaan tietokanta. Ampache lukee esimerkiksi mediatiedostojen skannauksesta saatavan metatiedon ja lisää sen tietokantaan. Tietokannasta tiedon lukeminen ja näyttäminen on nopeampaa verrattuna mediatiedostojen uudelleen skannaamiseen.

MariaDB:n sovelluskontista hyödynnetään virallista kehittäjäyhteisön tarjoamaa levykuva versiota (Docker Hub, 2019b). Tietokannan luominen ja tarjoaminen sovellusten käytettäväksi valmiista levykuvasta on hyvä esimerkki käytännöllisemmästä lähestymistavasta testaus tai kehitys ympäristöön, jossa muutokset ovat nopeita.

Tietokannan hallintasovellukseen ja tietokantaan tarvitaan pääkäyttäjä (root), sekä uniikki salasana. Pääkäyttäjän lisäksi tietokantaan luodaan normaali käyttäjä Ampache-sovellusta varten. MariaDB:n levykuvan luonnissa voidaan käyttää ympäristömuuttujia käyttäjätietojen lisäämiseen. Menetelmää ei suositella, sillä ympäristömuuttujien kautta lisättynä arkaluontoiset tiedot ovat näkyvissä selkokielisenä. Huomattavasti parempi käytäntö on käyttää Dockerin salaustoimintoja (secrets). Ajan puutteessa alustava versio luodaan ns. nopeasti ja likaisesti, Ampache-sovelluksen testaamiseen ja toiminnallisuuden toteamiseen.

Dockerfile-konfigurointitiedostoa hyödynnetään ympäristömuuttujien syöttämiseen ja samalla luodaan alustava pohja tulevaisuuden varalle. Virallisen levykuvan päälle voidaan esimerkiksi asentaa selaimessa toimiva graafinen käyttöympäristö tietokantojen hallintaan. Graafinen käyttöympäristö helpottaa useamman tietokannan ylläpitoa. MariaDB:n Dockerfile-konfigurointitiedosto on lisättynä liitteisiin. (ks. Liite 5, MariaDB: Dockerfile.)

Sovelluskontin perustavaksi levykuvaksi on valittuna MariaDB:n viimeisin versio tagilla “mariadb:latest”. Ympäristömuuttujissa määritellään tietokannan pääkäyttäjän salasana satunnaisesti generoiduksi. Tietokantaan lisätään uusi käyttäjä ja salasana Ampache-sovellukselle. Lopulta levykuva voidaan luoda Dockerfile-konfigurointitiedoston sisältävästä juurikansiosta. Liitteistä löytyy esimerkki komento levykuvan luontiin ja sovelluskontin tunnisteena käytetään ”mariadb-ctr”. (ks. Liite 1, Docker Engine: MariaDB-sovelluksen levykuvan luonti.)

Luodun levykuvan käynnistyksen parametreihin lisätään ulkoinen tiedostopolku. Ulkoiseen tiedostopolkuun tallennetaan tietokanta ja sen sisältämä data, jolloin sovelluskontin uudelleen rakentamisen aikana ei menetetä olemassa olevia tietokantoja. Sovelluskontin käynnistyksessä määritelty tiedostopolku pitää olla valmiiksi luotuna. Liitteiden esimerkki komennossa luodaan alustavasti tiedostopolku, käynnistetään sovelluskontti oikeilla parametreillä ja viitataan siihen. (ks. Liite 1, Docker Engine: MariaDB-sovelluskontin käynnistys.)

Käynnistämisen jälkeen sovelluskontin tiedoista tarvitsee hakea sen käytössä oleva IP-osoite. IP-osoitteen avulla Ampache-sovelluskontti voi yhdistää tietokantaan ennalta luodulla käyttäjätunnuksella ja salasanalla. Sovelluskontin IP-osoitteen voi noutaa Dockerin inspect-komennolla (ks. Liite 1, Docker Engine: MariaDB-sovelluskontin IP-osoitteen noutaminen). Sovelluskonttien parametrit renderöidään JSON-formaatissa, jolloin sen tiedoista voidaan parsia haettava tieto helposti.

6.4.1 Dockerin salausominaisuus teoriassa

Dockerissa voidaan siirtää arkaluontoista tietoa isäntäkoneen ja sovelluskontin välillä hyödyntäen salausominaisuuksia (secrets). Menetelmän toiminnasta löytyy esimerkkejä Dockerin virallisesta dokumentaatiosta (Docker, 2019h). Hyödynnettäessä salausominaisuutta Dockerin käyttämä järjestelmä asetetaan parveksi (swarm). Parvessa voidaan halutessaan klusteroida useampia palvelimia ja parantaa palveluiden saatavuutta. Yhden noodin sisältäväksi parveksi voidaan asettaa käytettävissä oleva palvelin. (ks. Liite 1, Docker Engine: Swarm-tilan alustus.)

Palvelin asetetaan oletuksena klusterin manageriksi ja asetuksien tekemisen aikana saadaan klusterille uniikki tunnus (engl. token). Tunnuksen avulla klusteriin voidaan liittää uusia palvelimia työläisiksi. Yksittäistä noodia käytettäessä tunnusta ei tarvita. Parven luontiin ja hallinnointiin löytyy syvennettyjä esimerkkejä Dockerin virallisesta dokumentaatiosta. (Docker, 2019i.)

Salausominaisuuksia voidaan käyttää hyödyksi parven alustamisen jälkeen. Salasanojen luomiseen voidaan käyttää esimerkiksi OpenSSL-ohjelman mukana tulevaa satunnaisgeneraattoria. Esimerkki komennoissa salasanoista tehdään 20 merkkisiä. Generoitu tai selkokielinen salasana kirjoitetaan salaisuuden sisältävään tiedostoon secrets-komennolla. Komennon lopussa oleva viiva “-” viittaa salaisuudeksi luettavaan tiedostoon. (ks. Liite 1, Docker Engine: Salasanojen generointi.)

Luomisen jälkeen ominaisuuden avulla tehtyjä salaisuuksia ei voida lukea selkokielisesti, sillä niiden sisältö salataan. Automaattista satunnaisesti generoitua salasanaa kannattaa käyttää esimerkiksi pääkäyttäjän salasanan luomiseen, mikäli pääkäyttäjän oikeuksia ei toiminnan kannalta tarvita.

7 POHDINTA

Opinnäytetyöhön sisältyy kirjava väripaletti tietotekniikkaan ja virtualisointiin liittyvistä käsiteltävistä asioista. Lähtökohtaisesti tarkoituksena oli työskennellä useamman tietotekniikkaan kuuluvan osa-alueen parissa. Työhön kuuluviin osa-alueisiin voidaan lukea esimerkiksi tietoverkkojen hallinnointi, palvelimien rakentaminen ja ylläpito.

Alkuperäisenä ajatuksena oli muuttaa olemassa oleva virtuaalikoneisiin perustuva palvelin käyttämään sovelluskontteja. Muutoksen myötä palvelimen resursseja voidaan hyödyntää tarkemmin sovelluskohtaisesti. Sovelluskonttien luomiseen ja hallintaan käytettävä virtualisointialusta on toiminnassa. Alustavan testauksen mukaan virtualisointialusta toimii odotetulla tavalla ja mahdollistaa uusien sovellusten kokeilemisen nopeasti.

Paikallinen tietoverkko rakennettiin loogisella tasolla uusiksi ja lähiverkot eriytettiin omiin osoiteavaruuksiin. Tulevaisuudessa verkkoon luodaan reitittimellä vyöhykkeisiin perustuvat alueet, jolloin verkkojen välistä liikennettä ei tarvitse ohjata pääsylista pohjaisesti. Verkon kokonaisrakenne tuntuu alustavasti toimivalta ja loogisella tasolla riittää kapasiteettia kasvuun.

Palvelimelle tehdyn porttiskannauksen perusteella ainoastaan palomuurien säännöissä määritellyt portit ovat avoinna ulkoverkkoon korjauksen jälkeen. Suojattu ulkoverkkoyhteys ja verkkotunnukselle haettu sertifikaatti testattiin. Lopullisen testituloksen perusteella suojauksen pitäisi olla kunnossa ja ulkoverkosta tuleville yhteyspyynnöille tehdyt säännöt toimivat halutulla tavalla.

Tulevaisuudessa ulkoverkkoon näkyvä verkkoliikenne voitaisiin ohjata VPN-tunnelin lävitse, jolloin palvelimen julkinen IP-osoite ei paljastuisi suoraan verkkotunnuksen perusteella tehdyssä haussa. Dynaamisen nimipalvelimen avulla tunnelin loppupäässä olevan yhteyspisteen IP-osoite saa vaihtua, sillä nimipalvelimelle tehtävä kirjaus päivitetään tietyin aikavälein.

Palvelimen hallintaan liittyviä sovelluksia ei nykyisessä kokoonpanossa vaihdettu sovelluskonteiksi. Hallintaan käytettävien sovellusten kriittisyys on toiminnan kannalta liian suuri. Testattaessa sovelluskontteja ensimmäistä kertaa tiettyjen osa-alueiden toiminnallisuus halutaan varmistaa.

Sovelluskonttien rakentamisesta käytettiin syvällisenä esimerkkinä median suoratoistoon käytettävää ohjelmaa. Nopeaan testaamiseen viittaavassa esimerkissä käytettiin tietokantaohjelmiston virallisia levykuvia ja syötettiin ainoastaan tarvittavat ympäristömuuttujat.

Sovelluskontteihin tarvitsee tehdä muutoksia tulevaisuudessa. Esimerkiksi median suoratoistoon käytettävän sovelluskontin rakennus vaiheisiin tarvitsee lisätä sovelluksen asetustiedostojen kopioiminen levykuvan luonnin aikana. Muutoksella uusien levykuva versioiden myötä ei tarvitse käydä ohjeistettua asennusta lävitse uudelleen.

Tietokantaohjelmiston sovelluskontin rakennuksessa nousi ongelmaksi esimerkiksi arkaluontoisen tiedon siirtäminen konttiin. Arkaluontoisen tiedon siirtämiseen soveltuvaa Dockerin salausominaisuutta pyrittiin hyödyntämään alkuperäisessä suunnitelmassa ja siitä jouduttiin luopumaan toistaiseksi.

Väliaikaisena ratkaisuna voidaan käyttää suoraan käyttöjärjestelmään asennettavaa tietokantaohjelmistoa ja hallinnoida tietokantoja sen välityksellä. Tietokannan salasanat luotaisiin manuaalisesti, kunnes sovelluskontti saadaan tulevissa versioissa toimimaan halutulla tavalla. Alustavasti korjaava suunta viittaa Dockerin swarm-tilaan liittyvien ominaisuuksien syvällisempään tutkimiseen ja oppimiseen.

Alustava pohja sovelluskonttien käyttämiseen virtuaalikoneiden sijaan on luotuna, tulevaisuudessa sitä voidaan laajentaa ja kehittää parempaan suuntaan. Mahdollisesti vaihtaa jopa erilaiseen tekniikkaan. Kotilaboratorion ideassa kiteytyy uusien asioiden oppiminen ja testaaminen.

LÄHTEET

Adeesh, F. 2017. Containers Deep Dive – LXC vs Docker. Robin Systems, Inc. Julkaistu 31.1.2017. Luettu 12.3.2019. https://robin.io/blog/containers-deep-dive-lxc-vs-docker-comparison/

Ampache, 2018. Github: Ampache Project. Luettu 8.4.2019. https://github.com/ampache/ampache/

Apache Software Foundation, 2019. Apache module mod_rewrite. Luettu 12.4.2019. https://httpd.apache.org/docs/current/mod/mod_rewrite.html

Badola V. 2015. Container virtualization: What makes it work so well? Cloud Academy Inc. Julkaistu 27.10.2015. Luettu 4.3.2019. https://cloudacademy.com/blog/container-virtualization/

Campbell S. & Jeronimo M. 2006. Introduction to Virtualization. PDF. Intel. Luettu 4.3.2019. https://software.intel.com/sites/default/files/m/d/4/1/d/8/An_Introduction_to_Virtualization.pdf

Debian, 2019. Package: ca-certificates. Päivitetty 1.10.2019. Luettu 2.4.2019. https://packages.debian.org/sid/ca-certificates

Die.net, 2019. Curl - Linux man page. Luettu 2.4.2019. https://linux.die.net/man/1/curl

Docker, 2019a. What is a Container? Luettu 4.3.2019. https://www.docker.com/resources/what-container

Docker, 2019b. Get Docker CE for Ubuntu. Luettu 18.3.2019. https://docs.docker.com/install/linux/docker-ce/ubuntu/

Docker, 2019c. Get Started, Part2: Containers. Luettu 18.3.2019. https://docs.docker.com/get-started/part2/

Docker, 2019d. Docker Hub. Luettu 18.3.2019. https://hub.docker.com/

Docker, 2019e. Which Docker Engine is right for you? Luettu 2.4.2019. https://www.docker.com/products/docker-engine

Docker, 2019f. Best practices for writing Dockerfiles. Luettu 3.4.2019. https://docs.docker.com/develop/develop-images/dockerfile_best-practices/

Docker, 2019g. Dockerfile reference. Luettu 12.4.2019. https://docs.docker.com/engine/reference/builder/

Docker, 2019h. Manage sensitive data with Docker secrets. Luettu 13.4.2019. https://docs.docker.com/engine/swarm/secrets/

Docker, 2019i. Run Docker Engine in swarm mode. Luettu 13.4.2019. https://docs.docker.com/engine/swarm/swarm-mode/

Docker Hub, 2019a. Docker container image for Ampache. Luettu 8.4.2019. https://hub.docker.com/r/ampache/ampache

Docker Hub, 2019b. Docker container image for MariaDB. Luettu 13.4.2019. https://hub.docker.com/_/mariadb

EFF, 2019. Certbot. Luettu 31.3.2019. https://certbot.eff.org/

HAProxy, 2019a. The reliable, high performance TCP/HTTP load balancer. Luettu 30.3.2019. https://www.haproxy.org/

HAProxy, 2019b. HAProxy Documentation. Päivitetty 11.2.2019. http://cbonte.github.io/haproxy-dconv/

Hiltunen, J. 2010. Palvelimen virtualisointi. Tietotekniikan koulutusohjelma. Lahden ammattikorkeakoulu. Opinnäytetyö.

Hoffman, C. 2017. Beginner Geek: How to create and use virtual machines. How-To Geek. Julkaistu 14.6.2017. Luettu 8.3.2019. https://www.howtogeek.com/196060/beginner-geek-how-to-create-and-use-virtual-machines/

IBM. 2012. Virtualization overview: Introduction to virtualization. Julkaistu 4.10.2012. Luettu 4.3.2019. https://www.ibm.com/support/knowledgecenter/en/POWER6/iphb2/iphb2virtoverview.htm

Intel. 2015. Container and KVM Virtualization for NFV. PDF. Luettu 15.3.2019. https://builders.intel.com/docs/container-and-kvm-virtualization-for-nfv.PDF

Joker.com, 2018. Dynamic DNS (DynDNS). Luettu 14.4.2019. https://joker.com/faq/content/11/427/en/what-is-dynamic-dns-dyndns.html

Kassner, M. 2008. SSL/TLS certificates: What you need to know. Julkaistu 26.8.2008. Luettu 2.4.2019. https://www.techrepublic.com/blog/data-center/ssl-tls-certificates-what-you-need-to-know/

Linux Foundation, 2019. Let’s Encrypt: Getting Started. Luettu 23.3.2019. https://letsencrypt.org/getting-started/

LXD Introduction. 2019. What’s LXD? Linux containers. Luettu 4.3.2019. https://linuxcontainers.org/lxd/introduction/

Mitchell, B. 2018. What is a Virtual LAN (VLAN)? Päivitetty 26.9.2018. Luettu 2.4.2019. https://www.lifewire.com/virtual-local-area-network-817357

Qualys, 2019. SSL server test. Luettu 15.4.2019. https://www.ssllabs.com/ssltest/

Ubuntu Community, 2014. UbuntuDesktopLVM. Päivitetty 11.6.2014. Luettu 20.3.2019. https://help.ubuntu.com/community/UbuntuDesktopLVM

Ubuntu Community, 2018. Installation/MinimalCD. Päivitetty 7.10.2018. Luettu 20.3.2019. https://help.ubuntu.com/community/Installation/MinimalCD

Webmin, 2018. Installing on Debian. Päivitetty 30.11.2018. Luettu 30.3.2019. http://www.webmin.com/deb.html

Zenmap, 2019. Nmap Security Scanner GUI. Luettu 15.4.2019. https://nmap.org/zenmap/

Zuba, M. 2015. Hardening HAProxy for an A+ rating. Julkaistu 22.7.2015. Luettu 14.4.2019. http://www.mattzuba.com/2015/07/hardening-haproxy-for-an-a-rating/

LIITTEET

Liite 1. Sovellusten komennot

NFS

NFS-työkalujen asennus

```
sudo apt install nfs-common
```

WEBMIN

Arkiston lisäys Ubuntun paketinhallintaan

```
echo -e "\n# Webmin repository \n deb \
https://download.webmin.com/download/repository sarge \
contrib" | sudo tee -a /etc/apt/sources.list
```

Kehittäjän GPG-avaimen lisäys luotettujen listalle

```
sudo apt install gnupg
sudo wget -P /root/ http://www.webmin.com/jcameron-key.asc
sudo apt-key add /root/jcameron-key.asc
```

Arkiston päivitys ja Webminin asennus

```
sudo apt update
sudo apt install apt-transport-https
sudo apt install webmin
```

(jatkuu)

Asennus paketinhallinnasta

```
sudo apt install haproxy
```

Asetustiedoston tarkastus

```
sudo haproxy -f /etc/haproxy/haproxy.cfg -c
```

CERTBOT

Perustyökalut ja arkiston lisäys

```
sudo apt install software-properties-common
sudo apt-add-repository universe
sudo apt-add-repository ppa:certbot/certbot
```

Arkiston päivitys ja ohjelman asennus

```
sudo apt update && sudo apt install certbot
```

DDClient-ohjelman asennus

```
sudo apt install ddclient
```

Sertifikaatin nouto itsenäisesti

Sertifikaatin noutaminen domain osoitteelle, vaihda “<domain>” kohtaan käyttämäsi domain nimi.

```
sudo certbot certonly --standalone --preferred-challenges \
http --http-01-port 80 -d <domain>
```

Sertifikaatin muunnos yksittäiseksi tiedostoksi 3 (5)

```
sudo mkdir /etc/haproxy/certs
sudo chmod -R go-rwx /etc/haproxy/certs
```

Lisää ”<verkkotunnus>” muuttujaan sertifikaatin noutamiseen käytetty verkkotunnus. Muuttujan avulla haetaan oikeasta tiedostopolusta verkkotunnuksen perusteella sertifikaatin tiedostot.

```
DOMAIN='<verkkotunnus>' sudo -E bash -c 'cat \
/etc/letsencrypt/live/$DOMAIN/fullchain.pem \
/etc/letsencrypt/live/$DOMAIN/privkey.pem > \
/etc/haproxy/certs/$DOMAIN.pem'
```

DOCKER ENGINE

Perustyökalut ja paketit

```
sudo apt install software-properties-common \
apt-transport-https \
ca-certificates gnupg-agent curl
```

Arkiston lisäys

```
sudo add-apt-repository "deb [arch=amd64] \
https://download.docker.com/linux/ubuntu \
$(lsb_release -cs) stable"
```

Sormenjäljen tarkastus

```
sudo apt-key fingerprint 0EBFCD88
```

Arkiston päivitys ja Docker-työkalujen asennus

```
sudo apt update && sudo apt install docker-ce \
docker-ce-cli containerd.io
```

Testaus sovelluskontti 4 (5)

```
sudo docker run hello-world
```

Ampache-sovelluksen levykuvan luonti

```
sudo docker build --tag=ampache-ctr .
```

Levykuvien listaus

```
sudo docker images <tunniste>
```

Ampache-sovelluskontin käynnistys

Sovelluskontille voidaan esitellä ulkoinen tiedostopolku ”<tiedostopolku>” muuttujassa, joka lisää viittauksen sovelluskontin sisällä olevaan /media tiedostopolkuun. Muuttujalla ”<portti>” voidaan määrittää sovelluskontille ulkoinen portti verkkoyhteyksiä varten.

```
sudo docker run --name ampache-kontti --volume \
<tiedostopolku>:/media:ro -p <portti>:80 -d ampache-ctr
```

Swarm-tilan alustus

```
sudo docker init
```

Salasanojen generointi

Docker salaisuuden luominen pääkäyttäjälle satunnaisesti generoidulla salasanalla:

```
sudo openssl rand -base64 20 | sudo docker secret create \
mariadb_root_password -
```

Vaihtoehtoisesti käyttäjä voidaan luoda selkokielisellä salasanalla (vaihda <salasana> lainausmerkkien sisällä).

```
echo "<salasana>" | sudo docker secret create \
mariadb_root_password -
```

Docker salaisuuden luominen Ampache käyttäjälle selkokielisestä tekstistä (vaihda <salasana> lainausmerkkien sisällä):

```
echo "<salasana>" | sudo docker secret create \
ampachedb_user_password -
```

MariaDB-sovelluksen levykuvan luonti

```
sudo docker build --tag=mariadb-ctr .
```

MariaDB-sovelluskontin käynnistys

```
sudo mkdir /mnt/data/mariadb
sudo docker run --name mariadb-kontti -v \
/mnt/data/mariadb:/var/lib/mysql -d mariadb-ctr
```

MariaDB-sovelluskontin IP-osoitteen noutaminen

Sovelluskonteista voidaan hakea IP-osoite suoraan seuraavalla komennolla. Komennon lopussa olevaan "<sovelluskontti>" parametriin vaihdetaan käynnissä olevan sovelluskontin tunnus.

```
docker inspect -f '{{range .NetworkSettings.Networks}} \
{{.IPAddress}}{{end}}' <sovelluskontti>
```

Liite 2. Dockerfile-komennot

Dockerfile-komennot pähkinänkuoressa 1 (2)

Tähän liitteeseen on koottuna supistetut versiot Dockerfile-konfigurointitiedostojen lähdekoodissa käytettävistä komennoista, jotka ohjeistavat levykuvan rakennuksen. Viitatut komennot ja niiden toiminnallisuus löytyy kokonaisuudessaan Dockerin dokumentaatiosta (Docker, 2019g.)

ADD *[--chown=<user>:<group>] <lähde> <kohde>*

Komennolla voidaan kopioida tai ladata uusia tiedostoja ja kansioita, tarvittaessa myös tietoverkossa sijaitsevista kohteista. Lisättävät tiedostot siirretään kohdepolkuun ja tarvittaessa voidaan määritellä tiedostojen omistaja(t) Linux-pohjaisissa sovelluskonteissa chown-komennolla.

CMD *<komento> <parametrit>*

Dockerfile-konfigurointitiedosto voi sisältää ainoastaan yhden CMD-komennon, jonka tarkoituksena on asettaa oletus käynnistys menetelmä suoritettavalle sovelluskontille. Käynnistyksessä voidaan käyttää yksittäistä komentoa, skriptiä tai suoritettavaa tiedostoa.

COPY *<lähde> <kohde>*

Komennolla kopioidaan uudet tiedostot ja kansiot määritellystä lähteestä ja ne lisätään sovelluskontissa olevaan kohteeseen.

(jatkuu)

EXPOSE *<portti> [<portti>/<protokolla>...]* 2 (2)

Komento määrittelee sovelluskontin sisäisesti kuuntelemat portit. Käytettäväksi protokollaksi voidaan määritellä UDP tai TCP. Mikäli protokollaa ei määritellä, käytetään oletuksena TCP-protokollaa.

FROM *<levykuva>[:tunniste]*

Komento määrittelee uuden rakennusvaiheen ja asettaa sovelluskontin rakentamiseen käytettävän lähde levykuvan. Validin Dockerfile-konfigurointitiedoston pitää alkaa FROM-komennolla.

RUN *<komento>*

Pohjalla olevan levykuvan päällä voidaan suorittaa sovelluskontin sisäisiä komentoja käyttämällä RUN-komentoa. Komennon suorituksen aikana tehdyt muutokset tallennetaan levykuvaan. Välillistä tallennettua levykuvaa käytetään seuraavaan vaiheeseen. Jonottamalla komentoja yhden RUN-komennon aikana voidaan pienentää välivaiheiden ja levykuvien määrää.

VOLUME [“/data”]

Komento luo ja merkitsee asennuspisteen sovelluskontista ulkoisesti pidettäville sijainneille. Komennon avulla voidaan luoda datalle pysyvä sijainti sovelluskontin ulkopuolelle.

Liite 3. Ampache: Dockerfile

Ampache: Dockerfile-konfigurointitiedosto 1 (2)

```
# Matti Hyttinen: Opinnäytetyö - Ampache DockerFile
# Perustuu seuraaviin lähteisiin:
# https://github.com/ampache/ampache
# https://github.com/ampache/ampache-docker

# Levykuva lähteen määrittely
FROM ubuntu:latest

# Arkiston päivitys ja viimeisimpien pakettien asennus
RUN apt update
RUN apt -y upgrade

# Ampachen vaatimat paketit: työkalut ja median \ enkooderit/dekoode-
rit (ffmpeg)

RUN DEBIAN_FRONTEND=noninteractive apt -y install git wget apache2 php
php-curl php-json php-mysql php-xml php-gd ffmpeg

# Ampachen lähdekoodin nouto (GitHub, "master" branch)
ADD https://github.com/ampache/ampache/archive/master.tar.gz \
/opt/ampache.tar.gz

# Ampachen tiedostot: Purku ja siirto Apachen kansioihin

RUN cd /opt/ && tar -xvzf ampache.tar.gz && mv ampache-master/*
/var/www

# Ampachen konfiguraation kopiointi
# ADD ./config/ampache.cfg.php /var/www/config/

# Composer: Ampachen riippuvuudet (Asennus ja hallinta)
RUN php -r "copy('https://getcomposer.org/installer', 'composer-
setup.php');"

RUN php -r "if (hash_file('sha384', 'composer-setup.php') ===
'48e3236262b34d30969dca3c37281b3b4bbe3221bda826ac6a9a62d6444cdb0dcd061
5698a5cbe587c3f0fe57a54d8f5') { echo 'Installer verified'; } else {
echo 'Installer corrupt'; unlink('composer-setup.php'); } echo
PHP_EOL;"

RUN php composer-setup.php && mv composer.phar /usr/local/bin/composer
RUN php -r "unlink('composer-setup.php');"

# Composer: Suoritus (Ajetaan Ampachen kansiossa)
# Lisätään rekursiivisesti www-data käyttäjälle oikeus kansioihin
RUN cd /var/www && composer install --prefer-source --no-interaction
&& chown -R www-data /var/www
```

(jatkuu)

```
# Apache2: Konfiguraatiotiedosto (kopiointi ja symbolisen linkki)
COPY ./config/apache-ampache.conf /etc/apache2/sites-available/
RUN rm -rf /etc/apache2/sites-enabled/*
RUN ln -s /etc/apache2/sites-available/apache-ampache.conf \
/etc/apache2/sites-enabled/
RUN a2enmod rewrite

# Ulkoiset tiedostopolut: Määritellään ulospäin näkyvät sijainnit.
VOLUME ["/media"]
VOLUME ["/var/www/config"]
VOLUME ["/var/www/themes"]

# Sovelluskontin portit: Avataan sisäinen portti 80 (HTTP).
EXPOSE 80

# Ampachen käynnistys: Kopioidaan sovelluskontin käynnistys skripti
COPY ./script/launch.sh /launch.sh
RUN chmod +x launch.sh

# Sovelluskontin käynnistyessä ajettava komento (skripti)
CMD ["./launch.sh"]
```

Ampache: Sovelluskontin käynnistys skripti (launch.sh)

```
#!/bin/bash
# Käynnistetään Apache2 palvelu etualalla (foreground), jolloin \ so-
velluskontti ei sulkeudu automaattisesti

apache2ctl -d /etc/apache2 -e info -DFOREGROUND
```

Liite 4. Konfiguraatiot

KONFIGURAATIOT 1 (2)

DDClient

Joker.com verkkotunnuksien välittäjän antamat asetukset kirjauksen päivittämiseen dynaamiseen nimipalvelimeen. Palveluntarjoaja antaa nimipalvelimen asetuksia muutettaessa käyttäjätunnuksen ja salasanan.

```
# ddclient.conf
# Verkkotunnuksen välittäjän DDClient asetustiedosto (Joker.com)

daemon=5m
use=web
web=svc.joker.com/nic/checkip
server=svc.joker.com/nic/update?
protocol=dyndns2
login=<käyttäjä>
password=<salasana>
host=<domain>
ssl=yes
```

Apache VHOST (apache-ampache.conf)

```
# Apache2: VHOST (Ampache) alidomainille ampache.nyeh.eu
<VirtualHost *:80>
            ServerName ampache.nyeh.eu
            ServerAlias ampache.nyeh.eu
            DocumentRoot /var/www
            ErrorLog ${APACHE_LOG_DIR}/error.log
            CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>

# Apache2: Juurikansion direktiivit
<Directory /var/www/>
    Options Indexes FollowSymLinks MultiViews
    AllowOverride All
    Order allow,deny
    Allow from all

    SetEnvIf Authorization "(.*)" HTTP_AUTHORIZATION=$1
</Directory>
```

(jatkuu)

HAProxy 2 (2)

```
# Salakirjoitusmenetelmät ja asetusten kovetus HTTPS liikenteelle
# perustuu lähteeseen:
# http://www.mattzuba.com/2015/07/hardening-haproxy-for-an-a-rating/

global
        maxconn 1024
        tune.ssl.default-dh-param 2048
        log /dev/log    local0
        log /dev/log    local1 notice
        chroot /var/lib/haproxy
        stats socket /run/haproxy/admin.sock mode 660 level admin ex-
pose-fd listeners
        stats timeout 30s
        user haproxy
        group haproxy
        daemon

        # Sallitut salakirjoitusmenetelmät
        ssl-default-bind-options no-sslv3 no-tls-tickets
        Ssl-default-bind-ciphers EECDH+AESGCM:EDH+AE
GCM:AES256+EECDH:AES256+EDH

defaults
        log     global
        mode    http
        option  httplog
        option  dontlognull
        option  forwardfor
        option  http-server-close

        timeout connect 5000
        timeout client  50000
        timeout server  50000

# Frontend (etupää): Asetukset HTTP ja HTTPS liikenteelle
frontend http-in
        # HTTP: Ohjaa portin 80 liikenne suojatulle
        bind *:80
        redirect scheme https code 301 if !{ ssl_fc }
frontend https-in
        # HTTPS: Käytetään Let's Encrypt sertifikaatteja
        bind *:443 ssl crt /etc/haproxy/certs/

        # Kovennetaan asetuksia suojatulle yhteydelle
        http-response set-header Strict-Transport-Security \ max-
age=31536000;\ includeSubdomains;\ preload
        http-response set-header X-Frame-Options DENY
        http-response set-header X-Content-Type-Options nosniff

        # Pääsylista URL perusteella (viittaus alidomainiin)
        acl ampache_web hdr_end(host) -i ampache.nyeh.eu
        use_backend ampache if ampache_web

# Backend (taustapää): asetukset ohjaukselle palvelimelle
backend ampache
        option httpclose
        option forwardfor
        server vanillapanda 127.0.0.1:8080 maxconn 32
```

Liite 5. MariaDB: Dockerfile

MariaDB: Dockerfile-konfigurointitiedosto

Lähdekoodissa on punaisella merkittynä salasanaan viittaava kohta, mikä tulee vaihtaa testausmielessä käynnistettävään sovelluskonttiin. Selkokielisen salasanan syöttäminen Dockerfile-konfigurointitiedoston ympäristömuuttujassa ei ole suositeltavaa.

```
# Matti Hyttinen: Opinnäytetyö - MariaDB DockerFile
# Perustuu koodipohjaan seuraavista lähteistä:
# https://hub.docker.com/_/mariadb
# https://mariadb.org/

# Levykuva lähteen määrittely
FROM mariadb:latest

# Ympäristömuuttujat: Määritellään käyttäjät ja tietokanta
ENV MYSQL_RANDOM_ROOT_PASSWORD yes
ENV MYSQL_USER ampache
ENV MYSQL_USER_PASSWORD <salasana>
ENV MYSQL_DATABASE ampache
```