OULUN AMMATTIKORKEAKOULU

Tuomas Imponen

TESTIYMPÄRISTÖN VIRTUALISOINTI SYSTEEMITESTAUKSEN TARPEISIIN

TESTIYMPÄRISTÖN VIRTUALISOINTI SYSTEEMITESTAUKSEN TARPEISIIN

Tuomas Imponen
Opinnäytetyö
Syksy 2019
Tietotekniikan tutkinto-ohjelma
Oulun ammattikorkeakoulu

TIIVISTELMÄ

Oulun ammattikorkeakoulu
Tietotekniikan tutkinto-ohjelma, laite- ja tuotesuunnittelu

Tekijä: Tuomas Imponen
Opinnäytetyön nimi: Testiympäristön virtualisointi systeemitestauksen tarpeisiin
Työn ohjaaja: Kari Jyrkkä
Syksy 2019 Sivumäärä: 70 + 5

Työn toimeksiantajana toimii Oulussa toimiva Bittium Oyj. Yrityksen on kyettävä testaamaan laitteitaan ja ohjelmistojaan uudenlaisessa ympäristössä, jollaista sillä ei ole käytössä entuudestaan. Opinnäytetyössä suunniteltiin ja rakennettiin ympäristö, jolla mahdollistetaan tarvittavat yhdistelmät keskeisten käyttötilanteiden testausta varten.

Toteutetussa ympäristössä testataan eri IP-protokollien ja verkkoliikenteen toimivuutta sekä laatua. Tavanomaiseen pöytätietokoneeseen luotiin kolmetoista virtualisoitua tietokonetta ja nämä liitettiin osaksi fyysistä testilaitteistoa. Testattavaan verkkoon luotiin useita virtuaalilähiverkkoja eri käyttötarkoituksiin. Verkon keskeisiin rooleihin asetettiin yrityksen omia laitteita.

Pohjatiedot työtä varten olivat olemassa jo ennen työn aloitusta. Reitityksen perusteet ja virtuaalilähiverkkojen peruskäsitteet oli opittu jo koulutuksen aikana sekä omien harrasteiden pohjalta. Virtualisointia oli tehty jossain määrin aikaisemmissa töissä. Edellä mainittujen osa-alueiden tiedot kasvoivat opinnäytetyön aikana runsaasti ja oppeja sovellettiin käytäntöön aiempaa laajemmin. Uudet asiat ja ratkaisut ongelmiin selvitettiin verkkolähteiden kautta lukemalla eri julkaisuja ja keskusteluita.

Tuloksena syntyi joustava ja monikäyttöinen ympäristö, joka on helposti muokattavissa ja laajennettavissa tarpeen mukaan. Nykyisessä tilassaan se täyttää työn tavoitteeksi asetetut keskeiset käyttötilanteet.

Työssä kertynyt osaaminen on hyödynnettävissä laajalti ja tarjoaa jatkossa hyvän pohjan tietojen syventämistä varten. Virtualisoinnin tuoma resurssien- ja tilansäästö laboratoriotiloissa on yrityksen kannalta järkevää, ja se tuleekin hyödyntämään virtualisointia jatkossa kasvavassa määrin.

Asiasanat: virtualisointi, virtuaalilähiverkko, testaus

ABSTRACT

Oulu University of Applied Sciences
Degree Programme in Information Technology, Device and Product Design

Author: Tuomas Imponen
Title of thesis: Testiympäristön virtualisointi systeemitestauksen tarpeisiin
Supervisor: Kari Jyrkkä
Autumn 2019 Number of pages: 70 + 5

Bittium Plc must be able to test its devices and software in a new kind of environment, which it is currently lacking. Thesis work consists of planning and implementing the needed environment, enabling testing of required combinations and use cases.

Testing the functionality of different IP-protocols, together with quality of general network traffic, is to be covered by the new environment. Standard desktop PC is used to create 13 virtualized computers, which are used together with the physical test place equipment. Inside the network under test, multiple virtual local area networks are created for different purposes. Company's own products will act in central roles in the network.

General knowledge about the technology used existed prior to the thesis work. Principles in network routing and in virtual local area networks had been learned during the degree programme and from different amateur works. Virtualization was familiar through previous work. Knowledge in the subjects grew substantially during the thesis work, and it was used on a larger scale than before. New information, and solutions to problems encountered, were mainly gathered using online sources.

The work resulted in flexible and versatile environment, which can be easily modified and extended to meet different needs. In its current state, it fulfils the requirements set for central use cases.

Knowledge accumulated during the thesis work can be applied extensively and provides a good foundation for further study. Savings in resources and space provided by virtualization are notable and the company will continue to use it in the future.

Keywords: virtualization, hypervisor, testing, networking

SISÄLLYS

1 JOHDANTO

Tuotekehityksen aikana Bittium Oyj:n on kyettävä testaamaan laitteitaan ja ohjelmistojaan yrityksen tiloissa, jotta mahdolliset viat voidaan havaita ja korjata jo ennen tuotteen toimitusta asiakkaalle. Testauksen kohteena toimivat niin yrityksen ohjelmistot kuin itse laitteetkin. Osa IP-verkkoliikennettä välittävistä laitteista tulee olemaan yrityksen omia tuotteita, jotka sisältävät myös niihin kehitetyn ohjelmiston. Sijoittamalla kehityksessä olevia laitteita keskeisiin rooleihin verkon rungossa saadaan niiden toiminta varmistettua. Toisaalla taas yrityksellä on puhtaita ohjelmistokomponentteja, jotka voidaan asentaa tavalliseen Linux- tai Windows-käyttöjärjestelmään. Nämä ohjelmistot tarjoavat palveluita lähettäen ja vastaanottaen informaatiota olemassa olevassa IP-verkossa.

Työn tavoitteena on suunnitella ja rakentaa ympäristö, jota tullaan käyttämään toimitettavan järjestelmän testaamiseen. Ympäristö pyritään rakentamaan siten, että se kattaa mahdollisimman monta keskeistä käyttötilannetta ja niiden yhdistelmää. Yleisesti kaapelointitekniikkana toimii kupari, mutta osa yhteyksistä tullaan toteuttamaan käyttäen valokuitua.

Testiympäristössä on kyettävä siirtämään IP-liikennettä ja yrityksen laitteiden on toimittava yhteen muiden kaupallisten laitteiden ja protokollien kanssa. Järjestelmä tulee sisältämään useita virtuaalilähiverkkoja ja osa yhteysväleistä toimii virtuaalilähiverkkojen runkoyhteytenä. Järjestelmään lähetettävä kuorma luodaan tavallisilla Linux-tietokoneilla, jotka on virtualisoitu KVM-alustaa käyttäen.

Käytettävät tekniikat ja protokollat ovat tunnettuja ja standardoituja. Työn painopiste on virtualisointiratkaisun suunnittelussa ja toteuttamisessa sekä verkossa toimivien reitittimien ja kytkimien oikeaoppisessa käytössä. Järjestelmän toimivuus osoitetaan suorittamalla valmiissa ympäristöön joitain testejä.

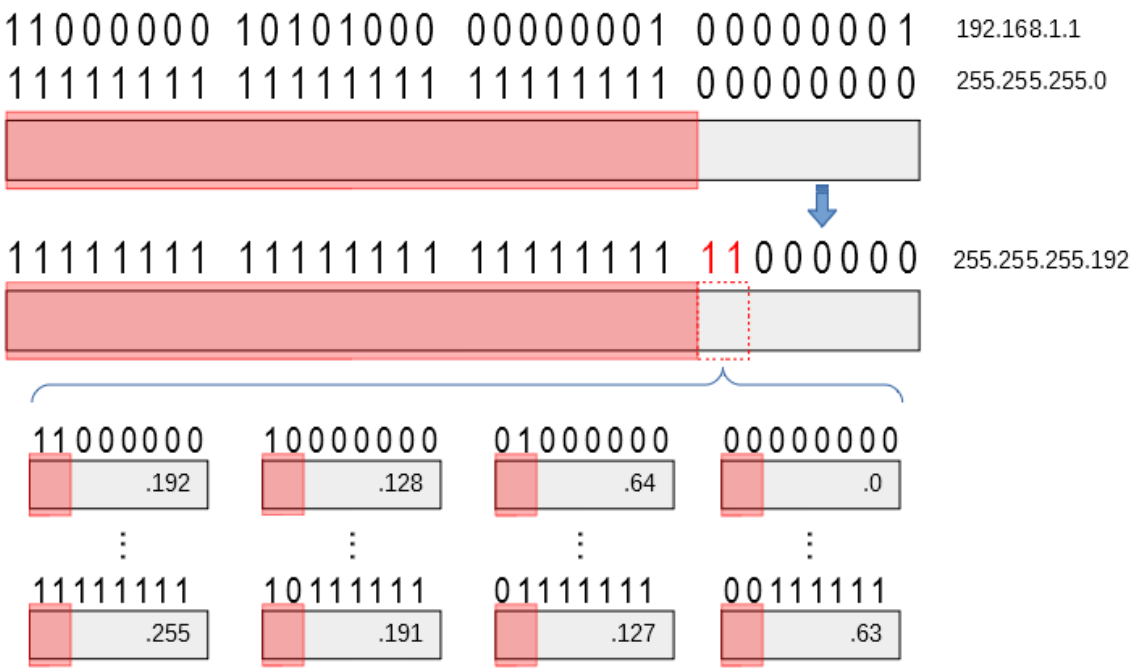
2 IP-VERKOT

Työssä käytettiin useita eri osoiteavaruuksia ja liikenne näiden välillä reititetään eri reititysprotokollia käyttäen. OSPF-protokolla (Open Shortest Path First) oli käytössä virtuaaliverkkojen sekä yhden Ciscon kytkimen ja taktisen reitittimen välisessä liikenteesssä. Taktisten reitittimien välillä käytössä oli OLSR-protokolla (Optimized Link State Routing).

2.1 Aliverkotus

Aliverkotuksessa yksi suurempi verkko pilkotaan pienempiin osiin käyttäen neljä oktettia pitkää aliverkon peitettä, jolla merkitään IP-osoitteen verkko-osaan kuuluvat bitit. Esimerkkinä tuttu osoite 192.168.1.1 saa usein aliverkon peitteen 255.255.255.0. Kyseisellä asetuksella ensimmäiset kolme oktettia, 192.168.1, määräävät verkon. Viimeinen oktetti määrää verkon koon eli siihen mahtuvien päätelaitteiden määrän. Aliverkon maski kirjoitetaan usein lyhyemmässä muodossa 192.168.1.1/24, jossa IP-osoitteen jälkeen ilmoitetaan aliverkon maskiin kuuluvien bittien määrä.

Aina aliverkon peite ei täsmää kokonaiseen oktettiin, jolloin IP-osoitetta ja aliverkon peitettä on tarkasteltava binäärisenä. Yksi oktetti kattaa kahdeksan bittiä. Esimerkkiosoite 192.168.1.1 on esitetty kuvassa 1 ylimpänä ja osoitteen alle on merkitty aliverkon peite 255.255.255.0 binäärimuodossa. Kasvattamalla aliverkon peitettä kahdella bitillä jakaantuu aiempi 255 osoitteen avaruus neljään pienempään verkkoon, joihin jokaiseen mahtuu 64 osoitetta.



KUVA 1. Aliverkon peitettä kasvattamalla verkko voidaan pilkkoa pienempiin osiin.

Työssä yksi lähiverkoista jaetaan neljään yhtä suureen aliverkkoon. Jokainen näistä aliverkoista tulee toimimaan omassa virtuaalilähiverkossaan.

2.2 Reititysprotokollat

Reititysprotokollien avulla verkon laitteet keskustelevat ja sopivat keskenään, mitä kautta liikenne reititetään. Reitit pohjautuvat metriikkaan, jonka päälle protokolla on rakennettu.

Verkon rakentaminen ilman reititysprotokollia onnistuu, jos verkon koko pysyy pienenä, mutta verkon kasvaessa ylläpito muodostuu lähes mahdottomaksi ja aikaavieväksi. Jokainen verkon reititin olisi konfiguroitava käsin ja inhimillisen virheen vaara on suuri. Lisäksi pienikin muutos verkossa voi aiheuttaa sen toimintakyvyttömyyden. Reititysprotokollat voidaan jakaa usealla eri tavalla taulukon 1 mukaan.

Exterior gateway -protokollat ovat käytössä liikenteen reitittämisessä autonomisesta järjestelmästä toiseen. Autonominen järjestelmä tarkoittaa jonkin yhden toimijan hallinnassa ja ylläpidossa olevaa järjestelmää, verkon osaa. Jokaisella autonomisella järjestelmällä on internetissä oma yksilöllinen numeronsa, jonka perusteella esimerkiksi border gateway -protokollareititys tapahtuu.

Vastaavasti interior gateway -protokollat ovat käytössä autonomisen järjestelmän sisäisessä liikenteessä. (CNNA Blog 2019a, viitattu 20.6.2019).

Distance vector- ja link state -reititysprotokollat ovat kaksi interior gateway -protokollatyypin alaluokkaa. Distance vector -reititysprotokollassa reittiä mainostetaan seuraavan hypyn perusteella. Reititin on oppinut, missä suunnassa ja kuinka monen muun reitittimen takana jokin verkko sijaitsee, tekee reityspäätöksen tämän perusteella ja lähettää paketin seuraavalle laitteelle reitin varrella. Lähettävä reititin ei tiedä, missä määränpää on, mutta se tietää jonkun, joka osaa ohjata paketin eteenpäin. Link state -reitityksessä reitittimellä puolestaan on verkon topologia tiedossaan. Se tuntee määränpään ja kaikki risteyskohdat sen varrella. Lähettävä reititin valitsee parhaan polun ja lähettää paketin eteenpäin seuraavalle reitittimelle valitsemansa polun varrella. (CNNA Blog 2019a, viitattu 20.6.2019.)

Luokattomien ja luokallisten protokollien erona on niiden toimivuus aliverkkoja käytettäessä. Luokalliset protokollat syntyivät ennen luokatonta verkkoalueiden välistä reititystä eivätkä sisällä aliverkon peitettä. Tämän johdosta luokallisia protokollia ei voida käyttää, mikäli verkko on aliverkotettu. Luokalliset protokollat ovat lähes poistuneet käytöstä moderneissa verkoissa. (CCNA Blog 2019b, viitattu 19.6.2019.)

TAULUKKO 1. Reititysprotokollien jako eri luokkiin (CNNA Blog 2019a, viitattu 20.6.2019).

	Interior gateway -protokollat				**Exterior gateway -protokollat**
	Distance vector		**Link state**		**Path vector**
Luokallinen	RIPv1	IGRP			EGP
Luokaton	RIPv2	EIGRP	OSPF v2	IS-IS	BGPv4
IPv6	RIPng	EIGRP IPv6	OSPF v3	IS-IS IPv6	BGPv4 IPv6

Rakennettavassa verkossa käytetään reititysprotokollia taktisten reitittimien välillä. Kahdesta yhteysvälistä toinen käyttää OSPF- ja toinen OLSR-reititysprotokollaa. Lisäksi OSPF-protokollan tuntemus on tarpeen Ciscon laitteita konfiguroitaessa. Seuraavat luvut avaavat hieman näiden protokollien toimintaa.

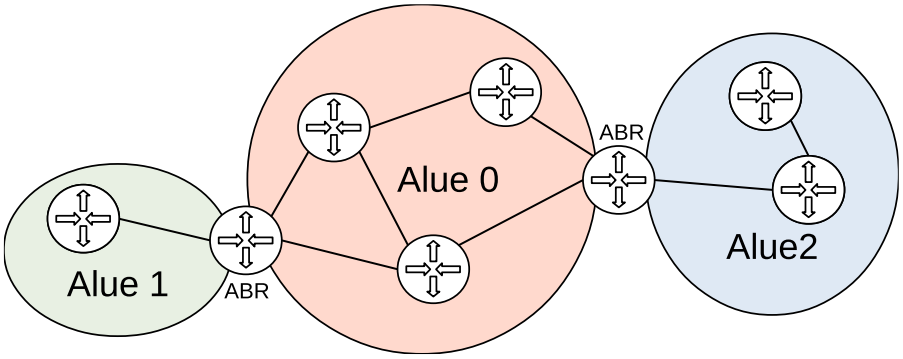
2.2.1 OSPF-protokolla

Open Shortest Path First on luokaton link state -reititysprotokolla. Yhteysväli saa eri arvoja sen tiedonsiirtokapasiteetin perusteella. Reittiä laskettaessa yhteysvälien arvot lasketaan yhteen, mistä saadaan reitin kokonaisarvo. Lopullista arvoa käytetään metriikkana parhaan reitin valitsemiseen. Yhteysvälin arvo asettuu automaattisesti verkkorajapinnan nopeuden mukaan, mutta asettamalla arvon manuaalisesti voi käyttäjä vaikuttaa reitityspäätöksen tekoon. (CCNA Blog 2019c, viitattu 19.6.2019.)

Käytettäessä OSPF-protokollaa käytössä on usein myös useita OSPF-alueita. OSPF-alueiden pilkkominen pienempiin osiin vähentää reitittimien resurssitarpeita, kun verkkotopologian ei tarvitse sisältää kaikkia verkossa olevia reitittimiä.

Topologia mainostetaan ja reitit lasketaan ainoastaan alueen sisällä. Alueen ulkopuolella olevat reitit yhdistetään yhdeksi reitiksi, jonka kautta liikenne alueelle ja alueelta tapahtuu. Mikäli verkkoa ei jaettaisi pienempiin osiin, topologian muuttuminen aiheuttaisi tilamuutosten jakamisen ja reittien uudelleenlaskemisen kaikille reitittimille. (Juniper Networks 2017, viitattu 19.6.2019.)

Kuvassa 2 alueiden välillä toimivat reitittimet vastaavat liikenteen välittämisestä alueesta toiseen. Toisin kuin alueen sisäiset reitittimet, ABR (Area Border Router) tietää kaikkien siihen suoraan liitettyjen alueiden täydellisen topologian ja voi tehdä reittiyhteenvetoja. ABR mainostaa alueen sisäisille reitittimille reittiä määränpäähän itsensä kautta, ja mikäli paketin määränpää osuu nykyisen alueen ulkopuolelle, reititin lähettää sen ABR:lle. (Juniper Networks 2017, viitattu 19.6.2019.)



KUVA 2. OSPF-alueiden välinen liikennöinti. Runkoalue nimetään yleensä alue nollaksi.

Vaatimuksena alueiden käyttämiselle on, että runkoalueen on oltava olemassa ja jokaisen OSPF-alueen on oltava suorassa yhteydessä siihen. Liikenne, jonka määränpää ei ole alueen sisällä, reititetään aina runkoalueelle. Runkoalue vastaa liikenteen ohjaamisesta kohdealueelle ABR:n kautta. Käyttämällä yhtä aluetta välittäjänä estetään väärän reititystiedon leviäminen verkossa. (Lindem, Yeung & Zinin 2003, 2.)

2.2.2 OLSR-protokolla

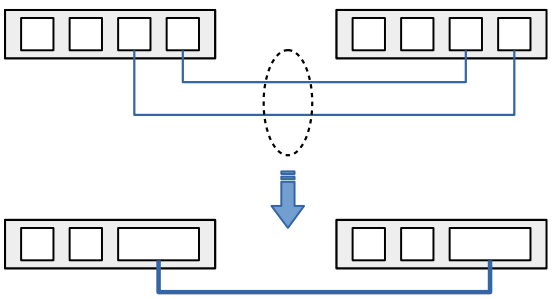
Optimized Link State Routing -protokolla on kehitetty liikkuviin ad hoc -mobiiliverkkoihin (MA-NET). Ad hoc -verkossa langattomat laitteet kommunikoivat suoraan keskenään ilman kiinteää verkkoa ja keskitettyä verkon hallintaa. Jokainen verkon laite on vastuussa liikenteen reitittämisestä ja voi vapaasti liikkua sisään tai ulos verkosta. (Clausen & Jacquet 2003, 4, 8–9.)

OLSR:ssä verkon laite X valitsee jotkin naapurisolmuistaan välittämään protokollaliikennettä, joka on tarkoitettu lähetettäväksi koko verkkoon. Valittu solmu, MPR (Multipoint Relay), mainostaa muille MPR solmuille reittiä laitteeseen X itsensä kautta. (Clausen & Jacquet 2003, 4, 8–9.)

Vastaanottamansa reittitiedon MPR jakaa edelleen laitteille X, jotka ovat valinneet sen protokolla-liikenteensä välittäjäksi. MPRiä käytetään laskettaessa reittejä mihin tahansa kohdeosoitteeseen verkossa, ja niiden käyttäminen minimoi tarvittavan protokollaliikenteen verkossa. (Clausen & Jacquet 2003, 4, 8–9.)

2.3 Verkkorajapintojen yhteenliittäminen

Työssä tarvitaan suurta tiedonsiirtokapasiteettia virtualisointiin käytettävän tietokoneen ja yhden verkon kytkimen välillä. Kapasiteetti saavutetaan yhdistämällä kaksi linkkiä kuvan 3 mukaisesti.



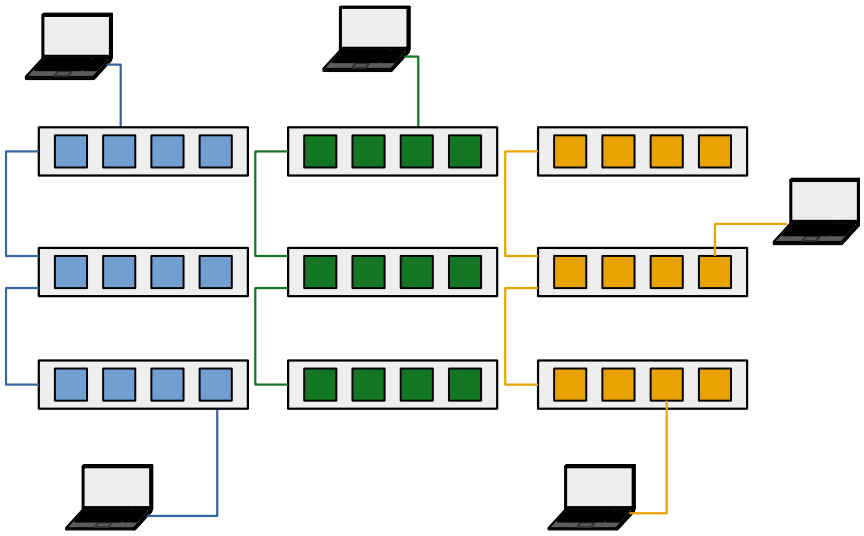
KUVA 3. Kytkinportit voidaan yhdistää loogiseksi kokonaisuudeksi.

Eri valmistajan käyttävät termistä eri nimiä ja tekniikasta voi olla suljettuja toteutuksia. Ciscon dokumentaatiossa tekniikkaan viitataan nimellä ether channell (Cisco Systems 2017, 4). HP:n dokumentit käyttävät nimeä trunk (Hewlett Packard Enterprise 2016, 52). Linuxin oma toteutus tuntee nimen bonding (Davis, Tarreau, Gavrilov, Tindel, Girouard, Vosburgh & Williams 2011). Kaikissa periaate on kuitenkin sama: useamman fyysisen linkin käyttäminen yhtenä loogisena linkkinä tarjoamaan joko vikasietoisuutta, suorituskykyä tai molempia. Yleisnimitys tekniikalle on Link Aggregation, jolle on luotu oma standardinsa 802.3ad (IEEE 2010, viitattu 6.9.2019).

3 VIRTUAALILÄHIVERKKO

Tavanomaisesti kytkimen kaikki portit kuuluvat samaan lähiverkkoon. Vaikka useiden aliverkkojen liittäminen samaan kytkimeen on mahdollista, on hyvän käytännön mukaista pitää yksi aliverkko yhtä lähiverkkoa kohti. Lisäksi jos jokin kytkimeen liitetyistä laitteista vaihtaa IP-osoitteensa toiseen aliverkkoon, se voi kommunikoida esteettä muiden verkon laitteiden kanssa.

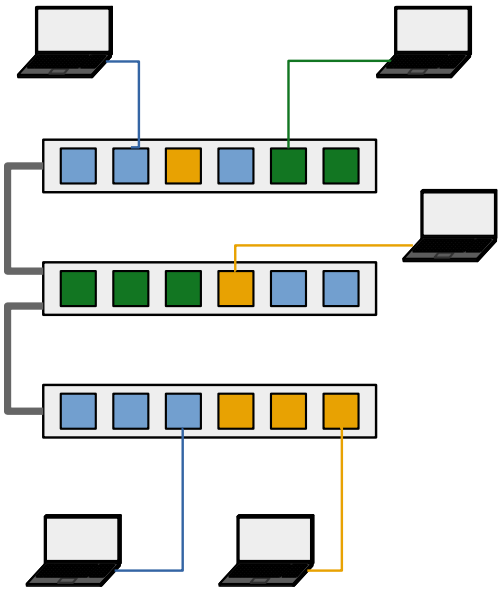
Jos aliverkko vaatii oman kytkimensä, verkkolaitteiden määrä kasvaa ja yhteyden vieminen eteenpäin vaatii oman kaapelointinsa jokaiselle kytkimelle. Kuvan 4 mukaisessa tilanteessa, jossa kaksi samassa kerroksessa olevaa tietokonetta on saatava eri lähiverkkoihin, molemmille tietokoneille on oltava oma kytkimensä ja molemmilta kytkimiltä on kuljetettava oma kaapelointinsa eteenpäin runkokytkimelle. Toisessa kerroksessa sama toistuu. Vaihtoehtoisesti voidaan tietysti kaapeloida suoraan tietokoneilta toisessa kerroksessa sijaitsevaan kytkimeen, mutta kaapelointi käy hyvin pitkäksi. Virtuaaliset lähiverkot tarjoavat helpompia keinoja edellä mainittuun tilanteeseen.



KUVA 4. Ilman virtuaalilähiverkkoja jokainen kytkin sisältää yhden lähiverkon.

VLAN (Virtual Local Area Network) on tekniikka, jossa yksi fyysinen kytkin voidaan jakaa useaan loogiseen osaan ja lähiverkko voidaan ulottaa usean kytkimen yli. Periaate on esitetty kuvassa 5. Eri puolilla rakennusta oleviin kytkimiin liitetyt koneet voivat toimia samassa lähiverkossa keskenään, mutta ne ovat silti eristettynä muista samaan fyysiseen kytkimeen liitetyistä laitteista. (Cisco Systems 2017, 2110.)

Virtuaalilähiverkot tuovat huomattavaa joustavuutta verkonhallintaan ja lisäävät laitteiden tehokasta käyttöä. Käyttämättömien kytkinporttien määrä vähenee, ja verkkolaitteita tarvitaan vähemmän. Mikäli kapasiteettia vielä on, voidaan uusia lähiverkkoja luoda konfiguroimalla verkkolaitteita, eikä uusia tarvitse hankkia.



KUVA 5. Virtuaalilähiverkkojen avulla yksi kytkin voi sisältää useita lähiverkkoja.

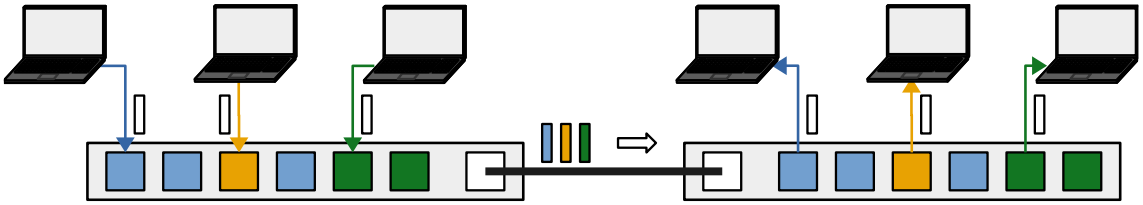
Työssä tullaan käyttämään useita virtuaalilähiverkkoja ja ne ovat keskeisessä roolissa reitityksen kannalta. Virtuaalilähiverkot ulotetaan usean kytkimen yli ja niiden käyttö myös mahdollistaa runkoyhteyksien testaamisen. Tekstissä käydään läpi muutamia keskeisiä virtuaalilähiverkon käsitteitä.

3.1 VLAN-tunniste

Yksi virtuaalilähiverkon eduista on helppo lähiverkkojen laajennettavuus. Vaikka kytkimessä olisi useita virtuaalilähiverkkoja, voidaan niiden sisältämä liikenne välittää eteenpäin yhden kytkinportin kautta. Kuljetettaessa Ethernet-kehyksiä kytkimeltä toiselle on kyettävä tunnistamaan, mistä virtuaalilähiverkosta ne ovat lähtöisin. IEEE 802.1Q-standardi määrittää tekniikan, missä kehykseen lisätään virtuaalilähiverkon yksilöivä kenttä (IEEE 2018, viitattu 6.9.2019).

Kun kuvan 6 mukaisesti portista, joka on määritelty kuulumaan useampaan kuin yhteen virtuaalilähiverkkoon, lähtee liikennettä, se sisältää virtuaalilähiverkon tunnisteen. Näin vastaanottavassa

päässä on mahdollista tietää, mihin virtuaalilähiverkkoon saapuva paketti kuuluu. Vastaanottava kytkin ohjaa liikenteen virtuaalilähiverkkoon, jolla on sama tunniste kuin saapuneella kehyksellä. Ylläpitoa ajatellen on suositeltavaa määrittää oma osoiteavaruutensa jokaista virtuaalilähiverkkoa kohden.



KUVA 6. Lähiverkon tunnistetta käyttämällä tiedetään, mihin verkkoon paketti kuuluu.

Kytkinportti, joka kuuluu vain yhteen virtuaalilähiverkkoon ja käyttäytyy normaalin portin tavoin, ei sisällä lähiverkon tunnistetta lähtevässä liikenteessä. Tällöin liikenteen ei pidä ajatella kuuluvan mihinkään virtuaalilähiverkkoon. Liikenne voi lähteä portista, joka on määritetty kuulumaan virtuaalilähiverkkoon 5, ja saapua virtuaalilähiverkkoon 14 kuuluvaan porttiin. Mikäli virtuaalilähiverkkoon 14 saapunut liikenne välitetään runkoyhteyttä käyttäen edelleen eteenpäin, se saa tunnisteen 14 eikä 5, johon alkuperäinen lähettävä portti kuului.

3.2 VLAN-runkoyhteys

Yhteysväli, joka kuljettaa virtuaalilähiverkon tunnisteen sisältäviä Ethernet-kehyksiä, toimii VLAN-runkoyhteytenä. Runkoyhteydestä käytetään usein englanninkielistä nimitystä trunk. Runkoyhteysporttiin saapuva ja siitä lähtevä liikenne yksilöidään useimmiten 802.1Q-standardin mukaisesti. (Cisco systems 2017, 2.)

VLAN-runkoyhteydestä käytetään useita nimiä. Cisco käyttää yhteysvälistä termiä trunk. Jotkin valmistajista viittaa termillä trunk standardin IEEE 802.3ad mukaiseen aggregointiin. Tässä dokumentissa termiä trunk, tai runkoyhteys, käytetään virtuaalilähiverkon tunnisteen sisältäviä Ethernet-kehyksiä kuljettavasta yhteysvälistä.

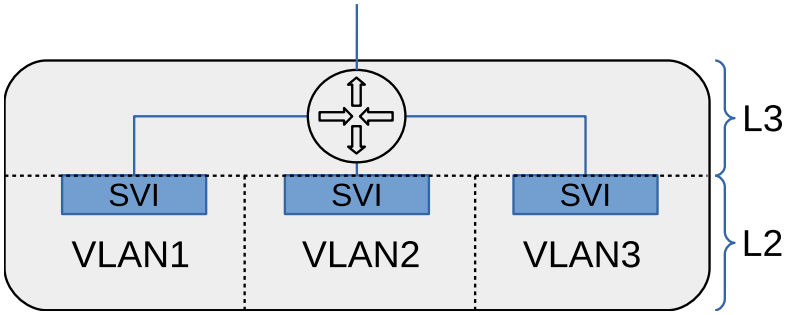
3.3 Virtuaalilähiverkkojen reitittäminen

Kytkimen hallintaa ja liikenteen reitittämistä varten jokaisella kytkimellä on oltava rajapinta reitittimeen liittymistä varten. Ciscon kytkimissä tämä rajapinta on SVI (Switch Virtual Interface). Kuvan

7 mukaisesti jokaiselle virtuaalilähiverkolle on oltava määritettynä oma SVI, jonka kautta reititys tapahtuu. (Cisco Systems 2017, 3.)

Kytkimet toimivat L2-tasolla eli OSI-mallin siirtoyhteyskerroksella ja reitittimet L3- eli kuljetuskerroksella. Niin kutsutut L3-tason kytkimet yhdistävät nämä kaksi ja osaavat tarpeen vaatiessa myös reitittää liikennettä.

Kun kytkimen reititysominaisuudet on otettu käyttöön ja SVI:lle on määritetty IP-osoite, jokainen virtuaalilähiverkko näkyy L3-tasolle suoraan kytkettynä verkkona. Yksi työssä käytetyistä Ciscon kytkimistä toimii reitittimenä ja sille on luotava SVI jokaista virtuaalilähiverkkoa kohden.



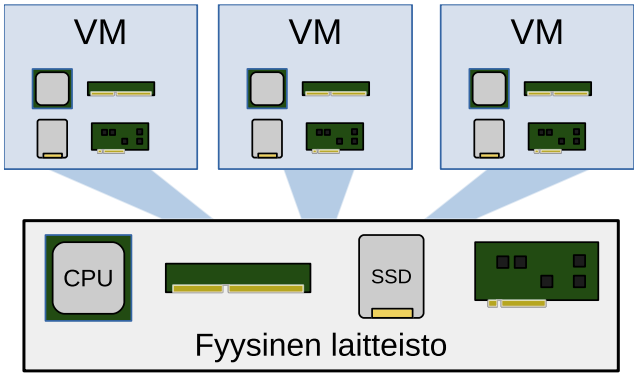
KUVA 7. SVI toimii sisäisenä rajapintana reitittimelle.

4 VIRTUALISOINTI

Samaan tapaan kuin virtuaalilähiverkot tuovat etuja verkkolaitteisiin, tietokoneiden resurssikäyttöä voidaan parantaa virtualisoinnilla. Yleisessä toimistokäytössä oleva tietokone on suurimman osan ajasta hyvin vähäisellä käytöllä. Kiintolevytilaa on riittämiin, prosessorin käyttöaste ei juurikaan nouse pitkiksi ajoiksi ja keskusmuistia on hyvin vapaana.

Tietokoneella on yleisesti arkikäytössä suoritettavana vain yksi käyttöjärjestelmä kerrallaan. Mikäli on tarve vaihtoehtoiselle käyttöjärjestelmälle, on joko hankittava toinen laitteisto, johon järjestelmä asennetaan, tai valittava toinen järjestelmä nykyiseen laitteistoon. Uuden laitteiston hankinta tietää kustannuksia ja varsinkin, mikäli käyttötarve on vähäinen tai satunnainen, hukataan resursseja. Useamman käyttöjärjestelmän vaihtelu keskenään taas ei ole käytettävyyden kannalta mukavin ratkaisu.

Mikäli isäntäkoneessa on riittävästi vapaita resursseja, voidaan tietokone virtualisoida sen tarvitsemaa laitteistoa myöten. Kuvan 8 virtualisoitu laitteisto mahdollistaa useiden täysin toisistaan riippumattomien käyttöjärjestelmien yhtäaikaisen ajamisen ja virtuaalikoneiden laitteistoa on helppo mukauttaa eri käyttötarpeisiin. Mikäli virtuaalitietokone tarvitsee uusia laitteita, kuten verkkokortin, lisää muistia tai prosessoriytimiä, näitä kaikkia voidaan lisätä helposti virtuaalikoneen asetuksia muutamalla.



KUVA 8. Samassa laitteistossa voi sijaita useita virtuaalikoneita.

Useita erilaisia järjestelmiä voidaan ajaa rinnakkain samassa fyysisessä laitteistossa, ja tarpeen mukaan ne voivat joko kommunikoida keskenään tai olla eristettynä toisistaan. Kun tarvitaan

erilaisia ympäristöjä ohjelmistojen testaamiseen, on järkevää asentaa useita virtuaalikoneita eri käyttöjärjestelmillä ja laitteistokonfiguraatioilla ja yhdistellä näitä tarpeen mukaan.

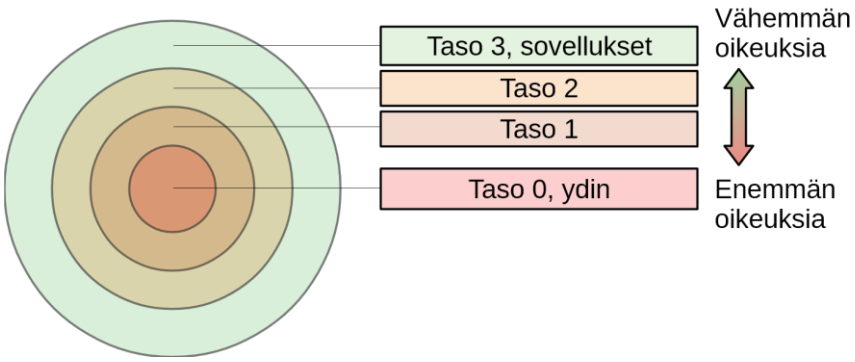
Seuraavat luvut käsittelevät virtualisointia enemmän prosessorin ja käskyjen suorituksen kannalta. Muistinhallinnan ja yleisen I/O:n käsitteleminen vaatisi kokonaan oman päälukunsa, ja virtualisoinnin perusteet nousevat paremmin esille prosessorin kautta. Virtualisointi on keskeinen osa toteutettavaa työtä ja luvut pyrkivät tarjoamaan siitä joitain pohjatietoja. Kaikki työssä käytettävät tietokoneet isäntää lukuun ottamatta tulevat olemaan virtuaalisia.

4.1 Suoritustasot

Virtualisointiin liittyvät läheisesti kuvan 9 mukaiset prosessorin eri suoritustilat, joissa ohjelmakoodia suoritetaan. Suoritettavalla koodilla on pääsy eri käskyihin sen mukaan, missä tilassa prosessori on. x86-prosessoriarkkitehtuurissa kehiä on nollasta kolmeen, mutta usein käytössä on vain sisimmäinen ja uloimmainen kehä. Sisimmästä kehästä käytetään myös nimitystä ydintila ja uloimmasta käyttäjätila.

Kehät ovat abstrakti käsite. Oikeammin kyseessä on vain koodia, jolla on oikeuksia suorittaa käskyjä, ja koodia, jolla ei niitä ole. Kehillä ei myöskään ole mitään yhteistä käyttäjä- tai järjestelmänvalvojan oikeuksien kanssa, mitkä useimmat tuntevat arkipäiväisestä tietokoneenkäytöstä. Tekstissä käytettävät termit tila, taso ja kehä tarkoittavat samaa asiaa. Englanninkielisiä termejä ovat protection ring, privileged state, unprivileged state, kernel mode ja user mode.

Syy vain kahden tilan käyttöön on modernien käyttöjärjestelmien toteutuksessa. Käyttöjärjestelmien toteuttama suojaustasojen hallinta sallii vain yhden merkitsevän bitin tilan merkitsemiseen, jolloin vaihtoehtoja on vain kaksi. Osasyynä on yhteensopivuus. Jotkin prosessoriarkkitehtuurit käyttävät vain kahta kehää, ja tekemällä näin myös ohjelmistossa on koodi helpommin siirrettävissä toiselle alustalle. (Hanrahan 2016, viitattu 19.6.2019.)



KUVA 9. Pääsyoikeudet järjestelmään kasvavat ydintä lähestyttäessä.

Sisimmällä kehällä, kehällä nolla, ajettavalla koodilla on suurimmat suoritusoikeudet. Tällä tasolla kaikki prosessorin käskyt ovat käytettävissä ja siten ohjelmakoodilla on rajoittamattomat oikeudet suorittaa mikä tahansa operaatio. Kehällä nolla on tarkoituksena suorittaa vain luotettua koodia, kuten käyttöjärjestelmän ydin ja laitteistoajurit, jotka tarvitsevat pääsyn suoraan laitteistoon. Tästä juontaa myös käytetty ydintilatila-nimitys. Ydintilassa ajettavaa koodia ei valvo mikään, joten on vain pakko luottaa siihen, että ainoastaan asianmukainen koodi voi käyttää sitä. (Introduction to Operating Systems 2012, viitattu 19.6.2019.)

Tasolla kolme suoritettavalta koodilta on evätty pääsy niihin prosessorikäskyihin, jotka on sallittua suorittaa vain ydintilassa. Näitä ovat yleisesti mitkä tahansa komennot, jotka muuttavat tai lukevat koneen yleistä tilaa, kuten käyttöjärjestelmää itseään tai muita ohjelmia, tai pyrkivät käyttämään esimerkiksi laitteen verkkokorttia. (Introduction to Operating Systems 2012, viitattu 19.6.2019.)

Jokaiselle ohjelmalle on määritetty sille kuuluva muistialue ja se voi käyttää vain aluetta, joka sille on myönnetty. Käyttöjärjestelmä tarjoaa sen päällä ajettaville ohjelmille abstraktoidut toiminnot, järjestelmäkutsut, suojattujen käskyjen suorittamiseen. (Introduction to Operating Systems 2012, viitattu 19.6.2019.)

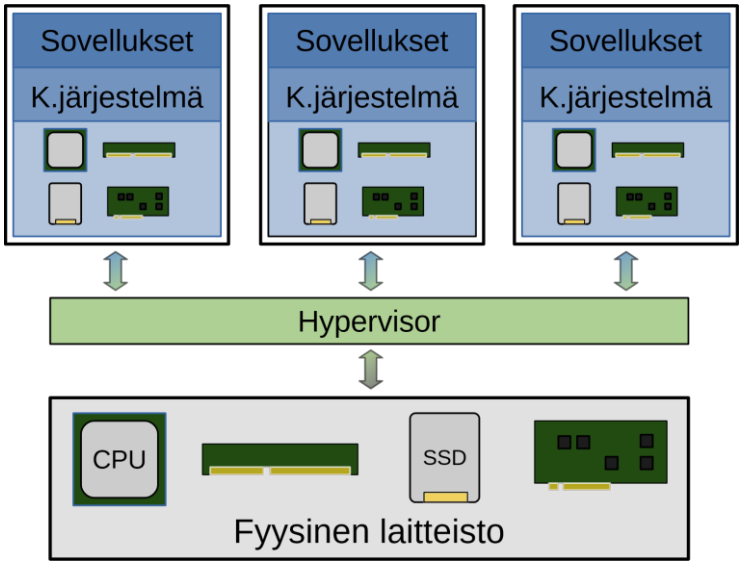
Kun tason kolme käyttäjätilan koodi haluaa suorittaa jonkin matalamman tason käskyn, se valmistelee pyynnön ja lähettää sen käyttöjärjestelmälle. Käyttöjärjestelmä tarkistaa, onko ohjelmalla oikeutta tehdä pyyntöä. Mikäli pyyntö hyväksytään, käyttöjärjestelmä ottaa sen vastaan ja siirtyy käskyn suoritukseen. Prosessorin tila vaihdetaan käyttäjätilasta ydintilaan ja nyt käyttöjärjestelmä voi suorittaa pyydetyt käskyt. Kun operaatio on valmis, prosessori palautetaan käyttäjätilaan ja pyynnön suorittanut ohjelma voi jatkaa omin avuin. (Introduction to Operating Systems 2012, viitattu 19.6.2019.)

Tilojen vaihtamiseen ja pyyntöjen käsittelyyn kuluu runsaasti aikaa. Virtualisoinnissa pyritään minimoimaan tilanvaihtojen tarve sekä tekemään ne niin tehokkaaksi kuin mahdollista. Tämä nousee hyvin esille tarkasteltaessa myöhemmin eri virtuaaliverkkojen toteutusta.

4.2 Resurssien jakaminen

Virtualisoinnin mahdollistaa hypervisor-ohjelma, joka jakaa isäntäkoneen resursseja virtuaalikoneille kuvan 10 mukaisesti. Kuten käyttöjärjestelmät palvelevat ja hallinnoivat ohjelmia, hypervisor palvelee ja hallinnoi käyttöjärjestelmiä.

Käyttöjärjestelmät on suunniteltu hallinnoimaan laitteistoa, johon ne on asennettu, täysin. Käyttöjärjestelmän tehtävä on normaalitilanteessa kontrolloida siinä ajettavia ohjelmia ja valvoa koko järjestelmän tilaa. Kehää nolla ei voida kuitenkaan luovuttaa virtuaalikoneelle, sillä laitteiston hallinta menetettäisiin kyseiselle käyttöjärjestelmälle. Tämä rikkoisi virtualisoinnin asettaman isolaation. Tämä johtaa siihen, ettei usean yhtäaikaisen käyttöjärjestelmän suorittaminen ei ole mahdollista ilman ylimääräistä ohjelmistoa, hypervisoria, ja sen tarjoamia virtuaalisia laitteistoresursseja.

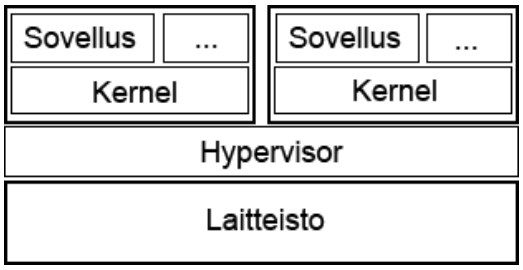


KUVA 10. Hypervisor on vastuussa resurssien jakamisesta virtuaalikoneille.

Hypervisor tarjoaa eristetyn ympäristön käyttöjärjestelmää varten ja valvoo virtuaalikoneiden suoritusta. Samalla tavoin kuin käyttöjärjestelmä jakaa resursseja ohjelmille, hypervisor jakaa resursseja käyttöjärjestelmille. Se kontrolloi ja pitää kirjaa virtuaalisista laitteistoresursseista, kuten mis-

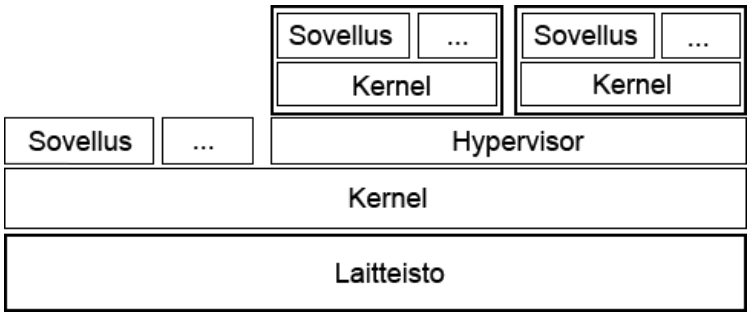
sä kohtaa keskusmuistia kunkin virtuaalikoneen muisti sijaitsee, jakaa prosessoriaikaa sekä pitää virtuaalikoneet eristettynä isännästä ja muista virtuaalikoneista.

Hypervisorit on jaettu karkeasti kahteen eri tyyppiin. Tyypin 1 hypervisorit toimivat suoraan laitteiston päällä ja virtuaalikoneet edelleen näiden päällä. Käyttöjärjestelmä ei varsinaisesti osallistu sen toimintaan, vaan hypervisor toimii siitä erillään omana kerroksenaan ja omistaa kontrollin fyysiseen laitteistoon. Liikenne kulkee suoraan virtuaalikoneiden, hypervisorin ja laitteiston välillä kuvan 11 mukaan. Erillistä hallintaympäristöä käytetään lähinnä hypervisorin ja siihen asennettujen virtuaalikoneiden hallintaan ja laitteistoajureiden tarjoamiseen. Työssä käytetty KVM (Kernel-based Virtual Machine) voidaan luokitella tähän tyyppiin.



KUVA 11. Periaattellinen tyypin1 hypervisor. Tarkempi totetus vaihtelee hypervisoreittain.

Tyypin 2 hypervisorit asennetaan olemassa olevan käyttöjärjestelmän päälle ja toimivat tavallisen ohjelman tavoin. Tyyppi 2 on riippuvainen laitteistoa kontrolloivasta käyttöjärjestelmästä. Liikenne kulkee virtuaalikoneen, hypervisorin ja käyttöjärjestelmän läpi laitteistolle. Periaate on esitetty kuvassa 12.



KUVA 12. Tyypin 2 hypervisor.

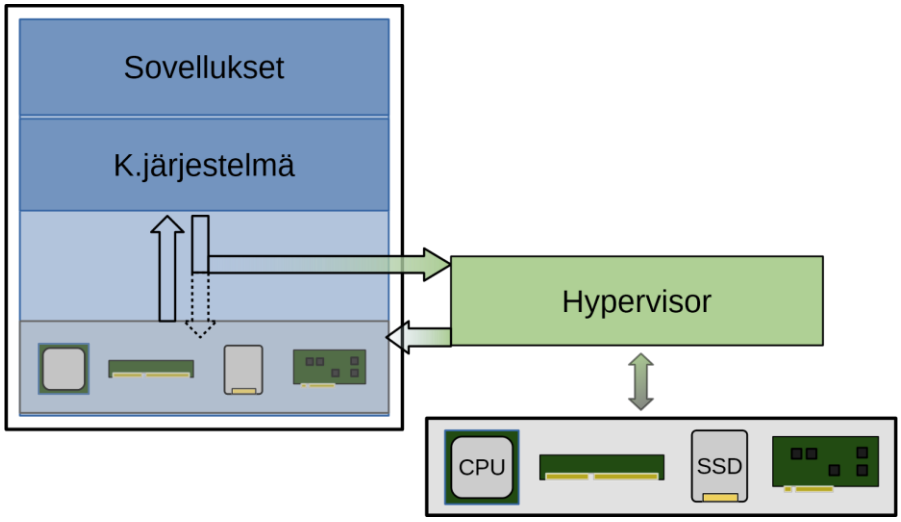
4.3 Virtualisointitekniikat

Virtualisointi voidaan toteuttaa usealla eri tavalla. Hypervisoriksi valittu KVM käyttää laitteistotasolta löytyviä laajennoksia virtualisoinnin suorituskyvyn parantamiseen ja paravirtualisoituja laitteistoajureita tehokkaaseen I/O-liikenteeseen. Pohjatiedot eri virtualisointityypeistä esitetään alla.

4.3.1 Täysi virtualisointi

Täysin virtualisoitu käyttöjärjestelmä ei tiedä olevansa virtualisoitu vaan suorittaa toimintoja, kuten se tekisi normaalistikin. Hypervisorin keskeinen tehtävä on tarjota vieraalle ympäristö, jota se ei kykene erottamaan oikeasta fyysisestä laitteistosta, ja samalla kuitenkin estää sitä käyttämästä laitteistoa suoraan. (Popek & Goldberg 1974, 413.)

Estämällä suoran pääsyn laitteistoon hypervisor ylläpitää illuusiota virtuaalikoneita varten (kuva 13). Jokainen virtuaalikone näkee vain, mitä sen halutaan näkevän. Käyttöjärjestelmä luulee olevansa asennettu suoraan fyysiselle laitteistolle, olevansa laitteiston ainut käyttöjärjestelmä ja käyttävänsä vapaasti fyysisen koneen kaikkia resursseja. Virtuaalikone luulee, että sen suorittamat komennot muuttavat järjestelmän tilaa, mutta todellisuudessa tilanne on toinen. Vieras ei kuitenkaan tätä tiedä, eikä sen tarvitsekaan tietää.



KUVA 13. Käyttöjärjestelmä ei tiedä hypervisorin olemassaolosta.

Täydellä virtualisoinnilla saavutetaan suurin yhteensopivuus ja vaivaton käyttö, sillä asennettavaa vierasta ei tarvitse muokata millään tavalla. Täydessä virtualisoinnissa voidaan ottaa ja käyttää

mitä tahansa käyttöjärjestelmää edellyttäen, että sen tarvitsema laitteisto voidaan emuloida ja vieras osaisi sitä muutenkin käyttää. Täysi virtualisointi on osana työtä laitteistoavustetun virtualisoinnin muodossa.

Ohjelmistoavustettu virtualisointi

Täyden virtualisoinnin toteuttamiseksi x86-arkkitehtuurissa on jouduttu ratkaisemaan useita ongelmia, sillä arkkitehtuuria ei suunniteltu virtualisointia silmälläpitäen. Esteitä klassisen virtualisointitekniikan toteuttamiseksi olivat ainakin prosessorin suoritustilan näkyminen ja keskeytyshyppyjen puuttuminen, kun tietyt ydintason käskyt ajetaan käyttäjätilassa. (Adams & Agesen 2006, 3.)

Ongelmia esiintyy, kun sovellus pyytää käyttöjärjestelmältä sitä suorittamaan jonkin ydintilan käskyn. Hypervisorin kontrolloidessa pääsyä fyysiseen laitteistoon ei käyttöjärjestelmällä ei ole oikeuksia käskyn suoritukseen ja käsky hylätään ilman jatkotoimenpiteitä. Tiettyjen käskyjen hiljainen hylkääminen estää virtuaalikoneen toiminnan. Kun käskyistä ei lähde minkäänlaista viestiä eteenpäin eikä niitä merkitä epäonnistuneeksi, ei hypervisor voi käsitellä niitä. (Fisher-Ogden 2019, 4.)

Ohjelmistoavustetussa virtualisoinnissa ohjelman suorittamaa binääriä tarkkaillaan ongelmakohtien varalta ja kun havaitaan ongelmallinen käsky, binääriin asetetaan keskeytyshyppy ennen käskyn suoritusta. Muokkauksen avulla hypervisor voi ottaa tilanteen hallintaansa ja jäljitellä ongelmallisten käskyjen suorituksen. (Fisher-Ogden 2019, 5.)

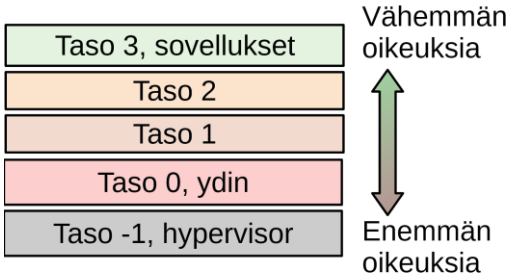
Huonona puolena on hitaus. Käskyjen emulointi vie prosessoriaikaa. Ohjelmistoavustettua virtualisointia ei juurikaan käytetä, sillä modernit x86-prosessorit tukevat laajennoksia virtualisoinnin toteuttamiseksi.

Laitteistoavustettu virtualisointi

Ohjelmistoavustettua virtualisointia kehittyneempi ratkaisu on laitteistoavustettu virtualisointi. Arkkitehtuurista aiheutuvien virtualisointihaasteiden korjaamiseksi x86-arkkitehtuuriin on jälkikäteen lisätty uusia käskyjä ja suoritustiloja.

Laitteistotason avut prosessorilla ja muistissa parantavat virtualisoinnin suorituskykyä ja tekevät x86-arkkitehtuurista klassisesti virtualisoitavan (Adams & Agesen 2006, 5). Sekä AMD:llä että Intelillä on prosessoreissaan laajennoksia, joilla virtuaalikoneen prosessorille lähettämiä käskyjä ei tarvitse ajaa hitaan käännöksen läpi. AMD kutsuu laajennoksia nimellä AMD-V ja Intel puolestaan VT-x. Laajennoksista käytetään nimitystä VMX, Virtual Machine eXtensions.

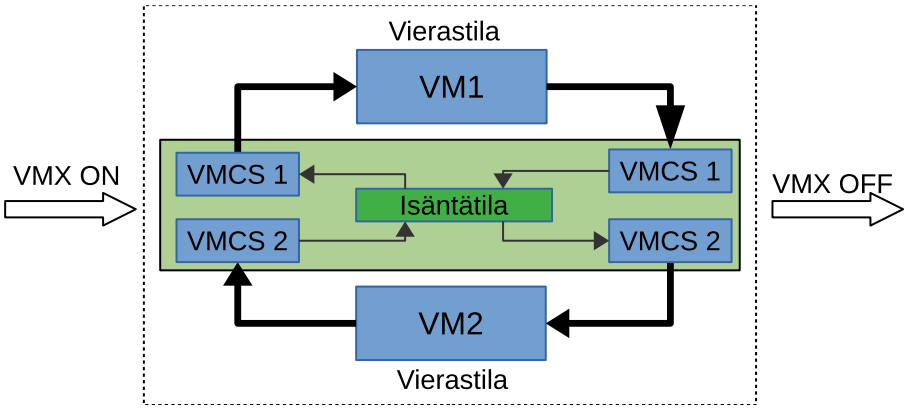
Laitteistoavustettu virtualisointi esittelee kuvan 14 mukaisen tason −1. Hypervisor sijaitsee negatiivisella tasolla näkymättömissä kehällä nolla olevalle virtuaalikoneelle. Kuten kehällä nolla suoritettavalla ytimellä on enemmän oikeuksia verrattuna kehään 3 nähden, samoin negatiivisella tasolla sijaitsevalla hypervisorilla on valta sen yläpuolisiin tasoihin.



KUVA 14. Hypervisor saa kayttöönsä oman tason.

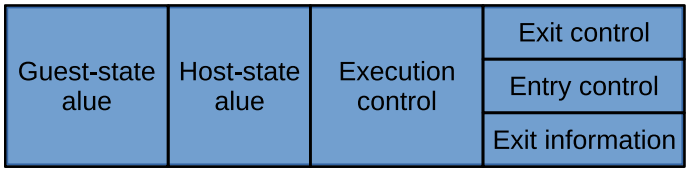
Laajennokset tuovat mukanaan kaksi uutta toimintatilaa prosessoriin. Prosessori voi nyt toimia joko isäntätilassa tai vierastilassa. Isäntätila vastaa toiminnallisesti tilaa ennen laajennoksien lisäämistä ja se on tarkoitettu matalimmalla kehällä olevan koodin käyttöön. Virtualisoinnissa hypervisor ottaa tämän tilan haltuunsa ja kontrolloi vierastilan käyttöä. Englanninkielisiä termejä ovat host mode, guest mode, root operation ja non-root operation. (Fisher-Ogden 2019, 5.)

Virtuaalikone suorittaa omaa koodiaan, kunnes jokin hypervisorin VMCS:ään asettama ehto täyttyy. Ehdon täyttyminen keskeyttää virtuaalikoneen suorituksen ja prosessori palaa isäntätilaan palauttaen kontrollin hypervisorille. Hypervisor käsittelee palautuksen ja laukaisee seuraavan virtuaalikoneen suorituksen. Sekvenssi on esitetty kuvassa 15. Virtuaalikoneiden annetaan siis suorittaa kehän nolla koodia natiivisti suoraan prosessorilla, mutta kuitenkin valvottuna. (Fisher-Ogden 2019, 5–6.)



KUVA 15. Hypervisor valvoo virtuaalikoneiden suoritusta.

Virtual Machine Control Structure (VMCS) on neljän kilotavun kokoinen alue muistissa, jota prosessori käyttää VMX-toimintoihin. Alue voidaan jakaa kuuteen osaan kuvan 16 mukaan. Vierastilasta poistumisen ehdot asetetaan VM-execution control -kenttään ja poistumisen syy tallennetaan VM-exit information -kenttään. Prosessorin viimeisin tila ennen poistumista tallennetaan VMCS guest-state -alueeseen, josta se ladataan, kun hypervisor taas laukaisee kyseisen VMCS:n suorituksen. Isäntätilalle on oma samaan tapaan toimiva alueensa. VMCS on mahdollista luoda virtuaalisen prosessoriytimen tarkkuudella. (Intel 2019, 7, 9.)



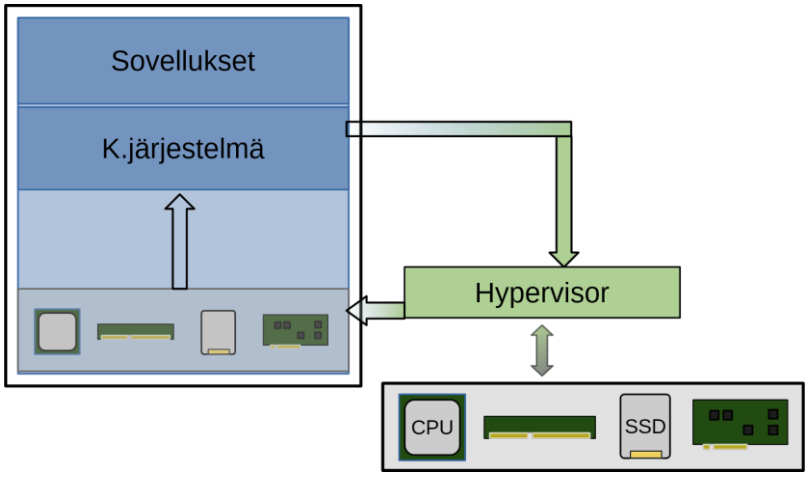
KUVA 16. VMCS:n loogiset ryhmät.

Laitteistoavustettu virtualisointi mahdollistaa virtuaalikoneiden tehokkaan toiminnan poistamalla prosessoriarkkitehtuurista aiheutuvia rajoituksia. Työssä käytetty KVM myös vaatii käyttöönsä prosessorin, joka tukee virtualisointilaajennoksia.

4.3.2 Paravirtualisointi

Kokonaan erilainen lähestymistapa on paravirtualisointi. Sen sijaan, että käyttöjärjestelmälle yritetään ylläpitää illuusiota, paravirtualisoinnissa virtuaalikoneessa olevaa käyttöjärjestelmää on muokattu siten, että se tietää olevansa virtualisoitu. Paravirtualisointi lisää vieraan käyttöön niin kutsutun hyperkutsun, jolla se pyytää hypervisoria suorittamaan tarvittavat käskyt. Periaate on esitetty kuvassa 17.

Toimintatapa on samankaltainen kuin järjestelmäkutsun tapauksessa, jossa ohjelma pyytää käyttöjärjestelmää suorittamaan toiminnon, jota se ei voi itse tehdä. Laitteistoapuja prosessorilta ei tarvita ja binääriä ei tarvitse valvoa, sillä paravirtualisoinnissa vieraat eivät yritä suorittaa ongelmallisia käskyjä. (Barham, Dragovic, Fraser, Hand, Harris, Ho, Neugebauer, Pratt & Warfield 2003, 5.)



KUVA 17. Käyttöjärjestelmä pyytää hypervisoria suorittamaan tehtäviä.

Saavutettavat nopeushyödyt ovat huomattavat, mutta kääntöpuolena on yhteensopivuuden heikkeneminen. Vieraan ydintä on muokattava ja sen on sisällettävä tarvittavat laitteistoajurit paravirtualisointia varten. Työssä käytetään Linuxille saatavilla olevia virtio-paravirtualisointiajureita.

4.4 KVM

Työssä käytettäväksi hypervisoriksi valittiin KVM sen kattavien ominaisuuksien ja laitteistoläheisyyden vuoksi. Toisena vaihtoehtona tutkittiin XEN-hypervisoria, mutta se ei lopulta ollut käytettävissä ympäristön rajoitusten takia.

KVM on Linuxin kernelmoduuli, jonka käyttöönottamalla voi käyttöjärjestelmän ydin toimia tyypin 1 hypervisorina. KVM:n käyttäminen ei estä käyttöjärjestelmän normaalia käyttöä, mutta mahdollistaa virtuaalikoneiden ajamisen ilman tyypin 2 hypervisoria. Moduuleista kvm.ko sisältää yleiset toiminnot ja tämän lisäksi otetaan käyttöön prosessorista riippuen joko kvm-intel.ko tai kvm-amd.ko. Kernelmoduuli sisältyy Linuxin ytimeen versiosta 2.6.20 eteenpäin, eikä sen mukaanotto vaadi kääntämistä lähdekoodista. (Kernel Virtual Machine 2019, viitattu 19.6.2019.)

Kernelmoduuli on jokin koodin tarjoama toiminnallisuus, joka voidaan ottaa tai poistaa käytöstä ytimessä tarpeen mukaan. Moduulien käyttäminen ei vaadi ytimen, siis koko järjestelmän, uudelleenkäynnistystä. Ytimen rakenne säilyy modulaarisena ja koko pienenä, sillä tarpeettomia osia ei tarvitse ladata, eikä koko ydintä tarvitse rakentaa ja ladata uudelleen, kun siihen lisätään jotain. (Salzman, Burian & Pomerantz 2007, 2.)

KVM käsittää hypervisorina toimivan ytimen ja käyttäjätilan QEMU (Quick Emulator) -komponentit. QEMU tarjoaa virtuaaliset laitteistoresurssit joko virtualisoimalla fyysisiä laitteistoresursseja tai emuloimalla uusia. Käytettäessä QEMUa laitteiston emulointiin voidaan ohjelmallisesti luoda laitteistoresursseja, joita ei fyysisessä laitteessa ei ole, esimerkiksi eri käskykantaa käyttävä prosessori. (Main Page 2017, viitattu 20.6.2019.)

4.5 Libvirt-kirjasto

Libvirt tarjoaa kokoelman tehokkaita kirjastoja ja työkaluja virtuaalikoneiden luontiin sekä hallintaan. Libvirt ei ole hypervisor-riippuvainen ja toimii sekä avoimen lähdekoodin että kaupallisilla hypervisoreilla.

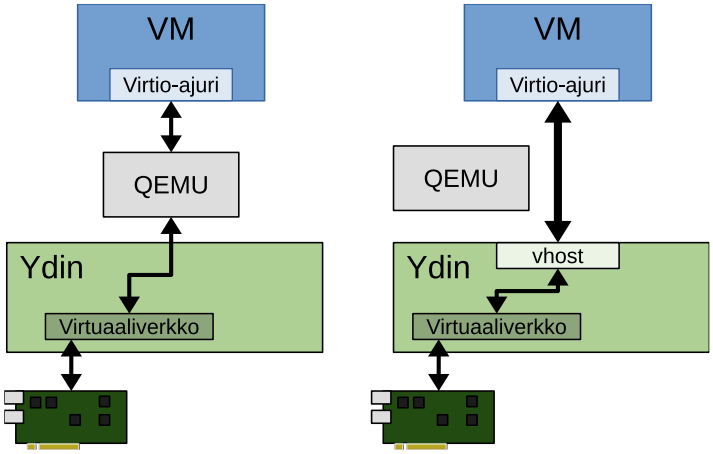
Yksi työkaluista on komentoriviltä käytettävä virsh. Tällä työkalulla virtuaalikoneita voidaan luoda, pysäyttää tai käynnistää ja virtuaalisia laitteistoresursseja tai virtuaaliverkkoja lisätä, muokata tai poistaa. Virsh tulee toimimaan pääasiallisena välineenä tämän työn virtuaalikoneiden hallintaan. (Libvirt FAQ 2017, What is libvirt?, viitattu 20.6.2019.)

4.6 Virtio-arkkitehtuuri

Virtio on yksi libvirtin mukana tulevista kirjastoista ja tarjoaa paravirtualisointiajurit isäntää ja vierasta varten. Paravirtualisointiajurien avulla virtuaalikoneessa oleva laiteresurssi osaa kommunikoida hypervisorin kanssa ja parantaa. Varsinainen emulointi tapahtuu hypervisorin puolella ja vieras saa käyttöönsä rajapinnan, jonka kautta se voi käyttää laitetta. (Jones 2010, viitattu 20.6.2019.)

4.6.1 Virtio-net -ajuri

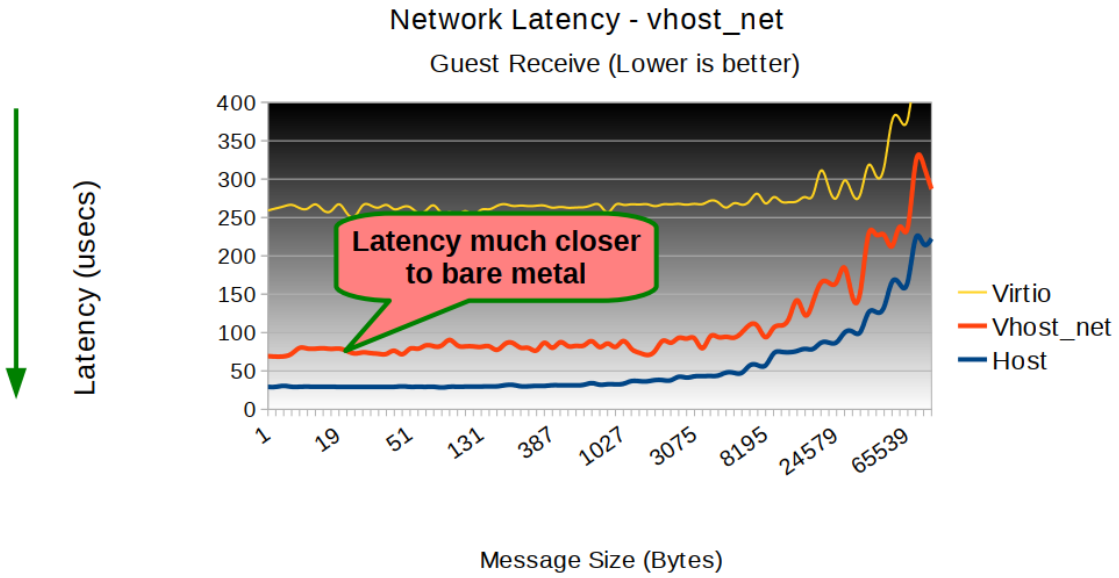
Virtio-net syntyi tarpeesta saada paravirtualisoitu ratkaisu verkkoliikenteelle isännän ja vieraan välillä. Toteutunut suorituskyky jäi kuitenkin heikoksi ratkaisun monimutkaisuuden takia. Liikenne kulki QEMUn läpi, mikä johti jatkuvaan prosessorin tilojen vaihtamiseen ja datan kopioimiseen paikasta toiseen. Jatkokehityksen tuloksena syntyi vhost_net. Periaattellinen ero on esitetty kuvassa 18. (Vhost Sample Application 2019, luku 31.1, viitattu 20.6.2019.)



KUVA 18. Tarpeeton suoritustilojen vaihto vähenee , kun QEMU ohitetaan.

4.6.2 Vhost_net -ajuri

Vhost siirtää virtio taustaohjelman pois käyttäjätilasta, jolloin emulointi tapahtuu isännän ytimen puolella eikä toistuva ydin- ja käyttäjätilan välillä vaihtaminen ole tarpeellista. Paketit välitetään isännän ytimen kautta QEMUn sijaan. Vieraaseen asetettu virtio-ajuri kommunikoi suoraan tämän taustaohjelman kanssa, parantaen verkon suorituskykyä huomattavasti kuvan 19 mukaan. (Vhost Sample Application 2019, luku 31.1, viitattu 20.6.2019.)



KUVA 19. Vhost_net -ajuri on pyritty optimoimaan virtuaalikoneiden verkkoliikenteeseen (Wagner 2011, 17).

5 VIRTUAALIVERKOT

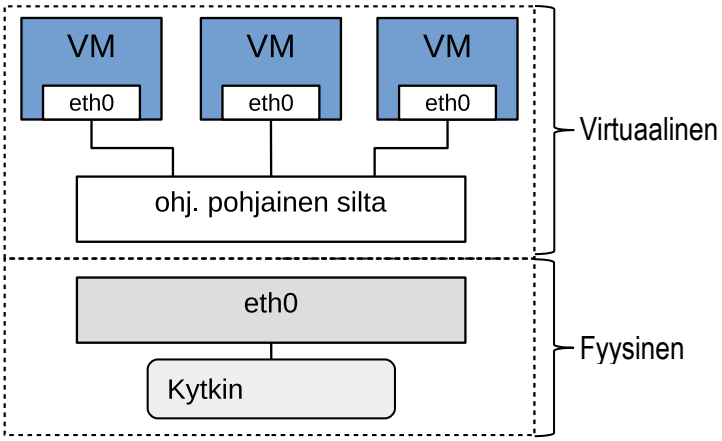
Libvirtin avulla voidaan tehdä erityyppisiä virtuaaliverkkoja. Kuten fyysiset vastineensa, virtuaaliverkot voivat sisältää eri ominaisuuksia kuten osoitteenmunnoksen, reititystä, liikenteen suodatussääntöjä tai virtuaalilähiverkkoja. Virtuaaliverkon asetusten mukaisesti liikenne voi joko kulkea virtuaalisen ja fyysisen verkon välillä tai se voidaan rajata omaksi ympäristökseen isäntäkoneen sisällä.

Virtuaaliverkkoa ei tule sekoittaa virtuaaliseen lähiverkkoon, VLANiin. Siinä missä VLAN käsittää yleisesti 802.1q-standardin ja muut siihen liittyvät konseptit, kuten kytkinporttien loogisen erottelun, voidaan virtuaaliverkkoa ajatella kokonaisena virtuaalisena kytkimenä. Virtuaaliverkko voi sisältää virtuaalisia lähiverkkoja. Virtuaalikoneet voidaan liittää yhteen tai useampaan virtuaaliverkkoon lisäämällä niille virtuaalinen verkkorajapinta ja asettamalla se kuulumaan haluttuun virtuaaliverkkoon.

Libvirtillä luodun virtuaaliverkon tyyppiä vaihtamalla voidaan liikenne johtaa fyysiseen verkkoon eri tavoilla. Alla on mainittuna muutama. Työssä tullaan käyttämään ohjelmistopohjaisia siltoja ja eristettyjä MacVTap-verkkoja.

5.1 Silta

Siltatatila toimii kuin tavanomainen silta Linuxissa ja voidaan toteuttaa puhtaasti ohjelmistopohjaisesti. Se on kuin virtuaalinen kytkin, jonka kautta siihen liitetyt laitteet voivat kommunikoida keskenään. Liikenne voi kulkea isännän ja vieraiden välillä, ja sillan asetusten mukaisesti virtuaalikoneet ovat joko täysin eristettynä ulkopuolisesta verkosta tai osa sitä. Kuvassa 20 on eristetty ratkaisu.



KUVA 20. Ohjelmistopohjaisesta sillasta ei ole yhteyttä fyysiseen verkkorajapintaan.

5.1.1 Osoitteenmuunnos

Osoitteenmuunnosta käyttämällä virtuaalikoneet liittyvät virtuaalikytkimeen ja voivat sen kautta kommunikoida keskenään. Verkolle asetettu osoiteavaruus on käytössä virtuaaliverkon sisäisessä liikenteessä, mutta kommunikoitaessa ulospäin suoritetaan liikenteelle osoitteenmuunnos yksityisestä osoitteesta julkiseen osoitteeseen.

Libvirtin luoma oletusverkko käyttää osoitteenmuunnosta ja onnistuu yleensä tarjoamaan perustason verkkoyhteydet ilman erillistä verkon konfigurointia. Tila mahdollistaa liikenteen ulos virtuaaliverkosta sekä isännän ja vieraiden välillä.

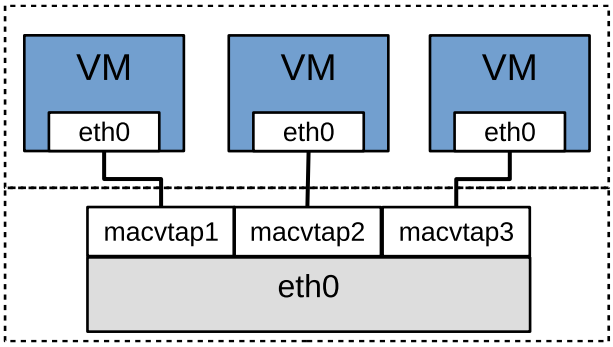
Kääntöpuolena on verkon suorituskyvyn heikkous. Osoitteenmuunnos vie resursseja, kasvattaa verkon viivettä sekä pienentää suorituskykyä.

5.1.2 MacVTap-tila

MacVTap tarjoaa kuvan 21 mukaisen suoran liitynnän isännän verkkokortille. Se on samankaltainen MacVLanin kanssa, missä fyysiseen verkkorajapintaan luodaan uusi alirajapinta omalla MAC-osoitteellaan (Media Access Control). Lisäksi jokaista virtuaalista verkkorajapintaa varten luodaan oma TAP-rajapinta (Terminal Access Point).

Luotu TAP on puhtaasti ohjelmistopohjainen rajapinta, johon vieraat voivat kirjoittaa ja lukea tiedoston kautta. Nämä TAP-rajapinnat liitetään virtuaalikoneen verkkorajapintaan, joilla jokaisella

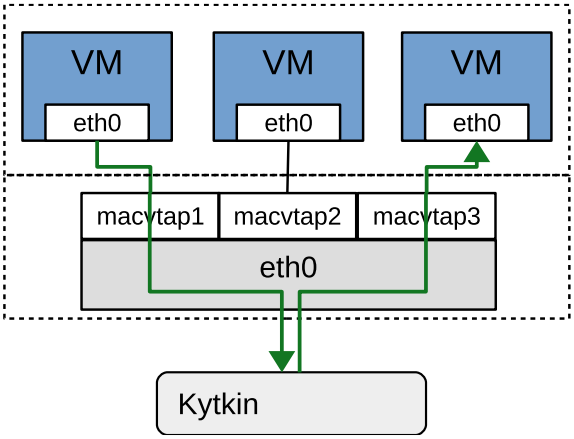
on oma MAC- ja IP-osoitteensa. Fyysisen alirajapintaan liitetyn TAP-rajapinnan MAC-osoite on sama kuin virtuaalikoneen TAP-rajapinnan.



KUVA 21. MacVTap ulottaa virtuaalikoneen verkkorajapinnan lähemmäs isännän verkkokorttia.

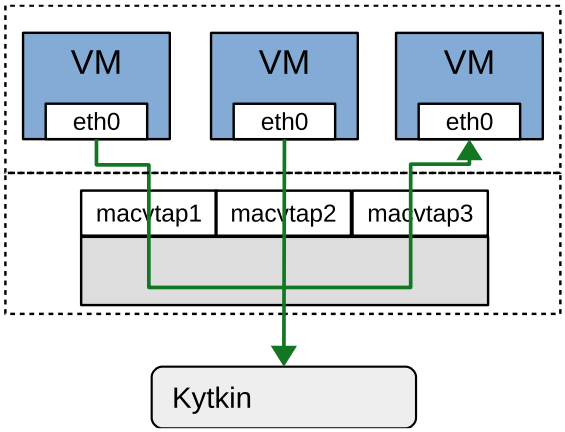
Macvtap tekee verkkoliikenteestä suoraviivaisempaa. Suorituskyky voi olla jopa 50 % parempi kuin ohjelmistopohjaisissa silloissa. Tämä säästää isännän resursseja ja parantaa skaalautuvuutta (MacVTap driver considerations 2019, viitattu 20.6.2019.)

Macvtap, kuten macvlan, voi toimia kolmessa eri tilassa. Kuvan 22 VEPA (Virtual Ethernet Port Aggregator) -tilassa kaikki liikenne lähetetään ulos fyysisestä rajapinnasta. Mikäli virtuaalikoneiden halutaan viestivän keskenään, täytyy verkkokortin kanssa kommunikoivan kytkimen tukea niin kutsuttua neulansilmätilaa (hairpin mode). Neulansilmätilassa kytkin lähettää liikenteen ulos samasta portista, josta se tuli sisään.



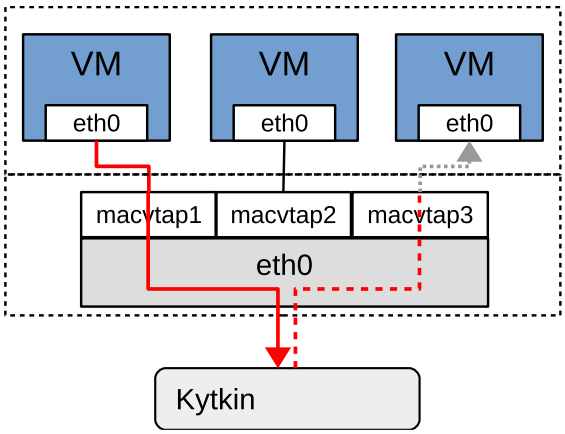
KUVA 22. VEPA-tilan liikenne.

Siltatilassa samaan fyysiseen rajapintaan liitetyt virtuaalikoneet voivat kommunikoida suoraan keskenään ilman ulkoista kytkintä eikä liikennettä lähetetä ulos isännän verkkorajapinnasta, ellei sille ole tarvetta. Periaate on esitettynä kuvassa 23.



KUVA 23. Siltatila sallii virtuaalikoneiden välisen kommunikoinnin ilman ulkoista kytkintä.

Eristetyssä tilassa virtuaalikoneet eivät voi kuvan 24 mukaisesti kommunikoida keskenään saman fyysisen rajapinnan kautta, vaikka ulkoinen kytkin tukisikin neulansilmätilaa.



KUVA 24. Eristetty tila estää samassa verkkorajapinnassa olevien virtuaalikoneiden välisen kommunikoinnin.

Toisin kuin muissa verkkotyypeissä, isännän ja vieraan välinen suora kommunikaatio ei yleensä ole mahdollinen käytettäessä macvtap-ratkaisua. (MacVTap 2017, viitattu 20.6.2019.)

6 KÄYTETTÄVÄT LAITTEET

Yrityksen omien tuotteiden lisäksi testipaikan rakentamista varten tarvitaan kytkimiä, virtuaalikoneita sekä virtualisointiin käytettävä alusta. Luku esittelee työssä käytetyt laitteet lyhyesti.

6.1 Kytkimet

6.1.1 Cisco Catalyst 3560CX-12PC

Catalyst 3560CX-12PC on 12-porttinen L3-tason kytkin VLAN-tuella. Lisänä kytkimessä on neljä porttia ylöspäin menevälle siirtoyhteydelle, joista kaksi erillisellä kuitumoduulilla. Kaikki kytkimen portit kykenevät toimimaan 1 Gbit/s:n tiedonsiirtonopeudella.

Ciscon kytkimiä käytetään virtuaalilähiverkkojen ja niiden runkoyhteyksien luontiin testattavassa verkossa. Yhdelle kytkimistä otetaan lisäksi käyttöön reitityspalvelu ja sen kanssa käytetään OSPF-reititysprotokollaa.

6.1.2 HPE OfficeConnect 1850 24G

OfficeConnect on 24-porttinen kytkin VLAN-tuella ja kahdella 10 Gbit/s ylöspäin menevällä siirtoyhteydellä. Ylöspäin meneviä portteja lukuun ottamatta portit toimivat nopeudella 1 Gbit/s.

Kytkintä käytetään virtuaalikoneiden verkkorajapintojen ulottamiseksi fyysisiin laitteisiin. 10 gigabitin portit tullaan liittämään yhdeksi loogiseksi portiksi runkoyhteyttä varten.

6.1.3 Zyxel ES 2108-G

Zyxelin ES on pieni 8-porttinen kytkin yhdellä ylöspäin menevällä siirtoyhteydellä ja VLAN-tuella. Ainoastaan ylöspäin menevä portti kykenee gigabitin tiedonsiirtonopeuksiin. Loput kytkinporteista toimivat enimmillään nopeudella 100 Mbit/s.

Zyxel liitettiin jälkikäteen mukaan kytkinporttien määrän lisäämiseksi. Osa virtuaalikoneiden ja testipaikalla sijaitsevien laitteiden yhteyksistä ei tarvitse suurta tiedonsiirtokapasiteettia. Nopeampia kytkinportteja voidaan säästää muuta tarkoitusta varten käyttämällä Zyxeliä testipaikan hallintayhteyksiin.

6.2 Taktinen reititin

TR (Tactical Router) on Bittiumin kehittämä ja valmistama armeijan kenttäolosuhteisiin tarkoitettu reititin. Etupaneelista löytyy neljä oletuksena lähiverkossa toimivaa ja neljä oletuksena reititinporttina toimivaa rajapintaa. Kaksi reititinporteista on toteutettu kuituyhteydellä. Kaikki portit kykenevät toimimaan 1 Gbit/s:n tiedonsiirtonopeudella.

Työtä varten kahta taktista reititintä käytetään liikenteen välittämiseksi kolmelle virtuaalikoneista. Yksi taktisista reitittimistä tulee käyttämään OSPF-reititysprotokollaa ja toinen OLSR-protokollaa. Virtuaalilähiverkkoja ei taktisilla reitittimillä tässä työssä käytetä.

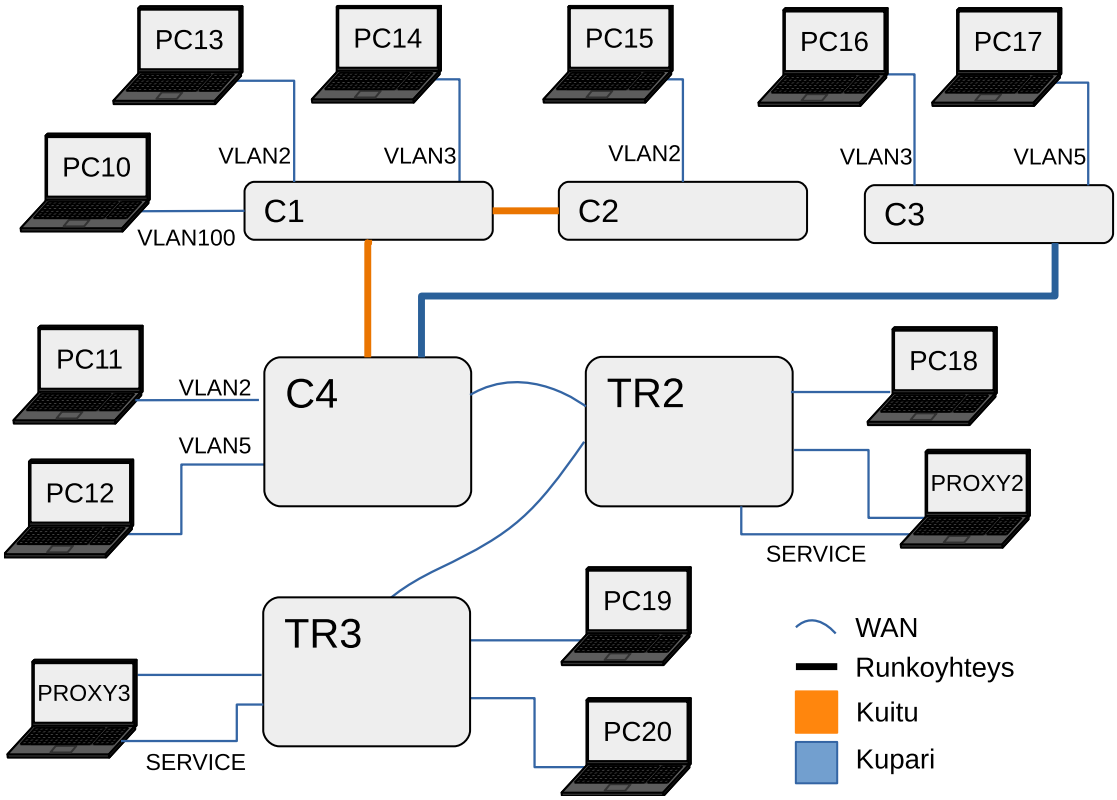
6.3 Tietokoneet

Virtualisointiin käytettävässä tavallisessa työpöytämallin tietokoneessa on 6-ytiminen Intelin i7-8700 prosessori HT-tuella, 32 Gt 2666 MHz DDR4 muistia ja 1 teratavun kokoinen NVMe-massamuisti. Integroidun 1 Gbit:n verkkorajapinnan lisäksi koneeseen on asennettuna PCIe-lisäkortti. Kortissa on kaksi 10 Gbit:n verkkorajapintaa runkoyhteyttä varten.

Virtualisoiduilla tietokoneilla on käytettävissään yksi prosessoriydin, 1 Gt muistia ja 25 Gt tallennustilaa. Verkkorajapinnoissa käytetään vhost_net-tekniikkaa.

7 TYÖN TOTEUTUS

Tavoitteena oli rakentaa kuvan 25 mukainen verkko laitteiston ja ohjelmiston testausta varten. Verkossa C4, TR2 ja TR3 toimivat reitittiminä. Tarkempi kuvaus rakennetusta ympäristöstä on liitteessä 1.



KUVA 25. Testausta varten rakennettu ympäristö.

Laitteet C1–C4, TR2 sekä TR3 ovat fyysisiä ja PC:t 10–20 virtualisoitiin KVM-alustaa käyttäen. Proxykoneita käytettiin pääasiassa taktisten reitittimien ohjaamiseen. Kuvatusta verkosta ei ollut pääsyä ulospäin, joten virtuaalikoneille lisättiin muita verkkorajapintoja tarpeen mukaan. Varsinaisen testiverkon lisäksi tarvittiin verkkoja testipaikan ja -laitteiden hallintaan. Virtualisointia hyödynnettiin sen tarjoaman tehokkaan resurssikäytön ja joustavuuden vuoksi.

Ensimmäinen vaihe sisälsi tarpeellisten ohjelmistojen asentamisen isäntäkoneeseen, HP:n ja ZyXELin kytkimen asetukset sekä isäntäkoneen verkkoasetukset virtuaalikoneita varten. Kytkimille ja isännälle luotiin virtuaalilähiverkot virtuaalikoneita ja testipaikan hallintaa varten. Runkoyhteys isännän ja HP:n kytkimen välille muodostettiin.

Toisessa vaiheessa luotiin tarvittavat virtuaalikoneet. Yksi virtuaalikone asennettiin valmiiksi ja siihen tehtiin yleiset asetukset. Virtuaalikonetta kloonattiin tarpeellinen määrä ja koneet muokattiin tehtäviinsä sopiviksi.

Kolmannessa ja viimeisessä vaiheessa luotiin testattava verkko. Ciscon kytkimille asetettiin virtuaalilähiverkot ja niiden runkoyhteydet. Virtuaalikoneet liitettiin niille tarkoitettuihin virtuaaliverkkoihin tai taktisiin reitittimiin. Reititys verkkojen välillä toteutettiin.

Työselostus ei ole tarkka komento komennolta -ohje ja joitain vaiheita on tästä dokumentista jätetty pois. Selostuksessa on kuitenkin keskeisimmät asiat, jotka on suoritettava ympäristön rakentamiseksi. Yksityiskohtaisempi dokumentaatio on tuotettu yrityksen käyttöön.

7.1 Ensimmäinen vaihe

7.1.1 Isäntäkoneen asennus

Virtualisointialustana käytettävä isäntäkone on valmisteltava virtuaalikoneiden asennusta varten. Ensin hankitaan paketinhallinnasta työkalut Linuxin siltojen hallintaan, QEMU-käyttäjätilan komponentit sekä libvirt komennolla `sudo apt-get install bridge-utils qemu-kvm libvirt-bin`.

Virtuaalikoneiden luontiin ja hallintaan tullaan pääasiallisesti käyttämään komentorivityökalua virsh. Graafinen käyttöliittymä ja täysi työpöytänäkymä virtuaalikoneeseen on kuitenkin oiva apu, joten isäntään asennetaan vielä virt-manager komennolla `sudo apt-get install virt-manager`.

Virtuaalikoneilla voidaan joko käyttää perinteistä laiteohjelmistoa, BIOSia, tai uudempaa UEFIa. Työssä luotavat virtuaalikoneet käyttävät UEFIa, joten asennetaan tarvittava komponentti. `sudo apt-get install ovmf`.

Koneiden kloonaamisessa tarvitaan työkalu yksilöimään koneet. Ilman yksilöintiä koneet todellakin ovat klooneja ja identtisiä keskenään. Niillä on samat nimet, samat SSH-avaimet ja niin edel-

leen. Tarvittava työkalu saadaan komennolla `sudo apt-get install libguestfs-tools`.

Työssä tarvittavien komponenttien pitäisi nyt olla asennettuna. Ennen seuraavaan vaiheeseen siirtymistä tehdään vielä muutama muutos. Hallinnan selkeyden vuoksi luodaan pooli, johon asennusmediat virtuaalikoneita varten myöhemmin siirretään.

```
virsh pool-define-as --name install-media --type dir --target
<path/to/folder>
```

Otetaan pooli käyttöön ja merkitään se tapahtumaan jatkossa automaattisesti.

```
virsh pool-start install-media
virsh pool-autostart install-media
```

Virtuaalikoneille tehdään myös oma poolinsa. Tätä sijaintia käytetään virtuaalikoneiden levykuvien varastointiin.

```
virsh pool-define-as --name guests --type dir --target
<path/to/folder>
virsh pool-start guests
virsh pool-autostart guests
```

Isäntäkone on teknisesti valmis virtualisointia varten. Virtuaalikoneiden asennusta ei seuraavaksi kuitenkaan vielä suoriteta, vaan ensin valmistellaan tarvittavat verkot.

7.1.2 Virtuaaliverkko

Testiverkon virtuaalikoneet liitetään toisesta verkkorajapinnastaan eristettyyn virtuaaliverkkoon, jonka kautta koneita voidaan hallita yhdestä osoiteavaruudesta. Isäntäkonetta ei myöskään tulla liittämään testattavaan verkkoon, joten kommunikointi sen kautta ei ole mahdollista tai edes suotavaa.

Virsh-työkalulla voidaan luoda uusia verkkoja määrittelemällä ne ensin XML-tiedostoon. Luodaan minimaalinen XML-tiedosto yksinkertaisen siltaavan verkon määrittelyä varten.

```
<network>
     <name>ctrl</name>
     <bridge name=’privatebr0’/>
     <ip address=’192.168.2.1’ netmask=255.255.255.0/>
          <dhcp>
               <range start=’192.168.2.2’ end=’192.168.2.254’/>
          </dhcp>
     </ip>
</network>
```

Verkko luodaan, otetaan käyttöön ja asetetaan käyttöönotto tapahtumaan jatkossa automaattisesti suorittamalla komennot

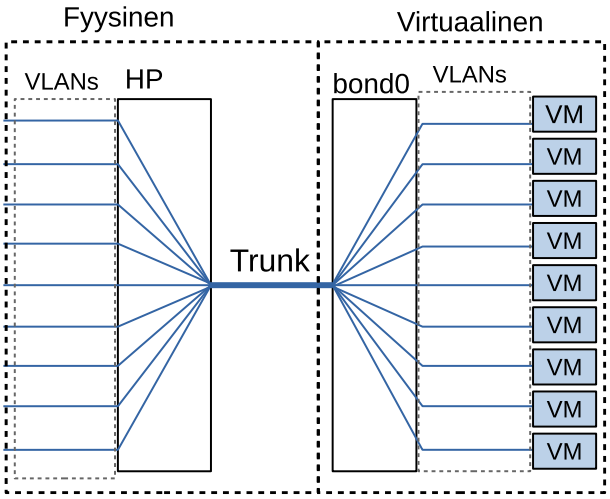
```
virsh net-define <xml_tiedosto>
virsh net-start ctrl
virsh net-autostart ctrl
```

Tuloksena on yksinkertainen ohjelmistopohjainen silta ilman mitään erityisiä sääntöjä. Mikäli virtuaaliverkkoon liitettävät laitteet on määritetty käyttämään automaattista osoitteenmääritystä, ne saavat osoitteen väliltä 192.168.2.2–192.168.2.254. Muutoin kiinteä osoite on määritettävä kuulumaan verkkoon 192.168.2.0/24. Mikäli SSH-palvelu on käytössä, luotu virtuaaliverkko mahdollistaa suoran kommunikaation isännän ja virtuaalikoneiden välillä.

Tarpeen vaatiessa isäntäkoneelle voidaan avata SSH-tunneli liikenteen välittämiseksi eteenpäin, jolloin virtuaalikoneet voivat myös käyttää paketinhallintaa. Nykytilassaan luodusta verkosta ei voi liikennöidä ulos, mikä sopii käyttötarkoitukseen hyvin.

7.1.3 Yhteydet fyysisiin laitteisiin

Virtuaalikoneiden liittämiseksi osaksi fyysistä testilaitteistoa on liikenteen kyettävä kulkemaan fyysisen ja virtuaalisen maailman välillä. Liikenne on myös kyettävä kuljettamaan ulos siten, että se on täysin eristettynä muista laitteista tiettyyn pisteeseen asti. Perusideana on, että jokainen HP:n ja ZyZELin kytkimen porteista toimii liityntäpisteenä suoraan sen parina olevan virtuaalikoneen verkkorajapintaan kuvan 26 mukaisesti. Kytkinportin ja virtuaalikoneen välillä liikenne ei saa kulkeutua mihinkään toiseen virtuaalikoneeseen tai verkkoon.



KUVA 26. Virtuaalilähiverkkoja käytetään virtuaalisten verkkorajapintojen ulottamiseksi fyysisiin kytkinportteihin.

Virtuaalikoneita tarvitaan toistakymmentä, mutta isäntäkoneessa on tarkoitusta varten käytettävissä vain kaksi fyysistä verkkorajapintaa. Lisäksi kaikkien samassa osoiteavaruudessa olevien virtuaalikoneiden liittäminen samaan kytkimeen ”oikosulkee” liikenteen, eikä se kierrä testattavan järjestelmän läpi. Yksi keino on lisätä fyysisten verkkorajapintojen määrää isäntäkoneessa. Ratkaisu ei kuitenkaan ole joustavin mahdollinen ja rajapintoja tarvittaisiin suuri määrä. Parempi ratkaisuvaihtoehto molempiin ongelmiin löytyy virtuaalilähiverkoista.

Asentamalla Linuxiin tarvittava komponentti ja määrittämällä verkkorajapinnan asetukset voidaan fyysinen liitäntä muuttaa virtuaaliverkkoja tukevaksi sillaksi. Luotuun siltaan voidaan luoda edelleen useita eri virtuaalilähiverkkoja ja virtuaalikoneet kiinnittää näihin MacVTap-ratkaisua käyttäen.

Kun fyysinen verkkorajapinta on siltaavassa tilassa ja virtuaalikoneet käyttävät MacVTapia, on liikenne eristetty isäntäkoneesta. Kun lisäksi käytetään virtuaalilähiverkkoja, on virtuaalikoneiden liikenne eristetty toisistaan. Käytetyllä ratkaisulla virtuaalikoneet on helppo liittää yksilöllisesti haluttuihin kohteisiin, aivan kuin ne olisivat oikeita fyysisiä koneita.

Koska runkoyhteys tulee kuljettamaan liikennettä usealta virtuaalikoneelta, tarvitaan yhteydeltä suurta tiedonsiirtokapasiteettia. Yleisimmät tietokoneiden ja kytkimien portit kykenevät siirtämään dataa 1 Gbit:n sekuntivauhtia. Mikäli kapasiteetti jaetaan tasaisesti, riittää siitä yhdelle virtuaalikoneelle vain noin 8 megatavua sekunnissa. Kaikki virtuaalikoneet eivät aina lähetä samanaikai-

sesti, mutta suurempi ongelma on, että jo yksi virtuaalikone voi käyttää koko runkoyhteyden kapasiteetin.

Ongelman välttämiseksi isäntäkoneessa ja kytkimessä on kaksi 10 gigabitin porttia. Portit on lisäksi yhdistetty aggregointia käyttäen ja ne muodostavat yhden 20 Gbit/s:n tiedonsiirtonopeuteen kykenevän loogisen runkoyhteyden.

Runkoyhteyden luomiseksi isäntäkoneen puolella asennetaan ensiksi paketinhallinnasta tarvittava moduuli komennolla `sudo apt-get install vlan ifenslave`. Toiminnan varmistamiseksi kaksi moduulia ja niiden käynnistysjärjestys määritetään /etc/modules-tiedostoon.

```
bridge
bonding
8021q
```

Siltaavien verkkorajapintojen ei haluta tekevän minkäänlaista suodatusta liikenteelle. Muokkaamalla /etc/sysctl.conf-tiedostoa voidaan suodatus ottaa pois käytöstä.

```
net.bridge.bridge-nf-call-ip6tables = 0
net.bridge.bridge-nf-call-iptables = 0
net.bridge.bridge-nf-call-arptables = 0
net.bridge.bridge-nf-filter-pppoe-tagged = 0
net.bridge.bridge-nf-filter-vlan-tagged = 0
```

Säännöt otetaan käyttöön käynnistyksen yhteydessä lisäämällä /etc/rc.local-tiedostoon rivi

```
/sbin/sysctl -p /etc/sysctl.conf
```

Tarvittavat valmistelut isäntäkoneen linkkien aggregointia ja virtuaalilähiverkkojen käyttämistä varten on nyt tehty. Seuraavaksi voidaan määritellä itse runkoyhteys ja verkkorajapinnat, joihin virtuaalikoneet liitetään. Verkkorajapintoihin liittyvät asetukset tehdään /etc/network/interfaces-tiedostoon. Alla on esitettynä tiedoston sisältö lyhennetyssä muodossaan.

```
auto lo
    iface lo inet loopback
# =================
# verkkorajapinnat
# =================
#yhteys labraverkon suuntaan
auto eth0
    iface eth0 inet dhcp

#eth1 ja eth2 on alistettu osaksi bond0 rajapintaa
auto eth1
iface eth1 inet manual
    bond-master bond0
auto eth2
```

```
iface eth2 inet manual
    bond-master bond0

#yksi 20Gb looginen linkki
auto bond0
iface bond0 inet manual
    bond-mode 4
    bond-miimon 100
    bond-lacp-rate 1
    bond-slaves eth1 eth2

# ===================
# virtuaalilähiverkot
# ===================

#PC11
auto bond0.1
    iface bond0.1 inet manual
    vlan-raw-device bond0

# .
# .
# .

#PROXY3 TR service lan
auto bond0.29
    iface bond0.29 inet manual
    vlan-raw-device bond0
```

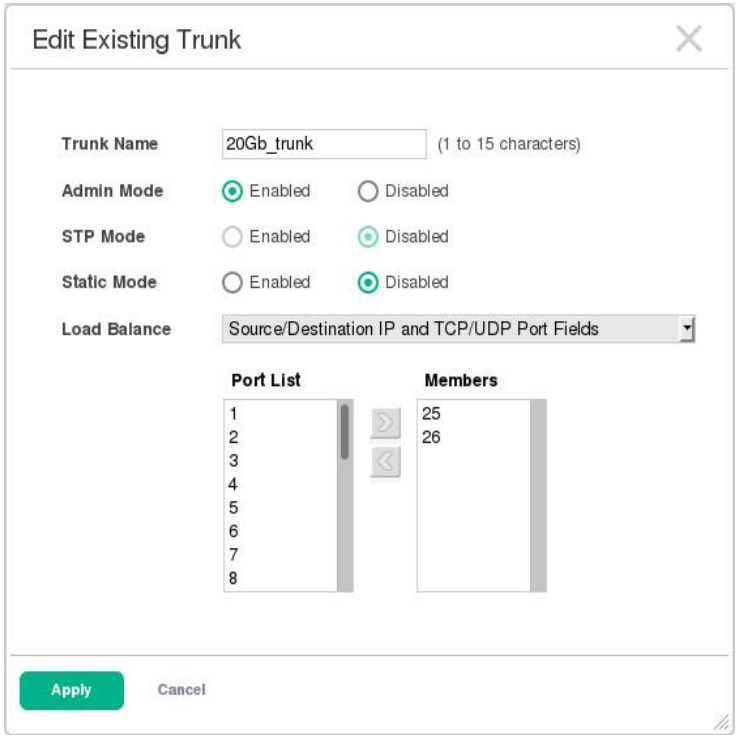
Bond0:aan määritetty bond-mode 4 käskee linkkiä käyttämään IEEE 802.3ad:n mukaista aggregointia. Miimon-parametri määrää millisekunneissa, kuinka usein fyysisten linkkien tilaa tarkkaillaan. Lacp-rate 1 pyytää lähettämään aggregointiin liittyviä protokollapaketteja sekunnin välein. (Davis ym. 2011, kappale 2.)

Asetuksilla luodaan tarpeelliset virtuaalilähiverkot, eikä niitä ole erityisesti rajoitettu mihinkään virtuaalikoneeseen. Kommentit on lisätty selkeyttämään suunniteltua toteutusta. Käyttöönoton jälkeen virtuaalilähiverkot ovat nähtävissä esimerkiksi komennolla `cat /proc/net/vlan/conf`.

Tarvittavat verkot on nyt luotu isännän puolella. Siirrytään virtuaaliympäristön toiseen päähän, HP:n ja ZyXELin kytkimiin.

7.1.4 HP:n ja ZyXELin asetukset

Aggregoinnin käyttämiseksi yhteyden molemmat päät on konfiguroitava samalla tavalla. HP:n asetuksia varten käytettävissä on ainoastaan selainpohjainen graafinen käyttöliittymä, mikä löytyy oletuksena osoitteesta 192.168.1.1. Kuten tekstissä on aiemmin mainittu, eri valmistajat käyttävät termiä trunk eri tarkoituksiin. Tässä tehtävä trunk on oikeammin link aggregation, eikä se tee yhteydestä automaattisesti virtuaalilähiverkon runkoyhteyttä. Luodaan TRK1 kuvan 27 mukaisesti. Tuloksena 20 gigabitin runkoyhteys PC:n ja kytkimen välillä on nyt toiminnassa.

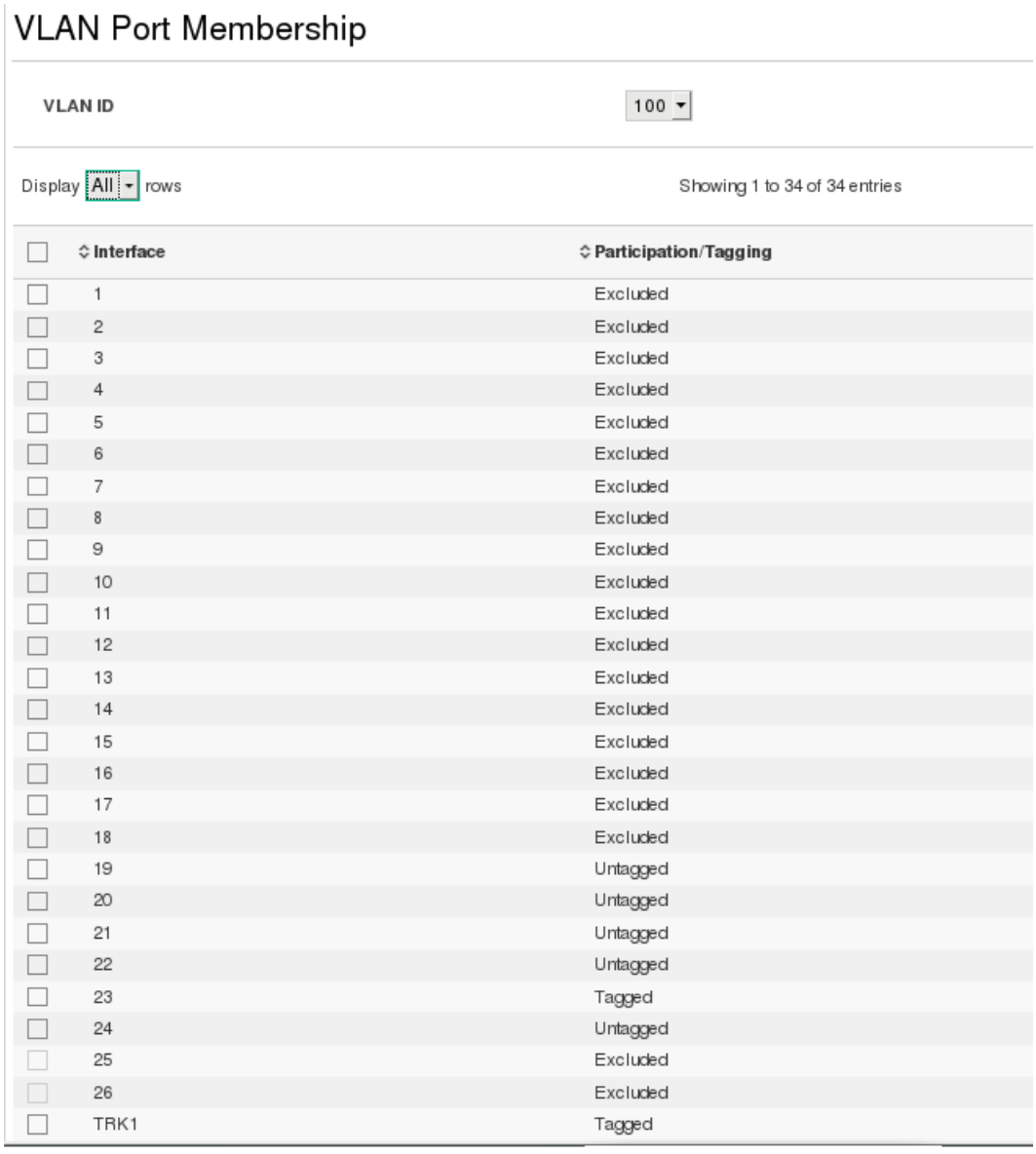


KUVA 27. Porttien aggregointi HP:n kytkimellä.

Kytkinportit on vielä jaettava omiin virtuaalilähiverkkoihinsa. Ensiksi kytkimelle on luotava tarpeelliset virtuaalilähiverkot, jonka jälkeen fyysiset portit voidaan jakaa halutusti. Virtuaalilähiverkko 1 on jo olemassa, joten uutena on lisättävä 2–18, 27–30 ja 100. Virtuaalilähiverkko 100 on valittu HP:n ja ZyXELin kytkinten hallintaa varten.

Kytkinportit on jaettava luotuihin virtuaalilähiverkkoihin. Portit 1–18 asetetaan kuulumaan sen numeroa vastaavaan virtuaaliverkkoon. Poikkeuksen muodostavat portit 19–22 ja 24, jotka liitetään virtuaalilähiverkkoon 100. Kaikkien näiden porttien tilaksi asetetaan ”untagged”. Virtuaalilähiverkon 100 asetus on esillä kuvassa 28.

Untagged-portit käyttäytyvät kuin normaali kytkinportti ja liikenne ulos niistä ei sisällä virtuaalilähiverkon tunnistetta. Luotu TRK1 ja portti 23 sitä vastoin asetetaan kuljettamaan liikennettä ”tagged”-tilassa ja ne toimivat virtuaalilähiverkon runkoyhteyksinä, kuljettaen liikennettä sisään ja ulos kaikista niille määritetyistä virtuaalilähiverkoista. Portti TRK1 kuuluu kaikkiin verkkoihin ja sisältää siis verkot 1–18, 27–30 ja 100. Portti 23 toimii runkoyhteytenä ZyXELin kytkimelle ja kuljettaa liikennettä virtuaalilähiverkoista 27–30 sekä 100.



VLAN Port Membership

VLAN ID 100

Display All rows Showing 1 to 34 of 34 entries

Interface	Participation/Tagging
1	Excluded
2	Excluded
3	Excluded
4	Excluded
5	Excluded
6	Excluded
7	Excluded
8	Excluded
9	Excluded
10	Excluded
11	Excluded
12	Excluded
13	Excluded
14	Excluded
15	Excluded
16	Excluded
17	Excluded
18	Excluded
19	Untagged
20	Untagged
21	Untagged
22	Untagged
23	Tagged
24	Untagged
25	Excluded
26	Excluded
TRK1	Tagged

KUVA 28. Virtuaalilähiverkkoon 100 kuuluvat portit.

Kaikkia HP:n kytkinportteja ei haluttu täyttää heti, vaan muutama varattiin myöhempää käyttöä varten. Varastosta löytyi virtuaalilähiverkkoja tukeva vanha ZyXELin kytkin, jota voitiin käyttää

porttimäärän kasvattamiseen. ZyXELin portit toimivat enimmillään vain 100 Mbit/s, mutta ne riittivät hyvin huoltoyhteyksiin ja testipaikan muun laitteiston ohjaamiseen.

ZyXELin hallinta sijoitettiin samaan osoiteavaruuteen HP:n kanssa ja sen osoitteeksi määrättiin 192.168.1.2. Yksi ZyXELin porteista, portti 9, asetettiin toimimaan runkoyhteytenä ja portit 1–4 virtuaalilähiverkkoihin 27–30. Jäljelle jäävät 4 porttia liitettiin testipaikkaverkkoon 192.168.1.0/24 eli virtuaalilähiverkkoon 100.

Ensimmäinen vaihe on nyt suoritettu ja ympäristö on valmis virtuaalikoneita varten. Luodaan seuraavaksi ensimmäinen virtuaalikone ja käytetään sitä pohjana muiden kloonaamiseen.

7.2 Toinen vaihe

7.2.1 Mallikoneen asennus

Asentamalla yksi virtuaalikone ja suorittamalla siihen kaikki yleiset alustustoimenpiteet voidaan sitä käyttää myöhemmin pohjana muiden virtuaalikoneiden kloonaamisessa. Työ nopeutuu ja helpottuu, kun identtisiä vaiheita ei tarvitse toistaa useaan kertaan. Tässä työssä virtuaalikoneessa käytettiin Debian9-Linux-jakelua.

Alla olevalla komennolla määritetään virtuaalikoneen keskeisiä parametreja ja käynnistetään asennus. Selkeyden vuoksi komento on jaettu useammalle riville. Osa parametreista ei myöskään ole pakollisia, mutta ne halutaan varmuudeksi määritellä johtuen käytetyn ohjelmiston vanhemmasta versiosta. Määrittämällä asetukset käsin voidaan varmistua niiden oikeasta tilasta. Vanhemman version oletusasetukset voivat poiketa halutusta tai jotkin asetukset voivat tunnistautua väärin.

```
virt-install \
--connect quemu:///system \
--virt-type kvm \
--accelerate \
--boot uefi \
--ram 1024 \
--vcpus 1 \
--disk
pool=guests,size=25,bus=virtio,sparse=false,format=raw,cache=non
e \
```

```
--network default,model=virtio \
--cdrom <path/to/install-media> \
--name template \
--os-type linux \
```

Käyttöjärjestelmän asennusprosessi on identtinen tavanomaisen, suoraan laitteistolle tehtävän, asennuksen kanssa eikä sitä käydä läpi tässä. Asennettaviksi komponenteiksi valitaan vain välttämättömät levytilan säästämiseksi ja käytettäväksi ikkunointijärjestelmäksi valitaan Xfce sen vähäisen resurssitarpeen vuoksi.

Kun käyttöjärjestelmän asennus on suoritettu, täytyy vieraalle tehdä samankaltaiset toimenpiteet kuin isännälle. Minimaalisen asennuksen vuoksi monet komponentit on asennettava itse. Oletusverkkoon liitetyllä virtuaalikoneella on pääsy paketinhallinnan suuntaan osoitteenmuunnoksen kautta ilman tarvetta SSH-tunnelille. Kuitenkin paketinhallinnan käyttämiseksi välityspalvelin on määritettävä tiedostoon `/etc/apt/apt.conf`. IP-osoitetta ei tässä voida näyttää.

```
Acquire {
     http { Proxy “http://xx.xx.xx.xx:xx/”; }
}
```

Muutetaan myös vieras käyttämään ethN-nimeämistapaa verkkorajapinnoissa. Muokataan tiedostoa /etc/default/grub ja lisätään parametriin `GRUB_CMDLINE_LINUX` arvot `net.ifnames=0`, `biosdevname=0` ja lisäksi `console=ttyS0`. Viimeinen arvo mahdollistaa tekstipohjaisen konsolin avaamisen virtuaalikoneeseen.

Muita tapoja käyttää virtuaalikonetta ovat täydellinen näkymä virt-managerin tai virt-viewerin kautta sekä normaali SSH-yhteys. SSH:n käyttäminen vaatii, että virtuaalikone on liitetty sellaiseen verkkoon, jota myös isäntä voi käyttää, ja SSH-palvelut ovat käytössä. Tekstipohjaisella konsolilla tai graafisella näkymällä ei ole näitä rajoitteita, joten se on aktiivinen välittömästi virtuaalikoneen käynnistyksen jälkeen eikä vieraan tarvitse olla käyttövalmis. Lopuksi otetaan muutokset käyttöön komennolla `sudo update-grub`. Asetetaan vielä verkkoasetukset /etc/network/interfaces-tiedostosta.

```
auto lo
     iface lo inet loopback

auto eth0
     iface eth0 inet dhcp
```

7.2.2 Kloonaus

Ympäristössä tarvitaan kahdentyyppisiä virtuaalikoneita. Suurempi ryhmä kattaa kolmestatoista koneesta yksitoista, joiden tehtävänä on toimia testattavan järjestelmän päätelaitteina. Loput kaksi tarvitaan testiautomaatiota varten ja esimerkiksi taktisia reitittimiä ohjataan niiden kautta. Erottelu tehdään, sillä virtuaalikoneet halutaan rajata kontrolloidusti omiin tehtäviinsä. Esimerkiksi mainittujen kymmenen ainoa tarkoitus on edustaa asiakkaan käyttämiä laitteita eikä mitään muuta.

Koneita tarvitaan useita, ja mallikoneen asennuksessa tehtävien toimenpiteiden toistaminen on hidasta sekä lisää inhimillisen virheen mahdollisuutta. Tuoretta virtuaalikonetta käytetään pohjana muiden tarvittavien virtuaalikoneiden luontiin.

Kloonaustyökalussa on kriittinen vika. Ongelma on korjattu virt-managerin versiossa 1.3.1-1 el7, mutta työhön käytettävissä oleva versio on tätä vanhempi. Vian seurauksena käytettäessä UEFI-laiteohjelmistoa virtuaalikoneissa ei NVRAM-tiedostoa kloonata oikein. Käytännössä kaikki virtuaalikoneet käyttävät samaa tiedostoa, mikä aiheuttaa ongelmia ennemmin tai myöhemmin. Välittömämpi vika ilmenee, jos yksikään klooneista poistetaan. Kloonin mukana poistuu myös siihen liitetty NVRAM-tiedosto, mikä tekee kaikista samaa pohjaa käyttävistä virtuaalikoneista käyttökelvottomia: ne eivät käynnisty.

Edellä mainituista syistä on tarpeellinen tiedosto kopioitava käsin. Kun uusi virtuaalikone on kloonattu komennolla `virt-clone --original template --name <newname> --file <path/to/guests/pool>`, kopioidaan myös NVRAM ja otetaan kopio käyttöön ajamalla komennot `sudo cp /var/lib/libvirt/quemu/nvram/template_VARS.fd /var/lib/libvirt/quemu/nvram/<newname>_VARS.fd` ja `virt-xml <newname> --edit --boot nvram=/var/lib/libvirt/quemu/nvram/<newname>_VARS.fd`

Sysprep-työkalulla voidaan määrittää monia asetuksia ja valita yksilöitävät tiedot. Tehtävät operaatiot ovat nähtävissä `virt-sysprep --list-operations`-komennolla ja oletuksena valittuna on lähes kaikki mahdollinen. Käytetään sysprep-työkalua kloonattuun virtuaalikoneeseen

oletusasetuksilla ja muutetaan samalla sen nimi samaksi domain-nimen kanssa. `virt-sysprep --domain <newname> --hostname <newname>`.

Virsh- tai virt-manager -työkalussa näkyvät virtuaalikoneiden nimet ovat niin kutsuttuja domain-nimiä. Näitä nimiä käytetään ainoastaan erottelemaan virtuaalikoneet toisistaan ja nimi, joka vieraan käyttöjärjestelmällä on käytössään, voi poiketa tästä.

Klooneja ei haluta käyttää oletusverkossa, vaan ne liitetään virtuaaliverkkoon. Muutos tapahtuu helposti komennolla `virt-xml <newname> --edit --network network='ctrl'`.

Kloonattu kone on nyt valmis ensimmäiseen käynnistykseen. Käynnistetään virtuaalikone komennolla `virsh start <newname>` ja avataan tekstipohjainen konsolinäkymä komennolla `virsh console <newname>`. SSH-session käyttäminen ei ole vielä mahdollista, sillä sen asetukset ovat puutteelliset.

Virtuaalikoneen eth1-verkkorajapintaan halutaan määrittää kiinteä IP-osoite ympäristön vakauden varmistamiseksi. Muokataan jo tuttua tiedostoa /etc/network/interfaces sisältämään osoitteenmääritys. Olkoon esimerkkinä ensimmäinen kone kahdestatoista, PC11, ja sen osoitteen viimeinen oktetti 11. Myöhempää tarvetta varten määritetään myös eth0.

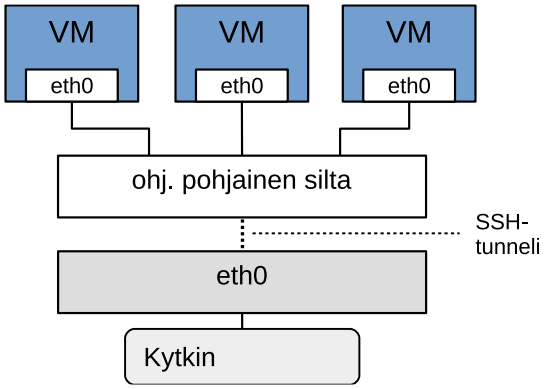
```
auto lo
     iface lo inet loopback

auto eth0
iface eth0 inet dhcp

auto eth1
iface eth1 inet static
     address 192.168.2.11
     netmask 255.255.255.0
     gateway 192.168.2.1
```

Virtuaaliverkossa oleville koneille /etc/apt/apt.conf-tiedoston välityspalvelin on asetettava eri osoitteeseen. Osalle virtuaalikoneista kerrotaan välityspalvelimeksi niiden virtuaaliverkon sillan osoite 192.168.2.1.

```
Acquire {
     http { Proxy "http://192.168.2.1:80/"; }
}
```

Lopuksi luodaan vielä vieraalle SSH-avaimet komennolla `ssh-keygen -f id_rsa -t rsa -N ’’` ja käynnistetään palvelu uudelleen komennolla `dpkg-reconfigure openssh-server`. Nyt virtuaalikoneelle voidaan ottaa SSH-yhteys tavalliseen tapaan.

Avaamalla tai sulkemalla kuvan 29 mukainen SSH-tunneli isäntäkoneella sillan osoitteesta eteenpäin voidaan helposti kontrolloida virtuaalikoneiden pääsyä ulos omasta eristetystä ympäristöstään. Tunnelin avaamiseksi suoritetaan isäntäkoneessa komento `sudo ssh –f –L 192.168.2.1:80:xx.xx.xx.xx:xx –N <knimi>@localhost`.



KUVA 29. Eristetystä virtuaaliverkosta päästään ulos SSH-tunnelia pitkin.

Kloonaus kattaa useita vaiheita ja vaikka se on täydellistä asennusta nopeampaa, vaiheiden toistaminen jokaisen vieraan kohdalla tekee koneiden luomisesta hidasta. Kloonaus on kuitenkin helppo automatisoida bash-skriptejä käyttäen. Tarpeeseen kirjoitettu skripti tekee kaikki luvussa aiemmin mainitut kloonaukseen liittyvät vaiheet ja tuloksena on käyttövalmis, yksilöity virtuaalikone, jonka yleiset asetukset on tehty valmiiksi.

Kaikki isäntäkoneessa suoritettavat komennot voidaan ajaa skriptillä suoraan. Vieraan käyttöjärjestelmässä suoritettaviin komentoihin käytetään sysprep-työkalun toimintoa. Työkalulla virtuaalikoneen ensimmäisen käynnistyksen yhteyteen voidaan määrätä ajettava skripti, jonka vieras suorittaa ensimmäisen käynnistyksen yhteydessä.

Skriptit clone_and_prep_VM.sh ja ifacesetup.sh ovat liitteenä 2 ja 3. Käynnistämällä clone_and_prep_VM-skriptin tapahtuvat tarvittavat toimenpiteet automaattisesti.

Skripti ajetaan 12 kertaa muuttaen aina uuden koneen nimi halutuksi. Käyttämällä nimen lopussa numeroa virtuaalikoneen tunnisteena asettuu eth1:n IP-osoitteen viimeinen oktetti automaattisesti

samaksi. Saman numeron käyttäminen virtuaalikoneen tunnisteena ja viimeisenä oktettina selkeyttää virtuaalikoneiden hallintaa.

Proxyt eivät seuraa samaa IP-osoitteiden nimeämiskäytäntöä ja ne liittyvät eri verkkoihin. Ifacesetup-skriptin parametrit asetetaan proxyja varten manuaalisesti ennen kloonausta.

Seuraavaksi liitetään jokainen virtuaalikone sille suunniteltuun virtuaalilähiverkkoon. PC:t 10–20 ovat jo yhteydessä toisiinsa 192.168.2.0/24-osoiteavaruudessa olevan ohjelmistopohjaisen kytkimen kautta. Virtuaalikoneille lisätään nyt toinen verkkorajapinta, jonka osoite tulee kuulumaan taktisen reitittimen lähiverkkoon ja osoite vastaanotetaan DHCP-palvelusta.

Liittämistä varten ei luoda uutta ohjelmistopohjaista siltaa, vaan se toteutetaan MacVTap-tekniikkalla. Liittäminen tapahtuu samaan tapaan kuin virtuaaliverkon luominen. Tehdään uusi XML-tiedosto, johon määritetään verkkorajapinnan ominaisuudet. Source dev vastaa virtuaalikoneelle suunniteltua virtuaalilähiverkkoa. Esimerkkinä PC11.

```
<interface type='direct'>
     <source dev='bond0.1' mode='private'>
     <model type='virtio'/>
</interface>
```

Verkkorajapinta lisätään ajamalla komento `virsh attach-device PC11 <xml> --config`. Nyt PC11 on käytettävissä HP:n kytkimen portin 1 kautta. Prosessi toistetaan jäljellä oleville virtuaalikoneille.

Tarvittavat kaksi proxykonetta saavat hieman eri asetukset. Jokainen tarvitsee kolme verkkorajapintaa. Ensimmäinen, eth0, liittyy suoraan taktisen reitittimen lähiverkkoon. Eth1 on yhteydessä testipaikan sisäiseen verkkoon ja eth2 on suorassa yhteydessä taktisen reitittimen huoltoporttiin. Verkkorajapinnat on vielä listattu taulukossa 2.

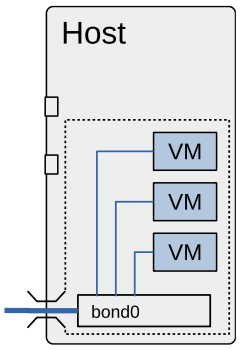
Proxyjen /etc/apt/apt.conf tiedostoon välityspalvelimen osoitteeksi asetetaan 192.168.1.100. Osoite on yksi isäntäkoneen verkkorajapinnoista, ja SSH-tunnelin avaamalla virtuaalikoneet voivat käyttää paketinhallintaa.

TAULUKKO 2. Virtuaalikoneiden verkkorajapinnat.

	Proxyt 2 ja 3	**PC:t 10–20**
eth0	TR:n lähiverkko	TR:n lähiverkko
eth1	Testipaikan sisäinen verkko 192.168.1.0/24	Testipaikan sisäinen virtuaaliverkko 192.168.2.0/24
eth2	TR huoltoyhteys	

7.2.3 Isännän verkkoyhteys

Isäntäkone ei vielä tässä vaiheessa ole osa testipaikkaverkkoa. Isännän verkkorajapintojen eth1 ja eth2 muodostama bond0 ja sen virtuaalilähiverkot eivät ole isännän verkkoliikenteen käytössä. Kuva 30 havainnollistaa ratkaisua. Liittäminen testipaikkaan halutaan tehdä erillisen fyysisen rajapinnan kautta, mutta vapaita verkkorajapintoja ei ole. Tarkoitusta varten isännälle lisätään USB-porttiin kytkettävä Ethernet-adapteri.



KUVA 30. Bond0 ei ole suoraan isännän käytettävissä.

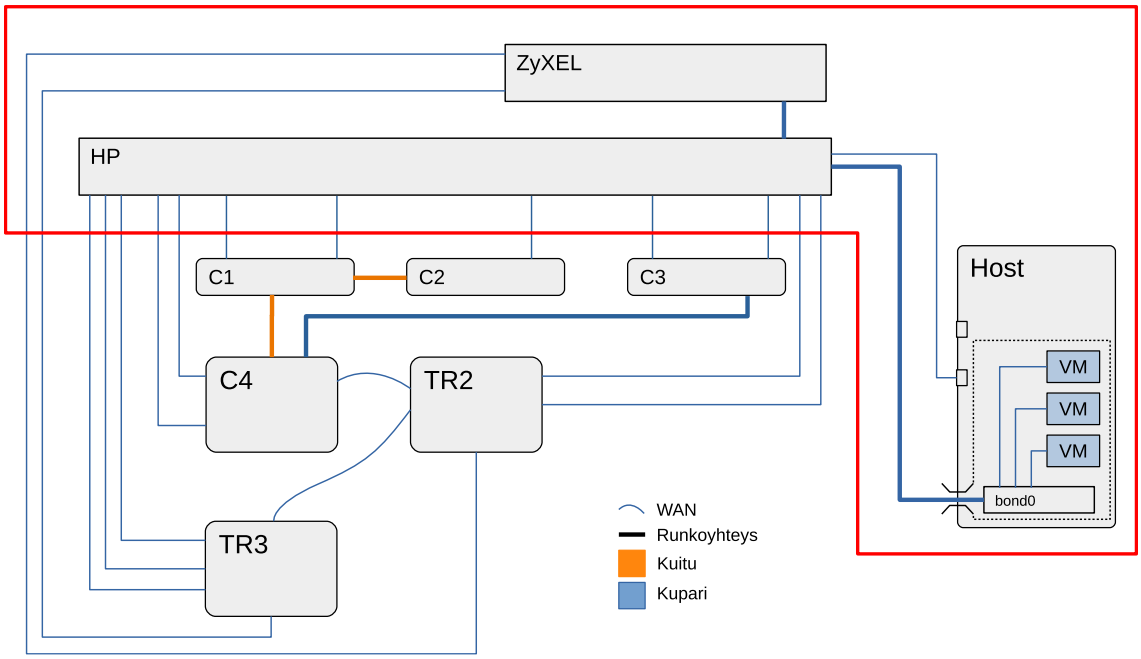
Adapteri liitetään ensin kiinni johonkin isännän USB-porteista. Komennolla `dmesg` nähdään kiinnitetty laite ja sen tuote- sekä valmistajatunnus. Adapterin MAC-osoite on tämän jälkeen nähtävissä suorittamalla komentoriviltä `sudo lsusb –v –d <valmistaja>:<tuote> | grep iMacAddress`.

Käyttäen laitteen MAC-osoitetta luodaan sääntö, jolla sovitin liittyy isäntään aina saman nimisenä verkkorajapintana. Lisäystä varten avataan tiedosto /etc/udev/rules.d/70-persistent-net.rules ja lisätään tiedostoon alla oleva rivi.

```
SUBSYSTEM==”net”,ACTION==”add”,DRIVERS”?*”,ATTR{address}==”<mac-oso-
ite>”,ATTR{dev_id}==”0x0”,ATTR{type}==”1”,KERNEL==”eth*”,NAME=”eth3”
```

Adapterin uudelleenkiinnityksen jälkeen se näkyy nimettynä verkkorajapintana. Kun adapterin lisäys on valmis, isäntä liittyy testipaikan sisäiseen verkkoon 192.168.1.0/24 sen kautta.

Virtualisointiin liittyvät toimenpiteet on nyt saatu suoritettua ja kaikki tietokoneet ovat valmiina käyttöön. Valmis osuus on nähtävillä kuvassa 31. Seuraavaksi on rakennettava itse testattava verkko.



KUVA 31. Punaisella merkittynä valmiiksi saatettu osuus.

7.3 Kolmas vaihe

7.3.1 Cisco Catalyst-kytkinten asetukset

Käynnistymisen jälkeen uusi kytkin kysyy, halutaanko laitteelle asettaa perusasetukset. Vastattaessa kyllä kytkimelle asetetaan nimi, salasanat sekä hallintaan liittyvät asetukset. Hallintaan käytettäväksi verkkorajapinnaksi asetetaan vlan100 ja sen SVI:n osoitteeksi 11.0.1.11/26.

Kytkimen hallitsemiseksi ilman USB-yhteyttä otetaan siinä käyttöön SSH-palvelu. Työssä autentikointiin tullaan käyttämään tavanomaista käyttäjänimi-salasanayhdistelmää.

```
enable
configure terminal
username <knimi> privilege 15 password <salasana>
line vty 0 15
login local
exit
```

Ensimmäisellä komennolla siirrytään tilaan, jossa on oikeudet muuttaa kytkimen asetuksia. Toinen käsky ottaa asetusten muokkaamisen käyttöön. Kolmas käsky luo käyttäjätilin ja asettaa sille salasanan. Privilege-taso määrittää käyttäjän oikeudet, joista taso 15 on suurin. Neljäs käsky siirtyy virtuaaliterminaalien asetuksiin ja samalla määrätään, montako yhtäaikaista terminaaliyhteyttä hallintaan on käytettävissä. Viides käsky sallii valituille terminaaliyhteyksille kirjautumisen käyttäjätunnus-salasanayhdistelmällä. Lopuksi palataan yleiseen hallintatilaan.

Käyttäjätili on nyt luotu, mutta SSH-palvelu ei ole vielä käytössä. Käyttöönottoa varten luodaan 1024-bittinen SSH-avain ja pakotetaan palvelu käyttämään SSH:n versiota 2. Tämän jälkeen kytkimelle on mahdollista kirjautua SSH-yhteyttä käyttäen normaaliin tapaan komennolla `ssh <knimi>@<vlan100 SVI IP>`.

```
crypto key generate rsa
1024
ip ssh version 2
end
```

Ympäristöön rakennettavassa verkossa on tarkoitus käyttää neljää virtuaalilähiverkkoa. Ciscojen portit on jaettava näihin neljään verkkoon ja lisäksi on luotava runkoyhteydet kytkinten välille. Virtuaaliverkkoja ei tarvitse erikseen luoda, vaan porttien liittäminen niihin myös luo verkon, mikäli sitä ei vielä ole. Portti 1 pidetään yhteydessä hallintaverkkoon, portit 2–4 liitetään virtuaalilähiverkkoon 2, portit 5–8 virtuaalilähiverkkoon 3 ja portit 9–12 virtuaalilähiverkkoon 5.

```
configure terminal
interface gigabitethernet 0/1
switchport mode access
switchport access vlan 100
```

```
exit
interface range gigabitethernet 0/2-4
switchport mode access
switchport access vlan 2
exit
interface range gigabitethernet 0/5-8
switchport mode access
switchport access vlan 3
exit
interface range gigabitethernet 0/9-12
switchport mode access
switchport access vlan 5
exit
```

Runkoyhteyteen käytettävät portit vaihtelevat kytkinten välillä. Kytkimellä C1 molemmat kuituportit pakotetaan toimimaan runkoyhteytenä. Porttia 16 tullaan käyttämään reitittimen suuntaan ja port-tia 15 kytkimen C2 suuntaan. Asetukset ovat molemmille porteille identtiset. Lisäksi portit 13 ja 14 poistetaan käytöstä.

```
interface range gigabitethernet 0/15-16
switchport mode trunk
switchport nonegotiate
switchport trunk allowed vlan remove 1
exit
interface range gigabitethernet 0/13-14
shutdown
exit
```

Kirjautuminen SSH-yhteyksien kautta rajoitetaan 11.0.1.0/26-verkkoon.

```
ip access-list standard permit-login
permit 11.0.1.0 0.0.0.63
exit
line vty 0 15
access-class permit-login in
end
copy running-config startup-config
```

Verkossa ei haluta käyttää VTP:tä, joten SSH-palvelun asetusten lisäksi virtuaalilähiverkkojen konfigurointi on tehtävä jokaiselle kytkimelle erikseen. Kaikki yllämainitut toimenpiteet toistetaan Ciscon kytkimille C2 ja C3 tietyin muutoksin. Kytkimellä C2 ainoastaan kuituportti 16 toimii runko-yhteytenä ja portit 13–15 poistetaan käytöstä. Kytkimellä C3 runkoyhteytenä toimii normaali kupa-riyhteyttä käyttävä portti 14. Portit 13, 15 ja 16 poistetaan käytöstä. SVI:n IP-osoitteiden viimeset oktetit ovat 12 ja 13.

Yksi kytkimistä, C4, joutuu hieman erilaiseen rooliin ja sen on työssä korvattava yksi taktisista reitittimistä. Kytkimellä otetaan käyttöön OSPF-reititysprotokolla ja DHCP-palvelu.

```
configure terminal
interface gigabitethernet 0/14
no switchport
ip address 10.0.1.9 255.255.255.252
exit
ip route
router ospf 1
network 10.0.1.8 0.0.0.3 area 0
redistribute connected subnets
exit
```

No switchport-käsky vaihtaa portin tilaa ja muuttaa sen L2-tason kytkinportista reitittäväksi L3-tason portiksi. Seuraava käsky asettaa sille IP-osoitteen ja ip route ottaa reitityspalvelut käyttöön. Käyttöönoton jälkeen luodaan OSPF-reititysprosessi ja määrätään, mihin verkkoon kuuluvia osoitteita sen tulee reitittää sekä mihin alueeseen ne kuuluvat. Reittien mainostaminen ja reititys suoraankytkettyihin aliverkkoihin otetaan käyttöön redistribute komennolla.

Kytkimen on kyettävä reitittämään liikennettä myös virtuaalilähiverkkojen välillä. Reitittämisen mahdollistamiseksi jokaiselle virtuaalilähiverkolle on reitittimessä lisättävä oma IP-osoitteellinen SVI.

```
configure terminal
interface vlan 2
ip address 11.0.1.65 255.255.255.192
exit
interface vlan 3
ip address 11.0.1.129 255.255.255.192
exit
interface vlan 5
ip address 11.0.1.193 255.255.255.192
end
```

Virtuaalilähiverkon 100 SVI:llä on oltava IP-osoite konfigurointia varten, mutta liikennettä kyseisestä verkosta tai verkkoon ei kuitenkaan haluta reitittää. Lisäämällä reitittimen virtuaaliseen porttiin ACL (Access Control List) voidaan liikennettä suodattaa luotujen sääntöjen perusteella. Oikeassa käytössä olevien reitittimien palomuurisäännöt voivat muodostua pitkiksi ja hienostuneiksi, mutta tässä tapauksessa lista on hyvin yksinkertainen.

```
configure terminal
```

```
ip access-list extended ISOLATE
deny ip any any
exit
interface vlan 100
ip access-group ISOLATE out
end
```

Luotuun sääntöön lisätään ainoastaan yksi estosääntö, jolla pysäytetään kaikki liikenne sisään tai ulos verkosta. Sääntö asetetaan toimivaksi reitittimeltä ulospäin verkkoon vlan100 suuntautuvaan liikenteeseen.

Virtuaalilähiverkoille 2, 3 ja 5 otetaan käyttöön DHCP-palvelu. Tietyt osoitealueet poistetaan jaettavien osoitteiden listalta osoiteristiriitojen estämiseksi. Näitä osoitteita ovat esimerkiksi kytkinten omat osoitteet tai verkon muut staattisesti määritellyt laitteet.

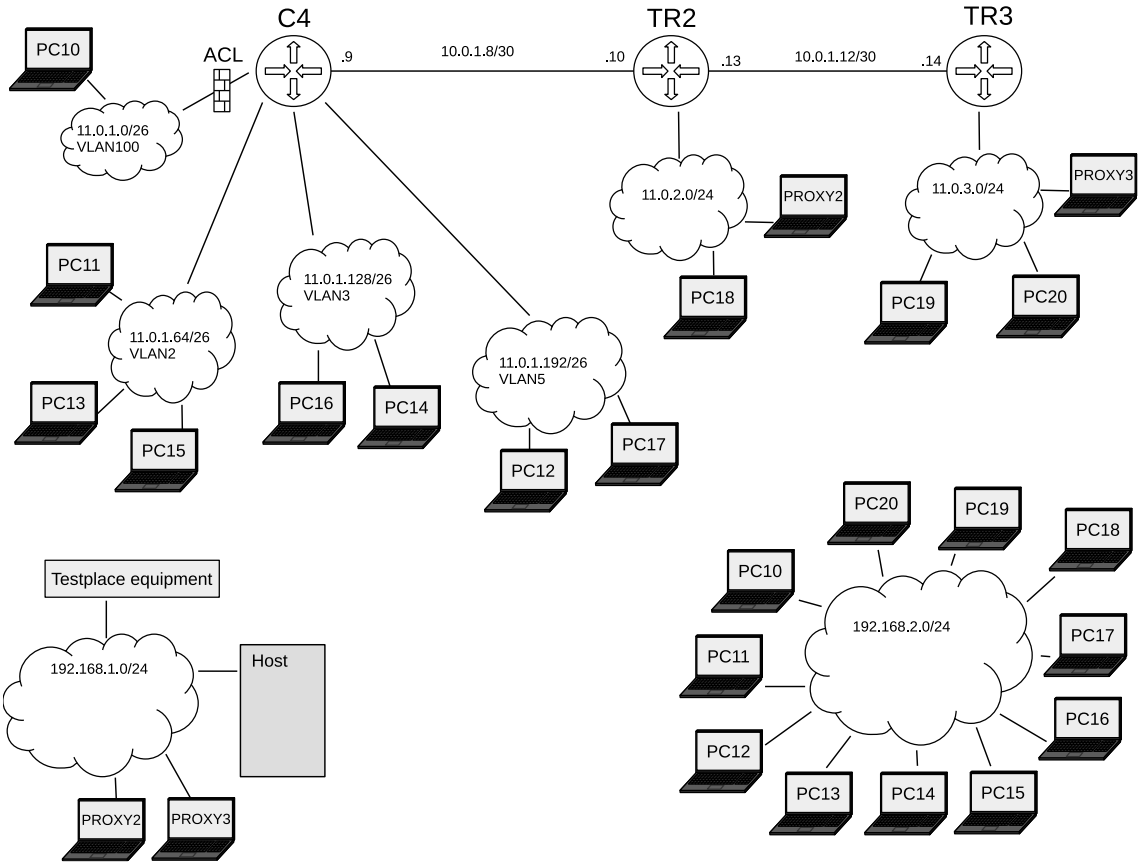
```
conf term
ip dhcp excluded-address 11.0.1.65 11.0.1.70
ip dhcp excluded-address 11.0.1.129 11.0.1.135
ip dhcp excluded-address 11.0.1.193 11.0.1.200
ip dhcp pool vlan2
network 11.0.1.64 255.255.255.192
default router 11.0.1.65
exit
ip dhcp pool vlan3
network 11.0.1.128 255.255.255.192
default router 11.0.1.129
exit
ip dhcp pool vlan5
network 11.0.1.192 255.255.255.192
default router 11.0.1.193
end
```

7.3.2 Taktisen reitittimen asetukset

Yhdellä taktisista reitittimistä on käytössä kaksi eri reititysprotokollaa. Ensimmäinen WAN-portti käyttää OSPF-protokollaa liikenteeseen Ciscon kytkimen kanssa. Toisella WAN-portilla on käytössään OLSR-reititysprotokolla. OLSR-protokollaa käytetään taktisten reitittimien väliseen kommunikointiin. Taktisilla reitittimillä ei tässä työssä käytetä virtuaalilähiverkkoja.

7.4 Testiverkko

Kuvassa 32 on esitetty testattava verkko. Oikean toiminnallisuuden varmistamiseksi seuraavassa luvussa suoritetaan muutama yksinkertainen testi.



KUVA 32. Testipaikalle rakennetut verkot.

Taulukkoon 3 on koottu kuvan 32 osoiteavaruuksien käyttö.

TAULUKKO 3. Verkkojen käyttökohteet.

Osoiteavaruus	Käyttötarkoitus
192.168.1.0/24	Testipaikan sisäisen verkko, sisältää pääasiassa fyysiset laitteet ja proxyt. Ei testauksen kohteena.
192.168.2.0/24	Virtuaaliverkko, sisältää virtualisoidut testikoneet. Verkosta ei yhteyttä fyysisiin rajapintoihin ilman SSH-tunnelin avaamista. Ei testauksen kohteena.
11.0.2.0/24	TR2 lähiverkko
11.0.3.0/24	TR3 lähiverkko
11.0.1.0/26	TR1 VLAN2
11.0.1.64/26	TR1 VLAN3
11.0.1.128/26	TR1 VLAN5
11.0.1.192/26	TR1 VLAN100
10.0.1.8/30	TR1-TR2 OSPF WAN
10.0.1.12/30	TR2-TR3 OLSR WAN

8 TOTEUTETUN JÄRJESTELMÄN TESTAUS

Toimivuuden ja oikeiden reittien varmistamiksesi luotiin yksinkertainen bash-skripti. Liittessä 4 esitetty skripti testasi yhteyden toimivuuden jokaiselta virtuaalikoneelta jokaiseen toiseen virtuaalikoneeseen. Ajon tuloksena oli matriisi, josta oli nähtävissä reittien tilat. Variaatioita suoritettiin seitsemän kappaletta, muuttaen aina järjestelmän tilaa ja olemassa olevia reittejä. Jokaiselle testille oli oma oletettu lopputuloksensa riippuen siitä, mitä kautta liikenteen tulisi kulkea. Mikäli saatu lopputulos poikkesi tästä, oli verkko rakennettu virheellisesti ja vaati korjauksia.

Onnistunut ping-testi jo itsessään varmistaa reitityksen toimimisen molempiin suuntiin, jolloin toimivuuden varmistamiseksi olisi riittänyt puolet toteutetuista yhteyskokeiluista testiä kohti. Lisäksi testistä olisi saatu kehittyneempi, mikäli vastaanottavassa päässä olisi esimerkiksi tarkkailtu saapuvia ping-pyyntöjä ja tulokset merkitty sen pohjalta.

Suorat kuvankaappaukset tuloksista ovat nähtävissä liitteessä 5. On huomattava, että kuvankaappauksissa tuloksissa näkyy aina OK koneen ottaessa yhteyttä itseensä. Taulukossa tämä on korvattu merkinnällä ”-”.

8.1 Ensimmäinen testi

Ensimmäisessä testissä kaikki fyysiset yhteydet olivat kiinnitettynä, reititysominaisuudet olivat käytössä ja ACL ei ollut toiminnassa. Oletus oli, että kaikki virtuaalikoneet voivat kommunikoida esteettä keskenään. Testin tulokset ovat esitettynä taulukossa 4. Pystysarakkeessa on kohdelaite ja vaakarivillä lähettävä virtuaalikone. Testin tulokset vastasivat odotuksia.

TAULUKKO 4. Lähtötilanteessa kaikki yhteydet olivat toiminnassa.

	PC10	PC11	PC12	PC13	PC14	PC15	PC16	PC17	PC18	PC19	PC20
PC10	-	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
PC11	OK	-	OK	OK	OK	OK	OK	OK	OK	OK	OK
PC12	OK	OK	-	OK	OK	OK	OK	OK	OK	OK	OK
PC13	OK	OK	OK	-	OK	OK	OK	OK	OK	OK	OK
PC14	OK	OK	OK	OK	-	OK	OK	OK	OK	OK	OK
PC15	OK	OK	OK	OK	OK	-	OK	OK	OK	OK	OK
PC16	OK	OK	OK	OK	OK	OK	-	OK	OK	OK	OK
PC17	OK	OK	OK	OK	OK	OK	OK	-	OK	OK	OK
PC18	OK	OK	OK	OK	OK	OK	OK	OK	-	OK	OK
PC19	OK	OK	OK	OK	OK	OK	OK	OK	OK	-	OK
PC20	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	-

8.2 Toinen testi

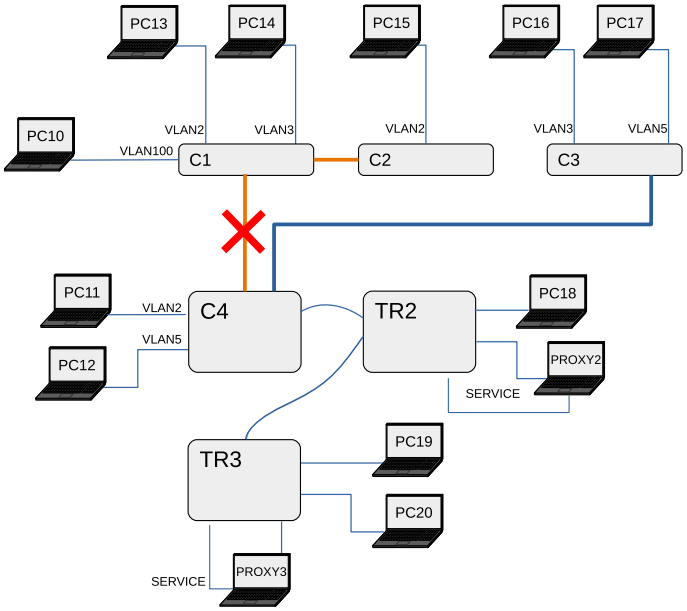
Seuraavaksi reitittävällä Ciscon kytkimellä otettiin käyttöön ACL. Oletus oli, ettei PC10 kykene kommunikoimaan verkon muiden koneiden kanssa eivätkä muut PC10:n kanssa. Tulokset on esitetty taulukossa 5 ja ne vastasivat odotuksia.

TAULUKKO 5. Tulokset, kun ACL oli käytössä.

	PC10	PC11	PC12	PC13	PC14	PC15	PC16	PC17	PC18	PC19	PC20
PC10	-	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK
PC11	NOK	-	OK	OK	OK	OK	OK	OK	OK	OK	OK
PC12	NOK	OK	-	OK	OK	OK	OK	OK	OK	OK	OK
PC13	NOK	OK	OK	-	OK	OK	OK	OK	OK	OK	OK
PC14	NOK	OK	OK	OK	-	OK	OK	OK	OK	OK	OK
PC15	NOK	OK	OK	OK	OK	-	OK	OK	OK	OK	OK
PC16	NOK	OK	OK	OK	OK	OK	-	OK	OK	OK	OK
PC17	NOK	OK	OK	OK	OK	OK	OK	-	OK	OK	OK
PC18	NOK	OK	OK	OK	OK	OK	OK	OK	-	OK	OK
PC19	NOK	OK	OK	OK	OK	OK	OK	OK	OK	-	OK
PC20	NOK	OK	OK	OK	OK	OK	OK	OK	OK	OK	-

8.3 Kolmas testi

Kolmannessa testissä reitittävälle Ciscon kytkimelle menevä kuituyhteys irrotettiin. Koneiden 13 ja 15 ei tulisi kyetä kommunikoimaan muiden kuin toistensa kanssa ja koneen 14 tulisi olla täysin eristettynä, sillä se sijaitsi omassa virtuaalilähiverkossaan. ACL pidettiin käytössä. Kuvaan 33 on merkitty irrotettu kaapeli. Tulokset on esitetty taulukossa 6 ja ne vastasivat odotuksia.



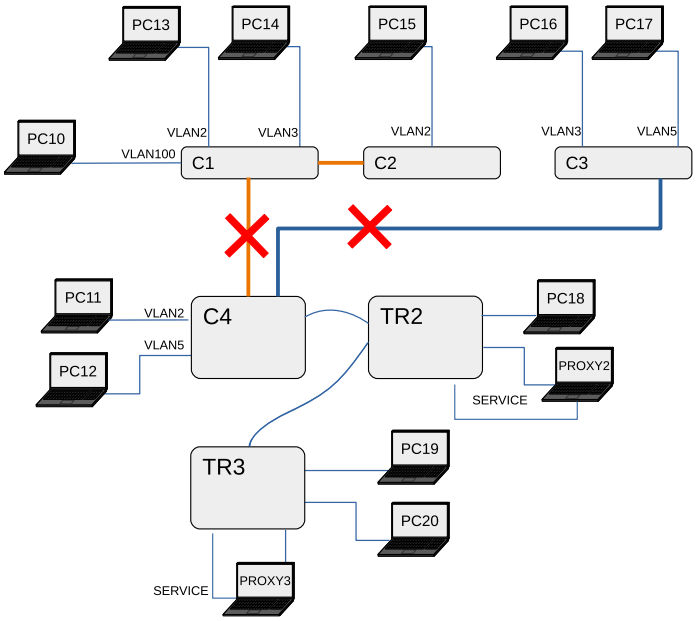
KUVA 33. Testissä 3 katkaistu yhteys.

TAULUKKO 6. Kolmannen testin tulokset.

	PC10	PC11	PC12	PC13	PC14	PC15	PC16	PC17	PC18	PC19	PC20
PC10	-	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK
PC11	NOK	-	OK	NOK	NOK	NOK	OK	OK	OK	OK	OK
PC12	NOK	OK	-	NOK	NOK	NOK	OK	OK	OK	OK	OK
PC13	NOK	NOK	NOK	-	NOK	OK	NOK	NOK	NOK	NOK	NOK
PC14	NOK	NOK	NOK	NOK	-	NOK	NOK	NOK	NOK	NOK	NOK
PC15	NOK	NOK	NOK	OK	NOK	-	NOK	NOK	NOK	NOK	NOK
PC16	NOK	OK	OK	NOK	NOK	NOK	-	OK	OK	OK	OK
PC17	NOK	OK	OK	NOK	NOK	NOK	OK	-	OK	OK	OK
PC18	NOK	OK	OK	NOK	NOK	NOK	OK	OK	-	OK	OK
PC19	NOK	OK	OK	NOK	NOK	NOK	OK	OK	OK	-	OK
PC20	NOK	OK	OK	NOK	NOK	NOK	OK	OK	OK	OK	-

8.4 Neljäs testi

Neljännessä testissä myös C3:n runkoyhteys irrotettiin. Edellisen testin tulosten lisäksi myös koneiden 16 ja 17 tulisi olla täysin eristettynä. Katkaistut yhteydet on merkitty kuvaan 34 ja tulokset esitetty taulukossa 7. Tulokset vastasivat odotuksia.



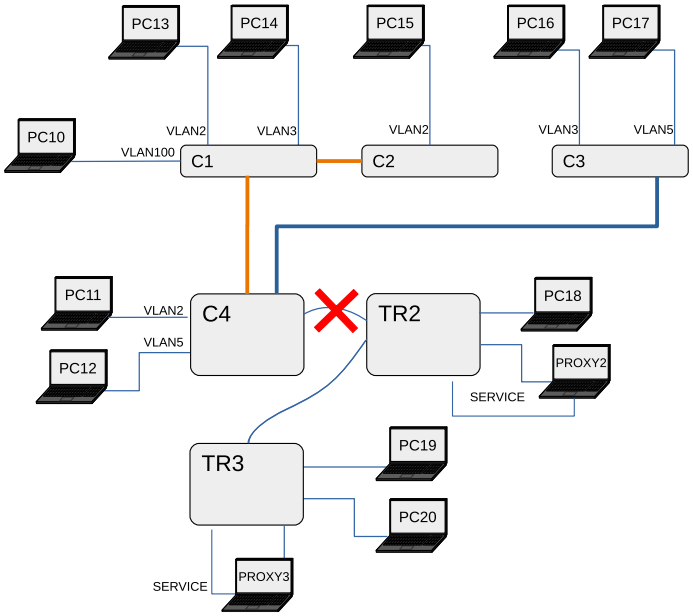
KUVA 34. Irrotetut runkoyhteydet testissä neljä.

TAULUKKO 7. Neljännen testin tulokset.

	PC10	PC11	PC12	PC13	PC14	PC15	PC16	PC17	PC18	PC19	PC20
PC10	-	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK
PC11	NOK	-	OK	NOK	NOK	NOK	NOK	NOK	OK	OK	OK
PC12	NOK	OK	-	NOK	NOK	NOK	NOK	NOK	OK	OK	OK
PC13	NOK	NOK	NOK	-	NOK	OK	NOK	NOK	NOK	NOK	NOK
PC14	NOK	NOK	NOK	NOK	-	NOK	NOK	NOK	NOK	NOK	NOK
PC15	NOK	NOK	NOK	OK	NOK	-	NOK	NOK	NOK	NOK	NOK
PC16	NOK	NOK	NOK	NOK	NOK	NOK	-	NOK	NOK	NOK	NOK
PC17	NOK	NOK	NOK	NOK	NOK	NOK	NOK	-	NOK	NOK	NOK
PC18	NOK	OK	OK	NOK	NOK	NOK	NOK	NOK	-	OK	OK
PC19	NOK	OK	OK	NOK	NOK	NOK	NOK	NOK	OK	-	OK
PC20	NOK	OK	OK	NOK	NOK	NOK	NOK	NOK	OK	OK	-

8.5 Viides testi

Viidettä testiä varten runkoyhteydet kytkettiin takaisin ja WAN-yhteys reitittävältä Ciscon kytkimeltä irrotettiin. Oletus oli etteivät PC:t 18–20 kykene kommunikoimaan muiden kuin toistensa kanssa. ACL pidettiin käytössä, jolloin PC10 oli eristetty. Katkaistu yhteys on merkitty kuvaan 35. Tulokset on esitetty taulukossa 8 ja ne vastasivat odotuksia.



KUVA 35. Viidennen testin yhteydet.

TAULUKKO 8. Viidennen testin tulokset.

	PC10	PC11	PC12	PC13	PC14	PC15	PC16	PC17	PC18	PC19	PC20
PC10	-	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK
PC11	NOK	-	OK	OK	OK	OK	OK	OK	NOK	NOK	NOK
PC12	NOK	OK	-	OK	OK	OK	OK	OK	NOK	NOK	NOK
PC13	NOK	OK	OK	-	OK	OK	OK	OK	NOK	NOK	NOK
PC14	NOK	OK	OK	OK	-	OK	OK	OK	NOK	NOK	NOK
PC15	NOK	OK	OK	OK	OK	-	OK	OK	NOK	NOK	NOK
PC16	NOK	OK	OK	OK	OK	OK	-	OK	NOK	NOK	NOK
PC17	NOK	OK	OK	OK	OK	OK	OK	-	NOK	NOK	NOK
PC18	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	-	OK	OK
PC19	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	OK	-	OK
PC20	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	OK	OK	-

8.6 Kuudes testi

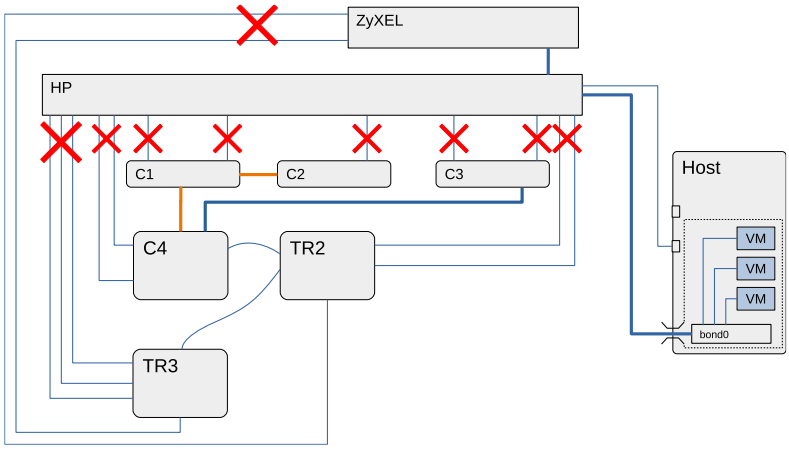
WAN-yhteys kytkettiin takaisin mutta Ciscon reititysominaisuudet kytkettiin pois käytöstä. Oletus oli, etteivät eri virtuaalilähiverkoissa olevat koneet voi kommunikoida keskenään ja etteivät koneet 18–20 kykene kommunikoimaan muiden kuin toistensa kanssa. ACL pidettiin käytössä. Tulokset ovat nähtävissä taulukossa 9 ja ne vastasivat odotuksia.

TAULUKKO 9. Kuudennen testin tulokset.

	PC10	PC11	PC12	PC13	PC14	PC15	PC16	PC17	PC18	PC19	PC20
PC10	-	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK
PC11	NOK	-	NOK	OK	NOK	OK	NOK	NOK	NOK	NOK	NOK
PC12	NOK	NOK	-	NOK	NOK	NOK	NOK	OK	NOK	NOK	NOK
PC13	NOK	OK	NOK	-	NOK	OK	NOK	NOK	NOK	NOK	NOK
PC14	NOK	NOK	NOK	NOK	-	NOK	OK	NOK	NOK	NOK	NOK
PC15	NOK	OK	NOK	OK	NOK	-	NOK	NOK	NOK	NOK	NOK
PC16	NOK	NOK	NOK	NOK	OK	NOK	-	NOK	NOK	NOK	NOK
PC17	NOK	NOK	OK	NOK	NOK	NOK	NOK	-	NOK	NOK	NOK
PC18	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	-	OK	OK
PC19	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	OK	-	OK
PC20	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	OK	OK	-

8.7 Seitsemäs testi

Vielä lopuksi kaikki Ciscoille ja TR:lle liitetyt johdot irrotettiin. Koneiden ei pitäisi pystyä kommunikoimaan toistensa kanssa. Tällä varmistuttiin siitä, ettei liikenne kulkenut jotain muuta reittiä virtuaalikoneesta toiseen. Kuva 36 havainnollistaa asetelmaa ja tulokset on esitetty taulukossa 10. Viimeisenkin testin tulokset vastasivat odotuksia.



KUVA 36. Viimeisen testin asetelma.

TAULUKKO 10. Seitsemännen testin tulokset.

	PC10	PC11	PC12	PC13	PC14	PC15	PC16	PC17	PC18	PC19	PC20
PC10	-	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK
PC11	NOK	-	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK
PC12	NOK	NOK	-	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK
PC13	NOK	NOK	NOK	-	NOK	NOK	NOK	NOK	NOK	NOK	NOK
PC14	NOK	NOK	NOK	NOK	-	NOK	NOK	NOK	NOK	NOK	NOK
PC15	NOK	NOK	NOK	NOK	NOK	-	NOK	NOK	NOK	NOK	NOK
PC16	NOK	NOK	NOK	NOK	NOK	NOK	-	NOK	NOK	NOK	NOK
PC17	NOK	NOK	NOK	NOK	NOK	NOK	NOK	-	NOK	NOK	NOK
PC18	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	-	NOK	NOK
PC19	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	-	NOK
PC20	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	-

Kaikki suoritetut testit tuottivat odotetut tulokset ja verkko toimii suunnitellusti. Testiympäristö voidaan ottaa käyttöön.

9 YHTEENVETO

Työn tarkoituksena oli suunnitella ja rakentaa ympäristö, jota voitaisiin käyttää yrityksen tuotteiden ja ohjelmistojen testaamiseen. Työ onnistui kokonaisuutena hyvin eikä sen aikana kohdattu suurempia haasteita. Valmiit pohjatiedot työhön liittyvistä teknologioista helpottivat ongelmien ratkaisua.

Tehtävä tarjosi mielenkiintoisen aihepiirin ja tiedot sen usealta osa-alueelta syventyivät, enimmäkseen virtuaaliverkoista ja virtuaalilähiverkoista. Oman mielenkiintonsa työhön toi myös laajan kokonaisuuden rakentaminen alusta loppuun.

Virtuaalilähiverkot olivat käsitteenä tuttu, mutta niitä päästiin hyödyntämään ensimmäisen kerran työn aikana. Fyysisten laitteiden kanssa kommunikaation mahdollistavia virtuaaliverkkoja ei myöskään ollut aiemmin käytetty. Prosessorin ja verkkoliikenteen virtualisoinnin teoria vei suurimman osan opiskeluun käytetystä ajasta, eikä tietoa varsinaisesti edes tarvittu virtualisoinnin käyttämiseksi. Taustojen ymmärtäminen koettiin kuitenkin hyödylliseksi jatkon kannalta sekä kokonaisuuden hahmottamiseksi.

Virtualisointi vähensi testipaikalta vaadittavia fyysisiä resursseja huomattavasti, noin kymmenesosaan, ja toimeksiantaja on ollut tulokseen tyytyväinen. Erilliset fyysiset laitteet olisivat vieneet huomattavasti suuremman tilan laboratoriosta sekä maksaneet enemmän. Ympäristön helppo laajennettavuus ja mukauttaminen ilman lisäinvestointeja ovat myös selkeä etu tulevaisuuden kannalta. Yrityksen harkintaan jätetäänkin muun muassa olemassa olevien testipaikkojen virtualisointi mahdollisilta osin sekä virtualisoinnin käyttö jatkossa ennen uusien laitteiden tilaamista. Mikäli teknologiaa halutaan soveltaa laajemmin, tarkoitusta varten on ehdotettu tarkoitukseen sopivan laitteiston hankintaa.

Testipaikka on ollut käytössä jonkin aikaa ja ongelmia ei pienten korjausten lisäksi ole esiintynyt. Systeemitestauksessa käytettävän automaation kyky toimia virtualisoidussa ympäristössä on varmistettu onnistuneesti ja tulevaisuudessa ympäristö tullaan liittämään osaksi yrityksen päivittäistä testiautomaatiota.

LÄHTEET

Adams, K. & Agesen, O. 2006. A Comparison of Software and Hardware Techniques for x86 Virtualization. Julkaisu konferenssissa ASPLOS'06, October 21–25, San Jose, California, USA. Viitattu 19.6.2019, https://www.vmware.com/pdf/asplos235_adams.pdf.

Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I. & Warfield, A. 2013. Xen and the Art of Virtualization. Julkaisu konferenssissa SOSP'03, October 19–22, BoltonLanding, New York, USA. Viitattu 19.6.2019, https://www.cl.cam.ac.uk/research/srg/netos/papers/2003-xensosp.pdf.

CCNA Blog 2019a. Dynamic routing protocols. Viitattu 19.6.2019, https://www.ccnablog.com/dynamic-routing-protocols/.

CCNA Blog 2019b. OSPF Part I. Viitattu 19.6.2019, https://www.ccnablog.com/ospf-part-1/.

CCNA Blog 2019c. OSPF Part II. Viitattu 19.6.2019, https://www.ccnablog.com/ospf-part-2/.

Cisco Systems 2017. ConsolidatedPlatformConfigurationGuide,CiscoIOS Release15.2(6)E (Catalyst3560-CXand 2960-CXSwitches). Viitattu 21.6.2019, https://www.cisco.com/c/en/us/td/docs/switches/lan/catalyst2960cx_3650cx/software/release/15-2_6_e/configuration_guide/b_1526e_consolidated_3560cx_2960cx_cg.pdf.

Clausen, T. & Jacquet, P. 2003. Optimized Link State Routing Protocol (OLSR). Project Hipercom, INRIA. Internet Engineering Task Force. Viitattu 19.6.2019, https://tools.ietf.org/html/rfc3626.

Davis, T., Tarreau, W., Gavrilov, C., Tindel, C. N., Girouard, J., Vosburgh, J. & Williams, M. 2011. Linux Ethernet Bonding Driver HOWTO. The Linux Kernel Organization, Inc. Viitattu 21.06.2019, https://www.kernel.org/doc/Documentation/networking/bonding.txt.

Fisher-Ogden, J. 2019, Hardware Support for Efficient Virtualization. University of California, San Diego. Viitattu 19.6.2019, https://cseweb.ucsd.edu/~jfisherogden/hardwareVirt.pdf.

Hanrahan, J. 2016. Vastaus keskustelussa Why do x86 CPUs only use 2 out of 4 rings? Super User. Viitattu 19.6.2019, https://superuser.com/questions/1063420/why-do-x86-cpus-only-use-2-out-of-4-rings.

Hewlett Packard Enterprise 2016. HPE OfficeConnect 1850 24G/48G Switch Series Management and Configuration Guide. Viitattu 6.9.2019, https://support.hpe.com/hpsc/doc/public/display?docId=emr_na-c05324972.

IBM 2019. MacVTap driver considerations. Viitattu 20.6.2019, https://www.ibm.com/support/knowledgecenter/en/linuxonibm/liaag/wkvm/wkvm_c_net_conmac.htm.

IEEE 2010. IEEE 802.3ad-2000 - IEEE Standard for Information Technology - Local and Metropolitan Area Networks - Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications-Aggregation of Multiple Link Segments. Viitattu 6.9.2019, https://standards.ieee.org/standard/802_3ad-2000.html (maksullinen).

IEEE 2018. IEEE 802.1Q-2018 - IEEE Standard for Local and Metropolitan Area Networks—Bridges and Bridged Networks. Viitattu 6.9.2019, https://standards.ieee.org/standard/802_1Q-2018.html.

Intel 2019. Intel® 64 and IA-32 ArchitecturesSoftware Developer's ManualVolume 3C: System Programming Guide, Part 3. Viitattu 19.6.2019, https://software.intel.com/sites/default/files/managed/7c/f1/326019-sdm-vol-3c.pdf.

Toomey, W. 2012. Introduction to Operating Systems. Luentomateriaali. Viitattu 19.6.2019, https://minnie.tuhs.org/CompArch/Lectures/week07.html.

Jones, M. 2010. Virtio: An I/O virtualization framework for Linux. IBM. Viitattu 20.6.2019, https://developer.ibm.com/articles/l-virtio/.

Juniper Networks 2017. Understanding OSPF Areas and Backbone Areas. Viitattu 19.6.2019, https://www.juniper.net/documentation/en_US/junos/topics/concept/ospf-areas-overview.html.

Kernel Virtual Machine 2019. Viitattu 19.6.2019, https://www.linux-kvm.org/page/Main_Page.

Libvirt FAQ 2017. Viitattu 20.6.2019, https://wiki.libvirt.org/page/FAQ#What_is_libvirt.3F.

MacVTap 2017. Linux Virtualization. Viitattu 20.6.2019, https://virt.kernelnewbies.org/MacVTap.

Main Page 2017. QEMU. Viitattu 20.6.2019, https://wiki.qemu.org/Main_Page.

Popek, G.J., Goldberg, R., P. 1974. Formal Requirements for Virtualizable Third Generation Architectures. Communications of the ACM 17 (7), 412–421. Viitattu 6.9.2019, https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.141.4815&rep=rep1&type=pdf.

Salzman, P. J., Burian, M. & Pomerantz, O. 2007. The Linux Kernel Module Programming Guide. The Linux Documentation Project. Viitattu 20.6.2019, https://www.tldp.org/LDP/lkmpg/2.6/lkmpg.pdf.

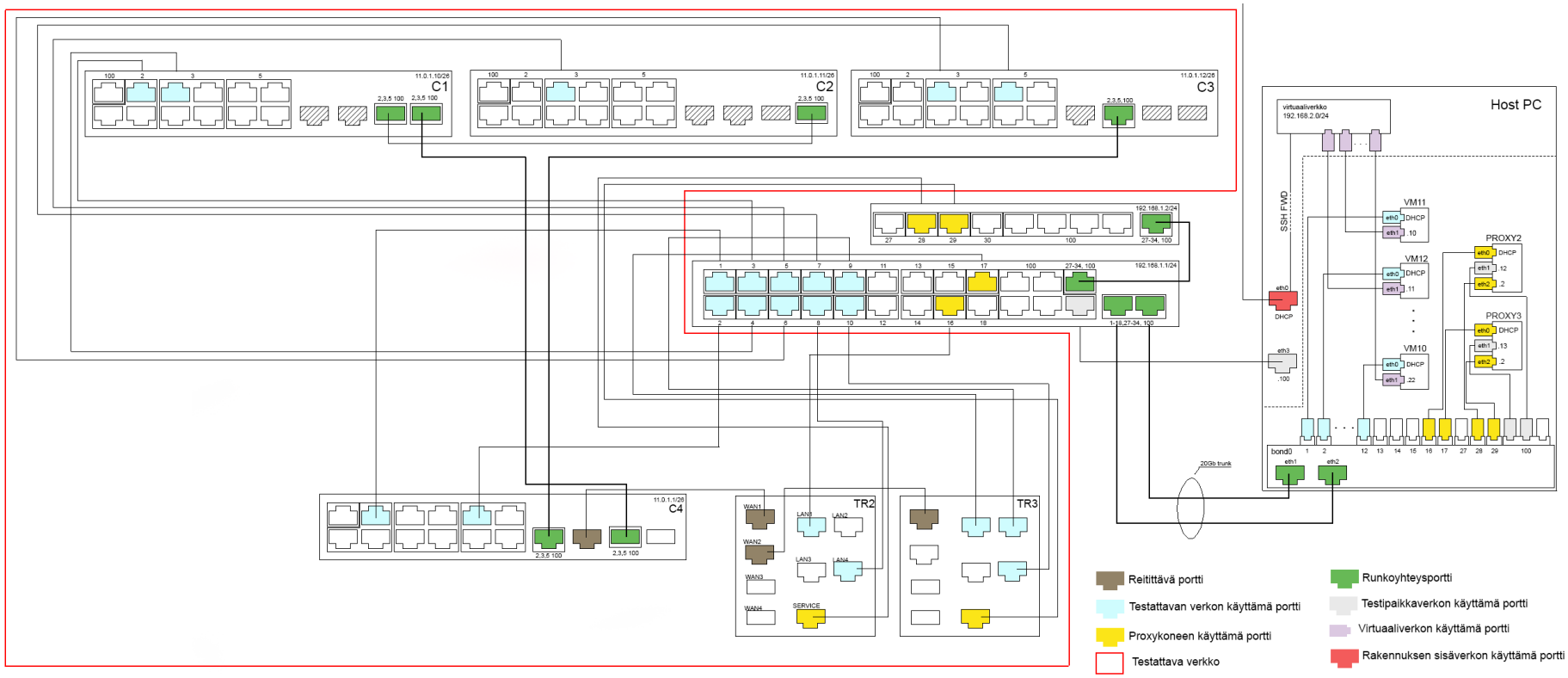
Vhost Sample Application 2019. Data Plane Development Kit. Viitattu 20.6.2019, https://dpdk.readthedocs.io/en/v16.04/sample_app_ug/vhost.html.

Wagner, M. 2011. KVM performance improvements and optimizations. Julkaisu konferenssissa KVM Forum 2011, August 15–16, Vancouver, Canada. Viitattu 20.6.2019, https://www.linux-kvm.org/images/5/59/Kvm-forum-2011-performance-improvements-optimizations-D.pdf.

Zinin, A., Lindem, A. & Yeung, D. 2003. Alternative Implementations of OSPF Area Border Routers. Internet Engineering Task Force. The Internet Society. Viitattu 19.6.2019, https://tools.ietf.org/html/rfc3509.

RAKENNETTU YMPÄRISTÖ

LIITE 1



clone_and_prep_VM.sh

```
#!/bin/sh
template=$1
new=$2
echo "Cloning new guest from ${template}"
virt-clone   --original   ${template}   --name   ${new}   --file
/home/testuser/KVM/guests/${new}.img
echo "Cloning nvram..."
sudo     scp     /var/lib/libvirt/qemu/nvram/${template}_VARS.fd
/var/lib/libvirt/qemu/nvram/${new}_VARS.fd
virt-xml             ${new}             --edit             --boot
nvram=/var/lib/libvirt/qemu/nvram/${new}_VARS.fd
echo "Connecting new guest to network..."
virt-xml ${new} --edit --network network='ctrl'
echo "Preparing cloned guest..."
sudo virt-sysprep --domain ${new} --hostname ${new} --firstboot
./ifacesetup.sh
echo "Completed"
```

ifacesetup.sh

```
#!/bin/bash
#Setup eth0
octet=$(hostname | grep -Po "([0-9]+)$")
echo "
source /etc/network/interfaces.d/*
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
      address 192.168.1."${octet}"
      netmask 255.255.255.0
      gateway 192.168.1.1 " > /etc/network/interfaces
systemctl restart networking
#Setup ssh
ssh-keygen -f id_rsa -t rsa -N ''
dpkg-reconfigure openssh-server
#Change tmpfs size, persistent
echo "tmpfs /run tmpfs size=64M 0 0" >> /etc/fstab
#Set proxy for apt
echo "Acquire
{
http { Proxy "\""http://xx.xx.xx.xx:xx/"\""; }
}" > /etc/apt/apt.conf
sed    -i    '/#    Authentication:/a    PermitRootLogin    yes'
/etc/ssh/sshd_config
```

ping_all.sh

```
#!/bin/bash
ALL_CTRL_IP=(192.168.2.10 192.168.2.11 192.168.2.12 192.168.2.13
192.168.2.14 192.168.2.15 192.168.2.16 192.168.2.17 192.168.2.18
192.168.2.19 192.168.2.20)
ALL_NETWORK_IP=(11.0.1.6    11.0.1.74    11.0.1.198    11.0.1.72
11.0.1.136  11.0.1.73  11.0.1.131  11.0.1.196  11.0.2.4  11.0.3.4
11.0.3.5)
PING_COUNT=2
i=0
printf '%-16s' "source/target" #padding
for each in "${ALL_NETWORK_IP[@]}"; do
     printf "%-12s" "${each}"
done
printf "\n"
for source in ${ALL_CTRL_IP[@]}; do
     printf '%-16s' "${ALL_NETWORK_IP[$i]}"
     for target in ${ALL_NETWORK_IP[@]}; do
          ssh root@$source "ping -W 2 -c ${PING_COUNT} ${tar-
get} >/dev/null && printf '%-12s' 'OK'|| printf '%-12s' 'NOK'"
     done
     ((i++))
     printf "\n"
done
```

1. ACL on poissa käytöstä ja kaikki yhteydet ovat aktiivisena.

source/target	11.0.1.6	11.0.1.74	11.0.1.198	11.0.1.72	11.0.1.136	11.0.1.73	11.0.1.131	11.0.1.196	11.0.2.4	11.0.3.4	11.0.3.5
11.0.1.6	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
11.0.1.74	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
11.0.1.198	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
11.0.1.72	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
11.0.1.136	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
11.0.1.73	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
11.0.1.131	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
11.0.1.196	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
11.0.2.4	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
11.0.3.4	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
11.0.3.5	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK

2. ACL on käytössä ja kaikki yhteydet ovat aktiivisena.

source/target	11.0.1.6	11.0.1.74	11.0.1.198	11.0.1.72	11.0.1.136	11.0.1.73	11.0.1.131	11.0.1.196	11.0.2.4	11.0.3.4	11.0.3.5
11.0.1.6	OK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK
11.0.1.74	NOK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
11.0.1.198	NOK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
11.0.1.72	NOK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
11.0.1.136	NOK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
11.0.1.73	NOK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
11.0.1.131	NOK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
11.0.1.196	NOK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
11.0.2.4	NOK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
11.0.3.4	NOK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
11.0.3.5	NOK	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK

3. Kuituyhteys reitittimelle on irti ja ACL on käytössä.

source/target	11.0.1.6	11.0.1.74	11.0.1.198	11.0.1.72	11.0.1.136	11.0.1.73	11.0.1.131	11.0.1.196	11.0.2.4	11.0.3.4	11.0.3.5
11.0.1.6	OK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK
11.0.1.74	NOK	OK	OK	NOK	NOK	NOK	OK	OK	OK	OK	OK
11.0.1.198	NOK	OK	OK	NOK	NOK	NOK	OK	OK	OK	OK	OK
11.0.1.72	NOK	NOK	NOK	OK	NOK	OK	NOK	NOK	NOK	NOK	NOK
11.0.1.136	NOK	NOK	NOK	NOK	OK	NOK	NOK	NOK	NOK	NOK	NOK
11.0.1.73	NOK	NOK	NOK	OK	NOK	OK	NOK	NOK	NOK	NOK	NOK
11.0.1.131	NOK	OK	OK	NOK	NOK	NOK	OK	OK	OK	OK	OK
11.0.1.196	NOK	OK	OK	NOK	NOK	NOK	OK	OK	OK	OK	OK
11.0.2.4	NOK	OK	OK	NOK	NOK	NOK	OK	OK	OK	OK	OK
11.0.3.4	NOK	OK	OK	NOK	NOK	NOK	OK	OK	OK	OK	OK
11.0.3.5	NOK	OK	OK	NOK	NOK	NOK	OK	OK	OK	OK	OK

4. Molemmat runkoyhteydet reitittimelle ovat irti ja ACL on käytössä.

source/target	11.0.1.6	11.0.1.74	11.0.1.198	11.0.1.72	11.0.1.136	11.0.1.73	11.0.1.131	11.0.1.196	11.0.2.4	11.0.3.4	11.0.3.5
11.0.1.6	OK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK
11.0.1.74	NOK	OK	OK	NOK	NOK	NOK	NOK	NOK	OK	OK	OK
11.0.1.198	NOK	OK	OK	NOK	NOK	NOK	NOK	NOK	OK	OK	OK
11.0.1.72	NOK	NOK	NOK	OK	NOK	OK	NOK	NOK	NOK	NOK	NOK
11.0.1.136	NOK	NOK	NOK	NOK	OK	NOK	NOK	NOK	NOK	NOK	NOK
11.0.1.73	NOK	NOK	NOK	OK	NOK	OK	NOK	NOK	NOK	NOK	NOK
11.0.1.131	NOK	NOK	NOK	NOK	NOK	NOK	OK	NOK	NOK	NOK	NOK
11.0.1.196	NOK	NOK	NOK	NOK	NOK	NOK	NOK	OK	NOK	NOK	NOK
11.0.2.4	NOK	OK	OK	NOK	NOK	NOK	NOK	NOK	OK	OK	OK
11.0.3.4	NOK	OK	OK	NOK	NOK	NOK	NOK	NOK	OK	OK	OK
11.0.3.5	NOK	OK	OK	NOK	NOK	NOK	NOK	NOK	OK	OK	OK

5. Reititinportin yhteys on irti ja ACL on käytössä.

source/target	11.0.1.6	11.0.1.74	11.0.1.198	11.0.1.72	11.0.1.136	11.0.1.73	11.0.1.131	11.0.1.196	11.0.2.4	11.0.3.4	11.0.3.5
11.0.1.6	OK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK
11.0.1.74	NOK	OK	OK	OK	OK	OK	OK	OK	NOK	NOK	NOK
11.0.1.198	NOK	OK	OK	OK	OK	OK	OK	OK	NOK	NOK	NOK
11.0.1.72	NOK	OK	OK	OK	OK	OK	OK	OK	NOK	NOK	NOK
11.0.1.136	NOK	OK	OK	OK	OK	OK	OK	OK	NOK	NOK	NOK
11.0.1.73	NOK	OK	OK	OK	OK	OK	OK	OK	NOK	NOK	NOK
11.0.1.131	NOK	OK	OK	OK	OK	OK	OK	OK	NOK	NOK	NOK
11.0.1.196	NOK	OK	OK	OK	OK	OK	OK	OK	NOK	NOK	NOK
11.0.2.4	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	OK	OK	OK
11.0.3.4	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	OK	OK	OK
11.0.3.5	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	OK	OK	OK

6. Reititysominaisuudet ovat poissa käytöstä, ACL on käytössä.

source/target	11.0.1.6	11.0.1.74	11.0.1.198	11.0.1.72	11.0.1.136	11.0.1.73	11.0.1.131	11.0.1.196	11.0.2.4	11.0.3.4	11.0.3.5
11.0.1.6	OK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK
11.0.1.74	NOK	OK	NOK	OK	NOK	OK	NOK	NOK	NOK	NOK	NOK
11.0.1.198	NOK	NOK	OK	NOK	NOK	NOK	NOK	OK	NOK	NOK	NOK
11.0.1.72	NOK	OK	NOK	OK	NOK	OK	NOK	NOK	NOK	NOK	NOK
11.0.1.136	NOK	NOK	NOK	NOK	OK	NOK	OK	NOK	NOK	NOK	NOK
11.0.1.73	NOK	OK	NOK	OK	NOK	OK	NOK	NOK	NOK	NOK	NOK
11.0.1.131	NOK	NOK	NOK	NOK	OK	NOK	OK	NOK	NOK	NOK	NOK
11.0.1.196	NOK	NOK	OK	NOK	NOK	NOK	NOK	OK	NOK	NOK	NOK
11.0.2.4	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	OK	OK	OK
11.0.3.4	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	OK	OK	OK
11.0.3.5	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	OK	OK	OK

7. Kaikki Ciscon kytkimiin ja TR:iin saapuvat yhteydet on irrotettu.

source/target	11.0.1.6	11.0.1.74	11.0.1.198	11.0.1.72	11.0.1.136	11.0.1.73	11.0.1.131	11.0.1.196	11.0.2.4	11.0.3.4	11.0.3.5
11.0.1.6	OK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK
11.0.1.74	NOK	OK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK
11.0.1.198	NOK	NOK	OK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK
11.0.1.72	NOK	NOK	NOK	OK	NOK	NOK	NOK	NOK	NOK	NOK	NOK
11.0.1.136	NOK	NOK	NOK	NOK	OK	NOK	NOK	NOK	NOK	NOK	NOK
11.0.1.73	NOK	NOK	NOK	NOK	NOK	OK	NOK	NOK	NOK	NOK	NOK
11.0.1.131	NOK	NOK	NOK	NOK	NOK	NOK	OK	NOK	NOK	NOK	NOK
11.0.1.196	NOK	NOK	NOK	NOK	NOK	NOK	NOK	OK	NOK	NOK	NOK
11.0.2.4	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	OK	NOK	NOK
11.0.3.4	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	OK	NOK
11.0.3.5	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	NOK	OK