jamk.fi

Avoimen lähdekoodin SDN-kontrollerien evaluointi

Hiski Karhinen

Opinnäytetyö
Huhtikuu 2015

Tietotekniikan koulutusohjelma
Tekniikan ja liikenteen ala

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

Kuvailulehti

Tekijä(t)	Julkaisun laji	Päivämäärä
Hiski Karhinen	Opinnäytetyö	15.04.2015
	Sivumäärä 93 + 14	Julkaisun kieli Suomi
		Verkkojulkaisulupa myönnetty: x

Työn nimi

Avoimen lähdekoodin SDN-kontrollerien evaluointi

Koulutusohjelma
Tietotekniikan koulutusohjelma

Työn ohjaaja(t)
Karo Saharinen
Mika Rantonen

Toimeksiantaja(t)
Marko Vatanen
Jyväskylän ammattikorkeakoulu / JYVSECTEC

Tiivistelmä

Opinnäytetyö toteutettiin Jyväskylän ammattikorkeakoulun IT-instituuttiin kuuluvalle JYVSECTEC kyberturvallisuuden tutkimushankkeelle. Työn tavoitteena oli arvioida SDN-kontrollereita, niiden soveltuvuutta ja käyttöä JYVSECTEC RGCE-ympäristössä. Teoriaosuudessa tutkitaan Software-defined networking –tekniikkaa sekä OpenFlow-protokollaa.

Työtä varten valikoitiin kaksi avointa SDN-kontrolleria, joiden ominaisuuksia ja käyttöä JYVSECTEC RGCE-ympäristöä varten evaluoitiin. Evaluointia varten valikoitui OpenDaylight ja OpenContrail.

Käytännön osuudessa toteutettiin ympäristöt OpenContrail ja OpenDaylight järjestelmistä. Näiden kahden toteutuksen avulla muodostettiin dokumentaatio järjestelmien perustoiminnoista ja vertailtiin eroavaisuuksia. Dokumentaation pohjalta voitiin luoda arvio järjestelmien käyttökohteista ja käyttötarkoituksista.

Käytännön osuudesta havaittiin, että OpenContrail on pilvipalvelujärjestelmä, joka sopii konesalitoimintaan. OpenDaylight soveltuu fyysisen verkon hallintaan ja toimii pohjana niille, jotka haluavat luoda oman SDN-kontrollerin. Lisäksi huomattiin, että SDN on tekniikkana nopeasti kehittyvä ja siksi tähän työhön liittyvät kontrolleritkin kehittyvät nopeaa vauhtia.

Työ onnistui osittain vaatimusten mukaisesti. Lopputuloksena tilaajalle luotiin testiympäristöt sekä kattava katsaus SDN-tekniikan perusteista ja sen soveltuvuudesta JAMK/JYVSECTEC toiminnassa, tutkimisessa ja opetuksessa.

Avainsanat (asiasanat)
Software-defined networking, SDN, OpenContrail, OpenDaylight

Muut tiedot

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

Description

Author(s)	Type of publication	Date
Hiski Karhinen	Bachelor's thesis	15.04.2015
		Language of publication: Finnish
	Number of pages 93 + 14	Permission for web publication: x

Title of publication

Evaluation of opensource SDN controllers

Degree programme
Information technology

Tutor(s)
Karo Saharinen
Mika Rantonen

Assigned by
Marko Vatanen
Jyväskylä University of Applied Sciences / JYVSECTEC

Abstract

This bachelor's thesis was assigned by cyber security project JYVSECTEC at Jyväskylä University of Applied Science. The objective of this thesis was to evaluate SDN controllers, and discuss the suitability and usage of selected SDN controllers in JYVSECTEC RGCE environment. This thesis covers the basic theory of software-defined networking and OpenFlow protocol.

For this thesis two open SDN controllers were selected. OpenDaylight and OpenContarail Features and usage of these two SDN-controllers in JYVSECTEC RGCE environment were evaluated.

The practical part of this thesis consists of making the test environments for OpenContrail and OpenDaylight. A documentation of basic functions was made for these test environments, and the differences of OpenContrail and OpenDaylight could thus be compared. An assessment for use cases and use purposes was created based on the documentation on OpenContrail and OpenDaylight

During the practical part, OpenContrail was found to be a cloud service system that is suitable for datacenters, whereas OpenDaylight was discovered to be suitable for controlling the physical network and to serve as platform for those who want to create their own SDN controller. Additionally it was noted that SDN is developing fast and therefore also SDN-controllers used in this thesis are developed and updated at fast pace.

The thesis was completed partly according to the requirements. Outcome is that test environments were created and comprehensive documentation of SDN technology was delivered for JAMK/JYVSECTEC to be used in research and training purposes.

Keywords/tags (subjects)
Software-defined networking, SDN, OpenContrail, OpenDaylight

Miscellaneous

Sisällysluettelo

Kuviot

Taulukot

Lyhenteet ja termit

AAA Authentication, Authorization ja Accounting. AAA-protokollaa voidaan käyttää toisen osapuolen tunnistautumiseen verkossa.

ACL Access Control List eli pääsylista on lista määrityksiä, joilla voidaan estää tai sallia tietoliikenteen kulku.

A-CPI SDN-kontrollerin ja SDN-ohjelman välinen rajapinta.

Agentti Ohjelma tai ohjelman osa, joka lähettää ja vastaanottaa tietoa ulkopuolisilta ohjelmilta.

Aliverkko Looginen verkon osa.

API Application Programmatic Interface on ohjelmointirajapinta, jonka kautta ohjelmat voivat keskustella keskenään.

ARP Address Resolution Protocol on protokolla, jota käytetään selvittämään MAC-osoitetta vastaava IP-osoite.

ASIC Application Specific Integrated Circuit. ASIC on mikropiiri, joka on suunniteltu yhden tehtävän tarpeiden mukaiseksi.

BCAM Binary Content-addressable Memory on muistityyppi, johon voidaan kohdentaa hakuja, jotka sisältää ainoastaan ykkösiä ja nollia.

BGP Border Gateway Protocol on reititysprotokolla, jota käytetään vaihtamaan reititystietoa.

BGP-LS BGP-Link State. BGP-protokollaa käytetään välittämään tietoa linkkitiloista.

BSS Business Support System on yrityksen osa, joka toimii yrityksen ja asiakkaan välissä. Vastaa esimerkiksi tilauksista.

DCI Data Center Interconnect. Tämän tyypin protokollia käytetään konesalien verkoissa.

D-CPI SDN-kontrollerin ja verkkoelementin välinen rajapinta.

DDoS Distributed Denial of Service eli hajautettu palvelunestohyökkäys, jossa hyökkäys tapahtuu useammasta kuin yhdestä kohteesta.

DNS Domain Name System. Hierarkkinen nimeämisjärjestelmä, joka muuttaa verkkotunnukset IP-osoitteiksi.

DoS Denial of Service eli palvelunestohyökkäys, jossa kulutetaan kohteen resurssit, jolloin palvelu estyy.

DSCP Differentiated Services Code Point on tapa merkitä IP-paketin prioriteetti.

FIB Forwarding Information Base. Reitittimen reititystaulu. Sisältää parhaat reitit muihin verkkoihin.

HTTP Hypertext Transfer Protocol on protokolla, jota käytetään selainten ja palvelinten väliseen tiedonsiirtoon.

Hyökkäysvektori Hyökkäyksen kohde, jota voidaan hyödyntää tietoverkkoon pääsemiseksi.

IDS Intrusion Detection System on järjestelmä, jolla voidaan havaita haitta-liikenne tietoverkossa.

IP Internet Protocol. Käytetään tietoliikennepakettien kuljettamiseen.

IPS Intrusion Prevention System on järjestelmä, jolla voidaan estää haitta-liikenne tietoverkossa.

IPsec Protokolla, jolla voidaan suojata IP-protokollan mukainen liikenne käyttämällä autentikaatiota ja liikenteen salausta.

JAMK Jyväskylän ammattikorkeakoulu.

JYVSECTEC Jyväskylä Sercurity Technology –tietoturvahanke.

Kontrolleri SDN-verkossa verkkoelementtejä ohjaava laite.

Kytkin L2-tason paketin välitykseen suorittava laite.

LAN Local Area Network eli lähiverkko.

Levykuva Tiedosto, joka sisältää massamuistin sisällön ja rakenteen.

LFIB Label Forwarding Information Base. Sisältää leimatiedon ja porttitiedon yhteydet. Käytetään leimakytkentäisessä paketinvälityksessä.

LIB Label Information Base. Ohjelmistotaulu, johon reititin voi tallentaa portti- ja leimatietoa leimakytkentäistä välitystä varten.

LISP Locator Identifier Separation Protocol reititysprotokolla, joka käyttää tunnisteita ja paikallistajia.

LLDP Link Layer Discovery Protocol –protokolla, jonka tomii L2-tasolla. Tämän protokollan avulla verkkoelementit voivat mainostaa omaa tunnustaan ja ominaisuuksiaan

MAC Media Access Control. Verkkosovittimen yksilöintiin käytettävä osoitteistus.

MPLS Multi-Protocol Label Switching. Leimakytkentäinen reititys. Tapa välittää IP-paketteja käyttämällä leimoja välityspäätöksiin.

NaaS Network as a Service on palvelumalli, jossa voidaan asiakkaalle tarjota tietoverkko palveluna.

NE Network Element eli verkkoelementti. Esimerkiksi kytkin tai reititin.

NFV Network Functions Virtualization on verkkoresurssien ja verkkoominaisuuksien virtualisointiin tarkoitettu tekniikka.

NPU Network processor. Integroitu mikropiiri, jonka tehtävät on suunnattu verkkolaskentaan.

ONF Open Networking Foundation.

OSGi OSGi kuvaa modulaarisen järjestelmän ja alustan, joka on dynaaminen komponenttimalli Java-ohjelmoinnille.

OSI Open Systems Interconnection kuvaa tiedonsiirtoprotokollien kerrokset.

OSPF Open Shortest Path First. Reititysprotokolla IP-liikenteelle.

OSS Operations Support System on yrityksen osa, joka vastaa tietoverkon ylläpidosta.

OVSDB Open vSwitch Database. Protokolla. jolla voidaan hallita Open vSwith-kytkimien konfiguraatioita.

Päätelaite Päätelaite on tietoverkon osa, jota käyttäjä käyttää. Esimerkiksi tietokone.

Paketti Termi, jota käytetään tietoliikenteessä datayksiköstä.

Palvelin Päätelaite, joka tarjoaa erilaisia ohjelmallisia palveluita.

PCEP Path Computation Element Protocol. PCEP-protokollaa voidaan käyttää reittien laskemiseen ja muodostamiseen tietoverkossa.

Polku Tietoliikenteen käyttämä reitti verkossa.

Putkistoprosessi Tapa, jolla vuotaulut käsitellään järjestyksessä.

QoS Quality of Service. Tietoverkon laadunhallintaa tarkoittava termi, jolla voidaan viitata tietoliikenteen priorisointiin ja luokitteluun.

Reititin L3-tason paketin välitykseen kykenevä laite.

REST Representational State Transfer on arkkitehtuurimalli, joka perustuu HTTP-protokollaan. REST käyttää HTTP-protokollan metodeja.

RGCE Realistic Global Cyber Environment. Kyberturvallisuuden kehitysympäristö JYVSECTEC-hankkeessa.

RIB Routing Information Base. Taulu, johon on tallennettu reitittimen reitit eri verkkoihin.

SCP Secure Copy on SSH-protokollaan pohjautuva protokolla, jolla voidaan siirtää tiedostoja kahden päätelaitteen välillä.

SDN Software-defined networking.

SNMP Simple Network Management Protocol. SNMP on verkkolaitteiden hall intaan käytettävä protokolla.

Split brain Tilanne, jossa kaksi hallintaa suorittavaa elementtiä menettää yhteyden toisiinsa ja silti jatkavat yhteisen hallittavan elementin hallintaa. Tällöin hallinnoitu elementti saa ohjeita kahdesta paikasta.

SSH Secure Shell on salauksellinen verkkoprotokolla, jolla voidaan hallita tekstipohjaisia päätteitä.

TCAM Ternary Content-addressable Memory on muistityyppi, joka sallii hauissa ykkösten ja nollien lisäksi merkin X. X-merkinnällä tarkoitetaan, ettei ole väliä, onko hakukohdassa kohdassa 1 vai 0.

TCP Transmissin Control Protocol. TCP-protokollalla muodostetaan tietoliikenneyhteyksiä päätelaitteiden välille.

TLS Transport Layer Security on protokolla, jolla voidaan salata tietoliikenne.

TRILL Transparent Interconnection of Lots of Links. Protokolla, jolla L3-tason reititystä käyttämällä muodostetaan suuresta määrästä linkkejä yksi näennäinen IP-aliverkko.

Tunnelointi Näennäinen yhteys toisen tietoliikenneprotokollan välityksellä.

Verkkoelementti Verkkoelementillä tarkoitetaan tietoverkon osaa kuten kytkintä tai reititintä.

VLAN Virtual LAN eli virtuaalinen lähiverkko, jolla voidaan tietoverkko jakaa loogisiin osiin.

VNC Virtual Network Computing. Tätä protokollaa voidaan käyttää tietokoneen etäkäyttöön.

VPN Virtual Private Network on tapa yhdistää yksityisiä verkkoja julkisten verkkojen yli.

VTN Virtual Tenant Network. Looginen verkko, joka kartoitetaan fyysiseen verkkotopologiaan.

Vuo Yhdensuuntainen tietoliikenteen reitti tietoverkossa.

Vuomerkintä OpenFlow-protokollan mukainen toiminto, joka suoritetaan tietoliikennepaketille.

Vuotaulu OpenFlow-protokollan mukainen lista vuomerkinnöitä.

VXLAN Virtual extensible LAN on laajennettu VLAN. VXLAN kasvattaa luotavissa olevien loogisten verkkojen määrää.

XML XML eli Extensible Markup Language on merkintäkieli jossa dokumentit luodaan niin, että ne ovat luettavissa koneellisesti sekä ihmisen toimesta.

XMPP Extensible Messaging and Presence Protocol. XML:ään pohjautuva viestipohjainen keskusteluprotokolla.

1 Opinnäytetyön lähtökohdat

1.1 Tilaaja

Tämän opinnäytetyön tilaajana toimii Jyväskylän ammattikorkeakoulun JYVSECTEC-hanke. Jyväskylän ammattikorkeakoulu (JAMK) on kansainvälinen korkeakoulu, joka työllistää noin 700 henkilöä. Opiskelijoita Jyväskylän ammattikorkeakoulussa on yli 8500. (Jyväskylän ammattikorkeakoulu, n.d.)

JYVSECTEC (Jyväskylä Security Technology) on kyberturvallisuuden kehitykseen, koulutukseen ja tutkimukseen perustettu hanke. JYVSECTEC pitää yllä kehitysympäristöä kyberturvallisuutta varten. Tämä ympäristö tunnetaan nimellä Realistic Global Cyber Environment, RGCE. (JYVSECTEC, n.d.)

1.2 Aihe

Työn tarkoituksena oli kartoittaa Software-defined networking (SDN) –tekniikan soveltuvuutta JAMKin ja JYVSECTEC-hankkeen käyttöön. SDN-tekniikasta haluttiin selvittää sen hyödyt ja haitat verrattuna perinteisiin verkkoinfrastruktuureihin. Lisäksi haluttiin selvittää kuinka SDN-tekniikan hyödyt voisivat auttaa JAMKin ja JYVSECTEC:in toimintaa ja tutkimusta.

Työssä myös pyrittiin selvittämään kuinka JAMK ja JYVSECTEC voisivat kouluttaa SDN-tekniikkaa opiskelijoille ja asiakkaille. Koulutusta varten haluttiin vertailla kahta avoimen lähdekoodin SDN-kontrolleria niiden toiminnallisuuksien osalta.

Tilaajalla määritteli kaksi kontrolleria, joista evaluointi ja toteutus suoritettiin. Nämä kontrollerit ovat OpenDaylight sekä Juniper Networks OpenContrail.

1.3 Opinnäytetyön tavoitteet

Tämän opinnäytetyön tavoitteena oli esittää perusteet SDN-tekniikasta ja sen soveltuvuuden mahdollisuuksia eri käyttökohteisiin. Teoriaosuudessa käsiteltiin SDN-tekniikan perusteet ja siihen vahvasti liittyvää OpenFlow-protokollaa.

Käytännön toteutuksessa tavoitteena oli luoda testiympäristö, jossa voidaan todentaa SDN-tekniikan tuomia hyötyjä verrattuna perinteiseen verkkoinfrastruktuuriin. Toteutuksesta tehtiin opinnäytetyöhön dokumentaatio, joka voisi osaltaan toimia perustana koulutuksen suunnittelussa.

2 Software-defined networking

2.1 Open Networking Foundation

Open Networking Foundation (ONF) on käyttäjälähtöinen yhdistys, jonka tarkoituksena on edistää Software-defined networking (SDN) -tekniikan tietoisuutta ja käyttöönottoa kehittämällä tekniikalle avointa standardia. (ONF Overview n.d.)

ONF koostuu yli 70 jäsenestä, jotka ovat mukana ONF:n SDN-tekniikan kehityksessä. Jäsenistä hyvin tunnettuja verkkoteknologian saralla ovat muun muassa Google, Brocade IBM, NEC Cisco Systems ja Juniper. (Member Listing n.d.)

ONF korostaa avointa yhdistävää kehitystä, joka tapahtuu loppukäyttäjän näkökulmasta. ONF on tunnettu kehittämästään OpenFlow-standardista, jota käytetään SDN-tekniikassa. OpenFlow on ensimmäinen SDN standardi ja sen toimintaa esitellään tarkemmin tämän opinnäytetyön luvussa 4. (ONF Overview n.d.)

2.2 SDN-tekniikan tarpeellisuus

2.2.1 Nykyisten verkkojen rajoittuneisuus

Tämän päivän vaatimuksilla on lähes mahdotonta luoda kaiken täydellisesti kattava verkkoarkkitehtuuri. Taloudelliset resurssit yritysten IT-hankinnoissa on rajoitettuja. Tämä johtaa siihen, että verkkolaitteista yritetään saada kaikki resurssit käyttöön käyttämällä laitetason hallintatyökaluja tai manuaalisia prosesseja. Suuremmilla verkko-operaattoreilla on vastaavia ongelmia liikkuvan Internetin yleistyessä ja kaistanleveyksien kasvaessa. (Software-Defined Networking: The New Norm for Networks 2012, 4-6.)

Nykyiset verkkoinfrastruktuurit eivät ole suunniteltu vastaamaan nykypäiväistä operaattori- tai yritysverkkojen käyttöä. Verkkojen suunnittelua haittaa nykyisten verkkoinfrastruktuurien rajoittuneisuus. (Software-Defined Networking: The New Norm for Networks 2012, 4-6.)

Verkkoteknologia on pitkään koostunut useista erilaisista protokollista, joilla luodaan luotettavia yhteyksiä erilaisten reittien ylitse. Nykypäivänä verkoilta vaaditaan yhä enemmän kaistaa, luotettavuutta ja suorituskykyä. (Software-Defined Networking: The New Norm for Networks 2012, 4-6.)

Kehitettäessä erilaisia protokollia voidaan ratkaista yksinkertaisia ongelmia, mutta sillä ei paranneta yleistä toiminnallisuutta. Useampien protokollien lisääminen monimutkaistaa verkkoja entisestään. Tämä on osaltaan johtanut nykyiseen tilaan, jossa uuden laitteen lisääminen verkkoon voi tuottaa suuren määrän työtä muiden verkossa olevien laitteiden konfiguroinnissa. Kytkimen lisääminen voi johtaa siihen, että verkon muista kytkimistä joudutaan päivittämään esimerkiksi ACL-, VLAN- ja QoS-asetuksia. Lisäksi laitteita ja palveluita lisättäessä on otettava huomioon valmistajakohtaiset yhteensopivuudet. Nykyisten verkkojen monimutkaisuus ajaa niitä kohti staattista tilaa, jossa mitään ei haluta muuttaa. (Software-Defined Networking: The New Norm for Networks 2012, 4-6.)

Verkkojen staattisuus ei vastaa nykyisin kaivattua dynaamista verkkoa, jota tarvitaan liikkuvuuden ja virtualisoinnin takia. Suuressa roolissa oleva virtualisointi vaatii verkolta dynaamisuutta, sillä virtualisoidut palvelimet ovat kasvattaneet palvelimien vaatimien yhteyksien määriä. Lisäksi virtualisoituja tietokoneita voidaan siirtää esimerkiksi pilvipalvelussa, jolloin verkon on mukauduttava kyseisen virtualisoidun tietokoneen siirtoon. (Software-Defined Networking: The New Norm for Networks 2012, 4-6.)

Nykyisten verkkojen ongelmana ovat myös erilaiset politiikat. Kun verkkoon asetettaaan uutta politiikkaa, voi tilanne pahimmillaan vaatia jopa tuhansien laitteiden asetusten muuttamista. Tällöin työ voi kestää tunteja tai päiviä. (Software-Defined Networking: The New Norm for Networks 2012, 4-6.)

Omana ongelmanaan nykyisissä verkkoinfrastruktuureissa voidaan pitää skaalautuvuutta. Konesalipalveluiden kysynnän kasvaessa kasvaa myös verkkoinfrastruktuuri. Kasvu johtaa monimutkaisuuteen, sillä verkkoon joudutaan lisäämään suuria määriä verkkolaitteita. Aiemmin liikennettä on voitu hallita sen ennakoitavuuden mukaisesti. Nykyiset virtualisoidut konesaliratkaisut johtavat liikenteen dynaamisuuteen, ja siksi

liikenne ei ole enää ennakoitavissa. (Software-Defined Networking: The New Norm for Networks 2012, 4-6.)

Lisäksi operaattoreilta odotetaan asiakaskohtaisempia ratkaisuja. Tämä on johtanut siihen, että operaattorien on muokattava verkkojen liikennettä vastaamaan asiakkaiden toiveita. Kuitenkin suurissa operaattoritason verkoissa pienetkin muutokset ovat hyvin monimutkaisia. Tällöin vaaditaan erityisiä verkkolaitteita verkon reunalle. (Software-Defined Networking: The New Norm for Networks 2012, 4-6.)

2.2.2 Tarve verkon uudistamiselle

Palvelinten virtualisointi, pilvipalvelut, mobiilipalvelut ja mobiililaitteiden yleistyminen ajavat uudistamaan perinteisiä verkkoinfrastruktuureja. Perinteisesti verkot on rakennettu käyttäen hierarkkista mallia, jossa laitteista muodostuu puu-tyyppinen rakenne. Tämä perinteinen rakenne on toimiva asiakas-palvelin tyyppisessä tietoliikenteessä. Tämä staattinen rakenne ei tue nykyistä dynaamista mallia, jota tarvitaan konesali-, kampus- ja operaattoriverkoissa. (Software-Defined Networking: The New Norm for Networks 2012, 3-4.)

Konesaliverkkojen tietoliikenteen muodostamat kaavat ovat muuttuneet huomattavasti. Palvelin-asiakasmallin ohjelmissa tietoliikenne tapahtuu pääasiallisesti palvelimen ja asiakkaan välillä. Nykyiset ohjelmat käyttävät useita tietokantoja ja palvelimia, jolloin tietoliikennettä syntyy myös palvelimien välille, ennen kuin liikennettä palautetaan asiakkaalle. (Software-Defined Networking: The New Norm for Networks 2012, 3-4.)

Tietoliikenteen kaavoja muuttavat myös yritysten omat työntekijät. Etätyössä työntekijän on kyettävä yhdistämään milloin vain ja mistä vain yrityksen omaan verkkoon, samalla muuttaen tietoliikenteen kaavoja. (Software-Defined Networking: The New Norm for Networks 2012, 3-4.)

Lisäksi verkossa on otettava huomioon käyttäjien henkilökohtaiset mobiililaitteet, älypuhelimet, tabletti-tietokoneet ja kannettavat tietokoneet, jotka ovat yhteydessä yrityksen verkkoon. Verkon ylläpidon on sovitettava nämä laitteet hienojakoisesti

verkkoon ja samalla suojattava yrityksen sisäisiä tietoja. (Software-Defined Networking: The New Norm for Networks 2012, 3-4.)

Yritykset omaksuvat kasvavissa määrin pilvipalveluita. Yksityisten ja julkisten pilvipalveluiden käytöllä yrityksissä halutaan kehittää palveluiden saatavuutta, verkkoinfrastruktuuria ja muita IT-osaston tarpeita. Tällöin ylläpidossa on otettava huomioon pilven tietoturva, yhteensopivuudet ja auditoinnin tarpeet. Pilvipalvelut kuitenkin vaativat elastisen ja skaalautuvan laskennan, tiedostotilan ja verkkoresurssit. Tämä korostuu yrityksen toteuttaessa pilvipalvelunsa osaksi yksityisenä ja osaksi julkisena pilvipalveluna. (Software-Defined Networking: The New Norm for Networks 2012, 3-4.)

3 SDN-tekniikka

3.1 Arkkitehtuuri

Uusien teknologioiden kanssa käy usein niin, että niiden varsinaisesta määritelmästä ei saada päätettyä. SDN ei ole poikkeus tässä suhteessa. SDN-tekniikan määritelmät vaihtelevat, mutta pääasiallisesti jokainen määritelmä kokee SDN-tekniikan perustana välitys- ja kontrollitason eriyttämisen. (Metzler 2012.)

ONF määrittelee SDN-tekniikan siten, että SDN arkkitehtuurissa kontrolli- ja välitystasot ovat eriytettyinä, verkon älykkyys ja tila on loogisesti keskitetty sekä verkon infrastruktuuri on eriytettynä ohjelmista. (Software-Defined Networking: The New Norm for Networks 2012, 7.)

ONF:n mukaan SDN tekniikasta on monia hyötyjä kuten valmistajasta riippumaton keskitetty hallinta verkkolaitteille, parannettu automaatio sekä laitehallinta käyttämällä yhteisiä ohjelmointirajapintoja. Lisäksi verkon ominaisuudet ja palvelut parantuvat, sillä yksittäisten verkkolaitteiden konfigurointia tai verkkolaitevalmistajien päivityksiä ei tarvita. Hyötyä lisää myös verkkolaitteiden ja yhtenäisten politiikkojen keskitetyn ja automatisoidun hallinnan tuoma verkon tietoturvan sekä luotettavuuden lisääntyminen. SDN-tekniikka mahdollistaa myös hienojakoisemman verkon hallinnan

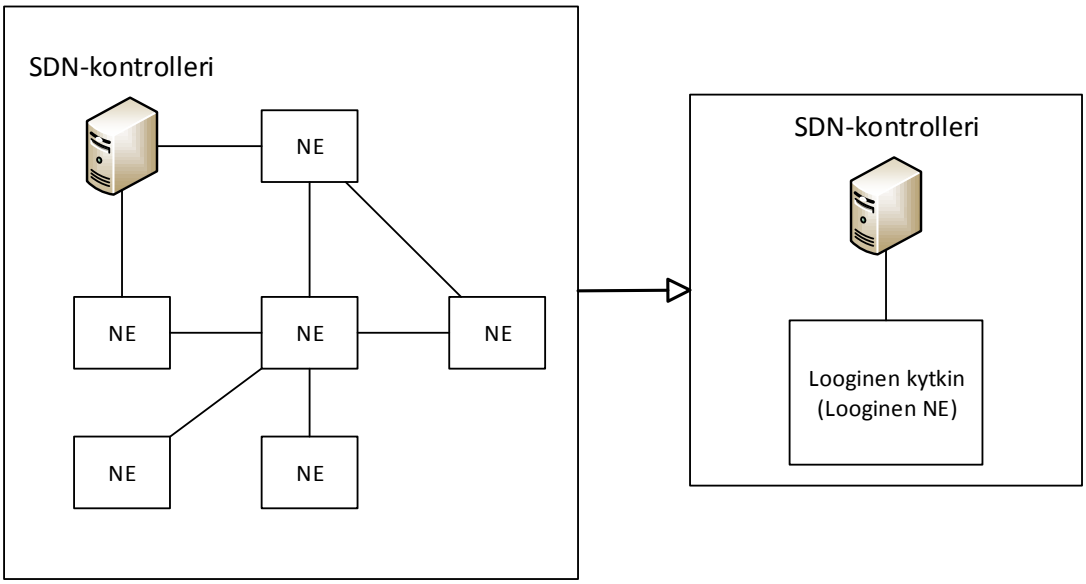
ja kyvyn lisätä politiikkoja useilla eri tasoilla, kuten käyttäjä-, laite- tai ohjelmatasoilla. (Software-Defined Networking: The New Norm for Networks 2012, 10-12.)

SDN-tekniikan arkkitehtuuri koostuu pääasiallisesti kahdesta laitetyypistä ja neljästä verkko-osasta. Laitetyypit ovat kontrolleri ja verkkoelementti (esimerkiksi kytkin tai reititin). Verkkotyypit ovat data-, hallinta-, kontrolli- ja Cluster Interconnect -verkko. (Pepelnjak 2013.)

Nykyisessä verkkoarkkitehtuurissa kontrollitaso on sidoksissa verkkolaitteisiin. SDN-tekniikassa kontrollitaso eriytetään verkkolaitteista erikseen ohjelmoitavaksi ja loogisesti keskitetyksi kokonaisuudeksi. Tällöin Kontrollitaso toimii erillisellä SDN-kontrollerilla. (Software-Defined Networking: The New Norm for Networks 2012, 7.)

Kontrollerin tehtävänä on ylläpitää jatkuvaa tietoa hallinnoidusta verkosta. Tämä tarkoittaa, että hallinnoitu verkko esittäytyy loogisesti keskitetyn kontrollitason myötä yhtenä loogisena kytkimenä. Näkymä yhtenä loogisena kytkimenä helpottaa verkon hallintaa ja yksinkertaistaa hallinnoitujen verkkolaitteiden toimintaa. Keskitetyn kontrollitason ansiosta vain kontrollerin tarvitsee ymmärtää useita protokollia. (Software-Defined Networking: The New Norm for Networks 2012, 7.)

Kuviossa 1 on esitettynä, kuinka hallinnoitu verkko voi rakenteesta huolimatta esittäytyä yhtenä loogisena kytkimenä tai verkkoelementtinä (NE eli Network element).

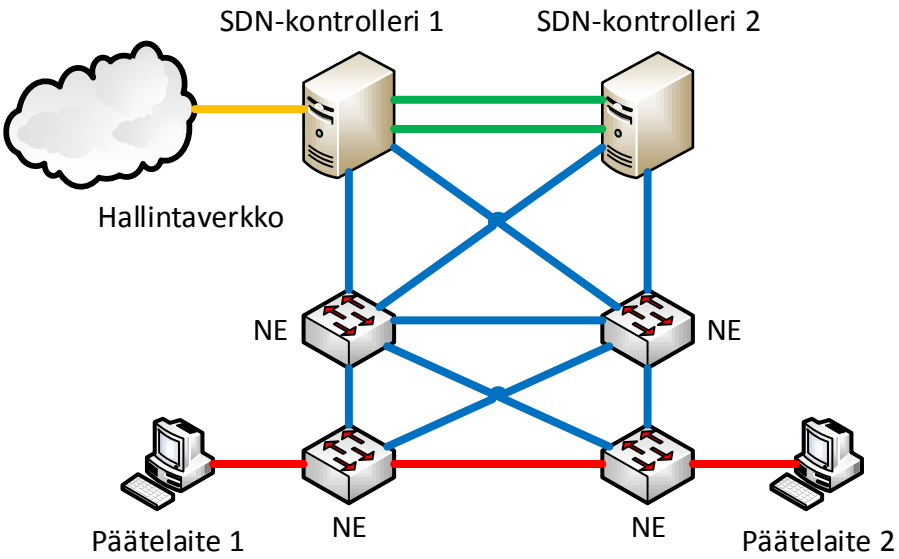


Kuvio 1. SDN-verkko loogisena kytkimenä

SDN-verkon verkkolaitteet ovat yhteydessä erilliseen SDN-kontrolleriin. Verkkolaitteiden ja SDN-kontrollerin välisiä protokollia on useita. Tunnetumpia niistä ovat OpenFlow ja OVSDB. Muita protokollia ovat muun muassa NetConf, YANG ja i2rs. Kontrollerin ja verkkolaitteiden yhteyteen käytettävä protokolla voi vaikuttaa muodostuvaan verkkoarkkitehtuuriin. Esimerkiksi OpenFlow pyrkii keskitettyyn ohjaukseen, kun taas i2rs keskittyy jaettuun ohjaukseen. (What are SDN Controllers? n.d.)

Kuten aiemmin todettiin, SDN-arkkitehtuurissa itse verkko voidaan jakaa neljään osaan: data-, kontrolli-, hallinta- ja cluster interconnect -verkkoon. Verkkolaitteilta voidaan muodostaa redundanttinen yhteys SDN-kontrollerille tai redundanttiselle kontrolleriklusterille. Tätä verkkoa sanotaan kontrolliverkoksi, ja se esittäytyy kuviossa 2 sinisenä. (Pepelnjak 2013.)

Kuviossa 2 on esitettynä, kuinka neljä verkkoa muodostuu yksinkertaisessa redundanttisessa SDN-verkossa. Kyseisessä esimerkkiverkossa on toteutettuna SDN-kontrollerien kahdennus.



Kuvio 2. SDN-verkon rakenne

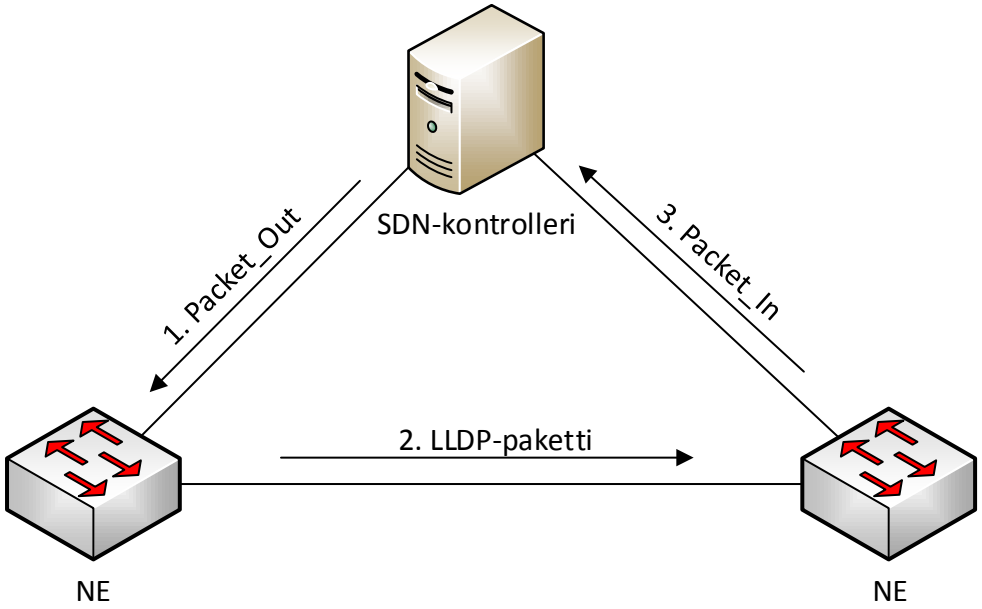
Verkkolaitteisiin kiinnitettyjen päätelaitteiden välille muodostuu SDN-verkossa yhteyksiä. Nämä yhteydet ovat dataverkko, jossa kulkee päätelaitteiden välinen liikenne. Dataverkko esittäytyy kuviossa 2 punaisena. Hallintaverkko on tarkoitettu SDN-kontrollerien hallintaan. Hallintaverkko voi olla kokonaan erillinen yhteys, joka ei ole varsinaisesti yhteydessä kontrollerien hallinnoimaan verkkoon. Hallintaverkko esittäytyy kuviossa 2 keltaisena. (Pepelnjak 2013.)

Cluster Interconnect –verkko on tarkoitettu klusterissa olevien SDN-kontrollerien yhdistämiseen redundanttisesti. Tällöin SDN-kontrollerit voivat vaihtaa tietoa keskenään ja toisen SDN-kontrollerin vikaantuessa tai kaatuessa, toinen kontrolleri voi yhä hallita verkkoa. SDN-kontrollerit tulisi yhdistää toisiinsa redundanttisesti, jolloin vältytään mahdollisen linkkivian aiheuttamalta split brain –tilanteelta. Split brain –tilanteessa yhteys klusteroitujen SDN-kontrollerien välillä katkeaa ja molemmat SDN-kontrollerit luulevat hallinnoivansa verkkoa yksin. Cluster Interconnect -verkko näyttäytyy kuviossa 2 vihreänä. (Pepelnjak 2013.)

3.2 Verkkokuvan muodostaminen

SDN-verkolla voi olla millainen topologia tahansa. Fyysinen topologia voi näyttäytyä yhtenä verkkoelementtinä, joka sisältää useita portteja. Tällöin fyysisestä topologiasta voidaan muodostaa pienempiä verkkoja, jotka sovitetaan SDN-kontrollerin toimesta fyysiseen verkkoon. Tämä tulee ilmi VTN-tekniikassa, joka käsitellään kappaleessa 3.5. Koska SDN-kontrolleri hallinnoi verkkoelementtejä, on SDN-kontrollerin pystyttävä kertomaan verkon fyysinen topologia. Tällöin SDN-kontrolleri kykenee asettamaan verkkoelementeille oikeat asetukset. (Pepelnjak 2013.)

SDN-kontrolleri voi muodostaa kuvan fyysisestä verkon rakenteesta käyttäen LLDP-protokollaa. Kuviossa 3 on esitetty SDN-kontrollerin fyysisen verkonkuvan muodostaminen. (Pepelnjak 2013.)



Kuvio 3. Verkkokuvan muodostaminen

Muodostettuaan yhteyden verkkoelementteihin, SDN-kontrollerilla ei ole tiedossa verkon rakenteesta muuta kuin siihen yhteydessä olevat verkkoelementit. Kontrollerilla on tiedossa verkkoelementin ominaisuudet, kuten käytössä olevat portit. (Pepelnjak 2013.)

Muodostaakseen kuvan fyysisestä verkkotopologiasta SDN-kontrolleri lähettää verkkoelementin kautta LLDP-paketin tietystä verkkoelementin portista käyttäen Packet-Out -viestiä. Tämä on esitettynä kuviossa 3 vaiheena 1. Ensimmäinen verkkoelementti toteuttaa kontrollerin Packet-Out –viestin ohjeet ja lähettää LLDP-paketin toiselle verkkoelementille. Tämä on kuviossa 3 vaihe 2. Toinen verkkoelementti vastaanottaa paketin, muttei tiedä kuinka paketti tulee käsitellä ilman SDN-kontrollerilta saatuja ohjeita, joten se lähettää SDN-kontrollerille tiedon vastaanotetusta paketista Packet-In -viestillä. Tämä on esitettynä kuviossa 3 vaiheena 3. Lähetetty Packet-In –viesti sisältää vastaanotetun paketin tiedot sekä tiedon portista, josta paketti on vastaanotettu. Tällöin SDN-kontrolleri kykenee muodostamaan tiedon siitä, mitkä portit yhdistävät verkkoelementit toisiinsa ja muodostamaan kuvan fyysisestä verkosta. Packet-In ja Packet-Out -viestit ovat OpenFlow-protokollan mukaisia viestejä, ja ne käsitellään tarkemmin kappaleessa 4. (Pepelnjak 2013.)

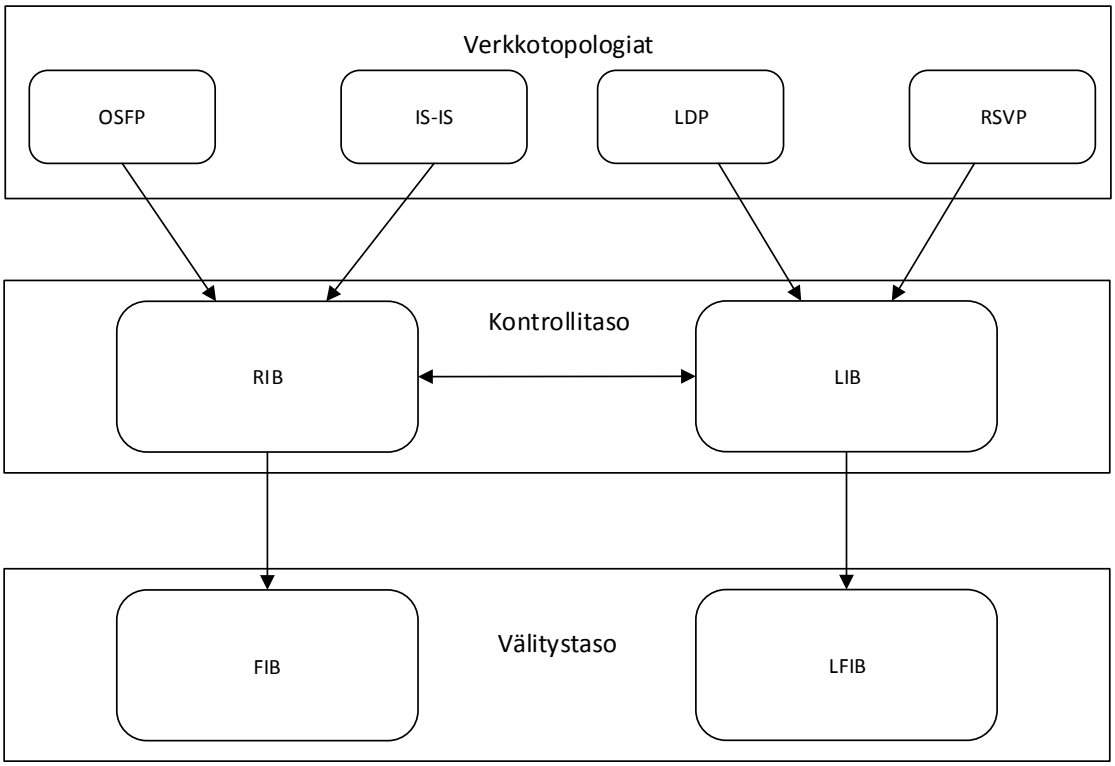
3.3 Verkon tasot

Nykyinen tasomallinen verkkoinfrastruktuuri on rakentunut kolmen verkkolaitteissa toimivan tason pohjalta. Nämä tasot ovat kontrolli-, data- ja hallintataso. Kontrolli- ja datataso ovat ydinosa IP-pakettien välityksessä. (Salisbury 2012a.)

3.3.1 Kontrollitaso

Kontrollitaso voidaan mieltää verkkolaitteen komponenttina. Tämä komponentti keskittyy siihen, kuinka yksittäinen verkkolaite on vuorovaikutuksissa muihin verkkolaitteisiin tilanvaihdossa. Routing Information Database (RIB) ja Label Information Database (LIB) käsitellään ohjelmallisesti. Tämä käsittelyn tuloksia käytetään täyttämään Forwarding Information Database (FIB) ja Label Forwarding Information Database (LFIB). Reititysprotokollat, kuten OSPF tai BGP, käyttävät näitä tietokantoja, mutta niiden käyttötapa ja hyödyntäminen on valmistajakohtaista. RIB ja LIB pitävät sisällään reititysprotokollan tuomia reittitietoja. Reititysprotokollia voi olla käytössä useita, joten reittejäkin voi tällöin olla useita. FIB ja LFIB sisältävät parhaat reitit kaikkien reititysprotokollien tuomista reiteistä, jolloin jokaiselle kohteelle voidaan käyttää parasta mahdollista reittiä. (Salisbury 2012a; Salisbury 2012b.)

Kuviossa 4 on esitettynä kuinka RIB, LIB, FIB ja LFIB sijoittuvat kontrolli- ja välitystasolle.



Kuvio 4. RIB, LIB, FIB ja LFIB

Nykyiset verkkolaitteet kykenevät toimimaan yksittäin. Verkkolaitteen kontrollitaso voi toimia itsenäisesti ilman toista verkkolaitetta, mikäli laitteet sijaitsevat erillisissä hallinnollisissa alueissa. (Salisbury 2012a.)

Kontrollitaso antaa välitystasolle tarvittavat tiedot, jotta välitystaso voi muodostaa välitystaulut ja päivittää tarvittavat topologian muutokset niiden tapahtuessa. Päivitykset eivät tapahdu kovinkaan usein, ja siksi kontrollitasoa voidaan sanoa hitaaksi poluksi perinteisissä verkkoarkkitehtuureissa. Perinteisen reititysprosessorin (routing processor) tehtävät ovat seuraavat:

- Tarvittavien resurssien määrittäminen välitystasolle
- Reititystilan ylläpito
- ARP kyselyiden käsittely
- Kontrollitason tietoturvaominaisuuksien hoitaminen (Näitä ominaisuuksia ovat muun muassa Telnet, SSH ja AAA)
- Hallintainstanssien muodostaminen ja ylläpito
- Reititystilan välittäminen naapuriverkkolaitteille
- Valmistaja ja alustakohtaiset toiminnot kuten klusterointi tai pariutus (Salisbury 2012a.)

3.3.2 Välitystaso

Välitystaso (Data plane tai Forwarding plane) on paketinvälityksen tehokkain osa. Välitystasolla tapahtuva pakettien otsakkeiden käsittely suoritetaan nopeissa ASIC-piireissä. Välitystason tehtäviin kuuluvat muun muassa QoS, pakettien suodatus, kapseloinnin purkaminen ja jonotus. (Salisbury 2012a.)

Välitystasolla tapahtuvien toimintojen on oltava nopeita. Välitystasoa voidaan kutsua siis nopeaksi poluksi. Tarvittava suorituskyky saadaan käyttämällä erilaisia komponentteja ja muistityyppejä. Näitä ovat esimerkiksi BCAM, TCAM ja NPU. (Salisbury 2012a.)

3.3.3 Hallintataso

Hallintataso vastaa siitä, kuinka käyttäjä voi vaikuttaa laitteeseen ja hallita sitä. Käytännössä hallinta voi tapahtua usealla eri tavalla. Verkkolaitteelle voidaan antaa komentoja esimerkiksi käyttäen SNMP, Telnet tai SSH yhteyksiä. (Salisbury 2012a.)

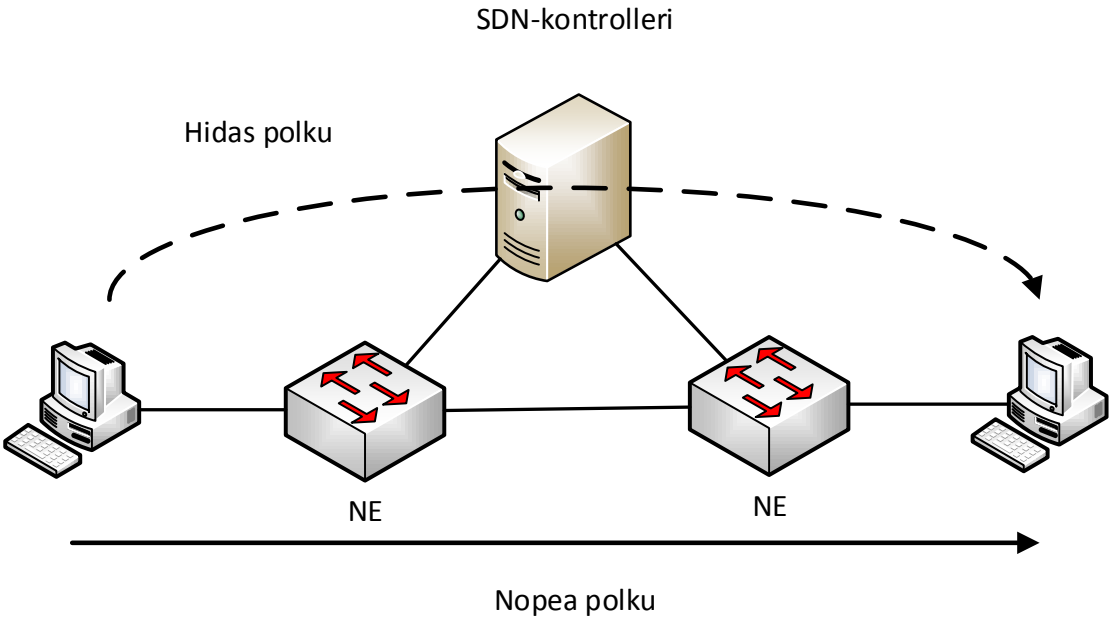
3.3.4 Verkon tasot SDN-tekniikassa

Nopea ja hidas polku ovat hyviä tapoja tarkastella kontrolli- ja välitystasoa. Kontrollitaso suorittaa rajoitetusti päivityksiä FIB- ja LFIB-tauluihin, kun taas välitystaso hakee ja käsittelee paljon suuremmalla taajuudella tietoa FIB ja LFIB tauluista. (Salisbury 2012a.)

SDN-verkossa ensimmäinen paketti joudutaan käsittelemään kontrollerin toimesta. Tämä voidaan mieltää hitaaksi poluksi, sillä kontrolleri joutuu tekemään muutoksia kontrollitasolla ja päivittämään verkon laitteiden välitystason tietokannat. (Salisbury 2012a.)

Seuraaville, saman reitin kulkeville paketeille, on jo olemassa olevat ohjeet. Tällöin ei tarvita kontrollitasoa ja välitys voidaan suorittaa käyttämällä nopeaa polkua eli välitystasoa. (Salisbury 2012a.)

Kuviossa 5 on esitettynä, kuinka nopea ja hidas polku käyttäytyvät SDN-verkossa.



Kuvio 5. Hidas ja nopea polku

3.3.5 SDN-kontrolleri verkon tasoilla

Perinteisissä verkoissa järjestelmät ovat toimineet verkon kanssa keskenään epäsuorasti. Tällöin puhutaan Business- tai Operations Support Systemsistä (BSS/OSS). Näihin sisältyvät erilaiset sopimukset, politiikat ja käyttöoikeudet, jotka halutaan toteuttaa verkossa. BSS tai OSS määritelmien perusteella tehdään sopivat toiminnot ja konfiguraatiot verkolle. Täten voidaan sanoa että OSS/BSS ohjaa SDN-ohjelman toimintaa.(SDN architecture 2014, 13-17.)

SDN-ohjelmat (SDN application) kommunikoivat kontrollitasolla sijaitsevan kontrollerin kanssa. Jokaisella SDN-ohjelmalla voi olla erillinen oikeus hallita SDN-kontrollerin sille paljastamia resursseja. SDN-ohjelmat voivat toimia yhdessä muiden SDN-ohjelmien kanssa. (SDN architecture 2014, 13-17.)

Liikenne kontrollerin ja SDN-ohjelman välillä tapahtuu ohjelma-kontrollerirajapinnan A-CPI, eli Application-controller plane interfacen kautta. Kontrollerin ja välitystason eli verkon fyysisten laitteiden välinen liikennöinti tapahtuu data-kontrollerirajapinnan D-CPI eli Data-controller plane interface kautta. (SDN architecture 2014, 13-17.)

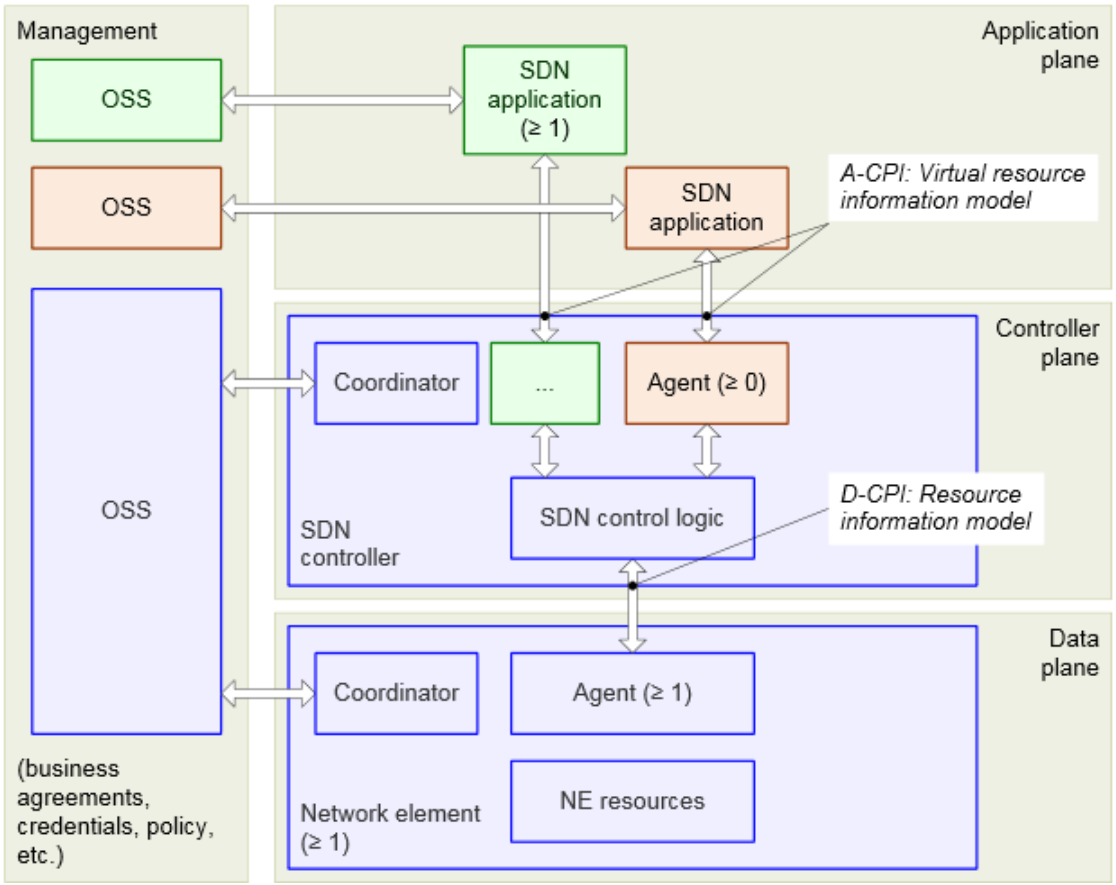
SDN-tekniikassa voidaan kuvata kontrolloitavan ja kontrolloivan komponentin suhdetta kontrolleri-agenttimallilla. Agenttien (agent) tehtävänä on toimittaa tietoa käy-

tettävissä olevista resursseista ja toiminnoista SDN-kontrollerille. Datatasolla toimivat agentit paljastavat suoraan toimintoja SDN-kontrollerille, kun taas ylemmällä tasolla olevat agentit paljastavat SDN-kontrollerin määrittämää tietoa ylemmille tasoille esimerkiksi asiakkaan SDN-ohjelmalle. (SDN architecture, 2014, 13-17, 26.)

Koordinaattorin (coordinator) tehtävänä on asettaa asiakaskohtaiset resurssit ja politiikat, jotka toimittaa OSS tai BSS. Sama tehtävä toistuu koordinaattorilla sekä kontrolli- että datatasolla. (SDN architecture 2014, 13-17.)

SDN-kontrollilogiikan (SDN control logic) tehtävänä on vastaanottaa SDN-ohjelmien pyynnöt ja muuttaa ne ohjeiksi verkkoelementeille (NE, Network Element). (SDN architecture 2014, 13-17.)

Kuviossa 6 on esitettynä kuinka SDN-kontrolleri asettuu verkon tasoille. Lisäksi kuviossa 6 on esitettynä arkkitehtuurin keskeisimmät komponentit ja niiden yhteydet. Kuviossa 6 on eritettynä eri asiakkaat eri värein. Väreistä sininen on palveluntarjoaja, joka omaa itse kontrollerin hallinnan. Punainen ja vihreä ovat palveluntarjoajan asiakkaita. On huomattava, että puhuttaessa SDN-tekniikasta verkontasot (plane) eroavat perinteisestä verkkomallista, jossa puhutaan kerroksista (layer). (SDN architecture 2014, 13-17.)



Kuvio 6. SDN-kontrolleri verkon tasoille (SDN architecture 2014, 15.)

3.4 NFV

Perinteinen verkkoinfrastruktuuri koostuu suureksi osaksi erilaisista suljetuista laitteista. Uusien laitteiden sijoitus verkkoon voi tuottaa ongelmia myös tilan takia. Laitteiden lisäys kasvattaa myös energian kulutusta. Lisäksi rautatason laitteet vanhenevat melko nopeasti tekniikoiden ja protokollien kehittyessä. Edellä mainitut syyt rajoittavat verkkoinfrastruktuurien kehitystä. (Benitez, Bugenhagen, Chiosi, Clarke, Cui, Damker, Delisle, Demaria, Deng, Fargano, Feger, Fukui, Guardini, Khan, Kolias, Loudier, López, Manzalini, Matsuzaki, Michel, Minerva, Ogaki, Reid, Ruhl, Salguero, Sen, Shimano & Willis. 2012, 3-6.)

Ratkaisuna tähän on Network Functions Virtualisation (NFV). NFV pyrkii helpottamaan verkkoinfrastruktuurin muodostamista käyttäen nykyistä virtualisointitekniikkaa. Tämä tapahtuu sijoittamalla verkkoinfrastruktuurin vaatimukset tehokkaille palvelimille, kytkimille ja varastotilaan. Tällaiset resurssit voisivat sijaita esimerkiksi konesaleissa. (Benitez ym. 2012, 3-6.)

NFV käytännössä tarkoittaa verkon toimintojen siirtämistä rautatasolta ohjelmistotasolle. Verkkolaitteet muokataan toimimaan fyysisen laitteen sijasta erilaisena ohjelmistona. Verkkolaitteiden ollessa ohjelmistona, voidaan ohjelmistoa suorittaa erillisillä palvelimilla. Tällöin ohjelmisto eli verkko voidaan implementoida useissa kohteissa ja verkkoa voidaan sekä siirtää että muokata tarpeen mukaan. (Benitez ym. 2012, 3-6.)

NFV tukee ja täydentää SDN-tekniikkaa. NFV-tekniikalla voidaan luoda verkkoinfrastruktuuri, johon SDN-tekniikkaa voidaan soveltaa. NFV- ja SDN-tekniikka eivät kuitenkaan ole toisistaan riippuvaisia. Verkkotoimintojen virtualisointi voidaan toteuttaa ilman SDN-tekniikkaa. (Benitez ym. 2012, 3-6.)

NFV–tekniikalla voidaan toteuttaa verkon toimintoja hyvin laajasti sekä kiinteissä että mobiiliverkoissa. NFV-tekniikalla voidaan toteuttaa muun muassa seuraavia verkon osia tai toimintoja:

- Kytkimet ja reitittimet
- Tunnelointi kuten IPSec ja VPN
- Liikenteen analysointi
- Tietoturvaominaisuudet kuten palomuurit ja IPS/IDS (Benitez ym. 2012, 3-6.)

3.5 VTN

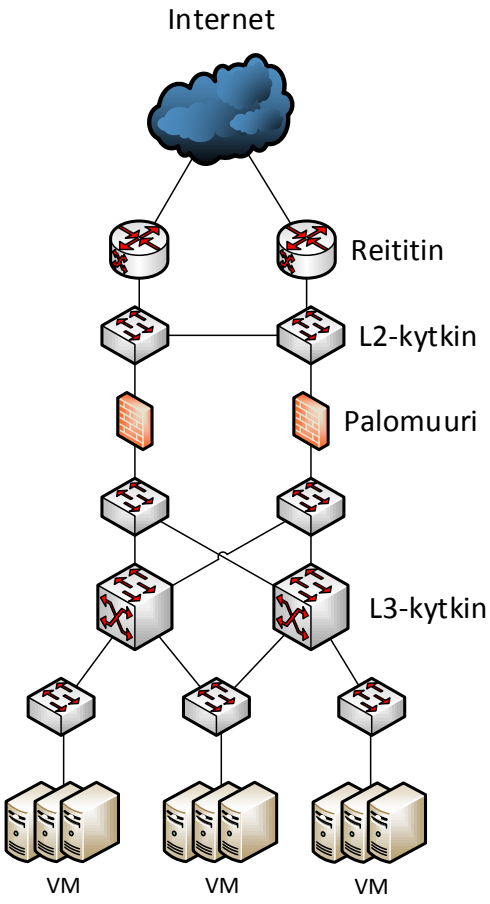
3.5.1 Yleistä

Virtual Tenant Network (VTN) konseptissa tarjotaan asiakkaiden virtuaaliset verkot SDN-kontrollerin toimesta. Kontrolleri asettaa virtuaalisen verkon fyysiseen verkkoon. (OpenDaylight Virtual Tenant Network (VTN):Overview n.d.)

Perinteisissä verkoissa infrastruktuuri joudutaan asentamaan asiakaskohtaisesti ja joitain laitteita ei voida jakaa asiakkaiden kesken. VTN-tekniikassa hyödynnetään SDN-tekniikan kontrolli- ja datatason eriyttämistä. Tällöin voidaan suunnitella ja ottaa käyttöön halutun mallinen verkko, tietämättä verkon fyysistä topologiaa. (OpenDaylight Virtual Tenant Network (VTN):Overview n.d.)

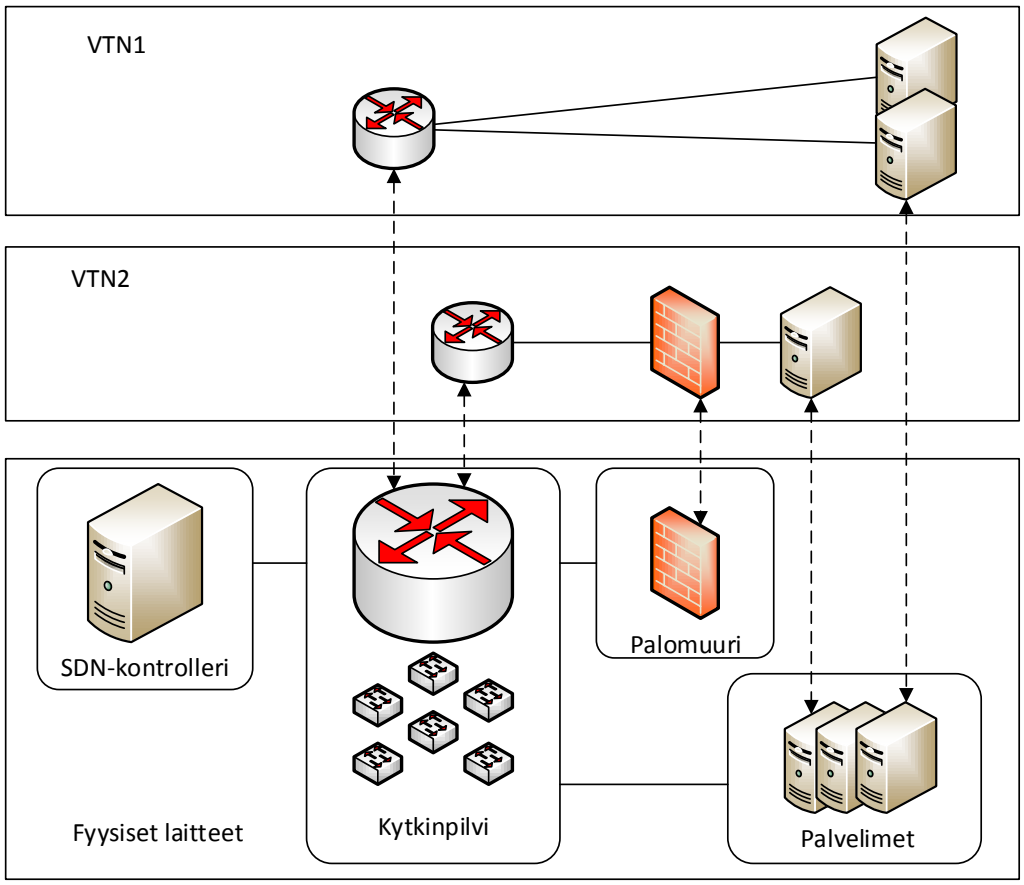
VTN-tekniikassa voidaan suunnitella OSI-mallin mukaisen L2- tai L3-tason verkko. Verkon ollessa halutun mallinen se voidaan automaattisesti kartoittaa fyysiseen verkkoon. Konfiguraatio fyysiselle verkkolaitteille tapahtuu käyttämällä SDN-hallintaprotokollaa, kuten OpenFlow'ta. VTN-tekniikkaa käyttämällä voidaan häivyttää fyysisen verkon monimutkaisuus ja saavutetaan pienemmät konfiguraatioajat sekä palveluiden käyttöönottoajat. Lisäksi minimoidaan verkon konfiguraatiovirheet. (OpenDaylight Virtual Tenant Network (VTN):Overview n.d.)

Kuvioissa 7 ja 8 on esitettynä fyysisen verkon ja VTN-verkon eriytys. Kuviossa 7 on esitettynä kuvitteellinen fyysinen verkko.



Kuvio 7. Fyysinen verkko

Kuviossa 8 on esitettynä VTN-verkkojen yhteydet verrattuna kuvion 7 mukaiseen fyysiseen verkkoon. Halutut VTN-verkot voidaan sovittaa fyysisen verkon rakenteeseen.



Kuvio 8. VTN-verkot

VTN koostuu erilaisista verkkoelementeistä, jotka ovat verrannollisia fyysiseen verkkoinfrastruktuuriin. VTN-verkoilla voidaan tehdä sekä L2-tason, että L3-tason verkkoja. (OpenDaylight Virtual Tenant Network (VTN):Overview n.d.)

Taulukossa 1 on esitettynä VTN-verkon verkkoelementit ja niiden kuvaus. (OpenDaylight Virtual Tenant Network (VTN):Overview n.d.)

Taulukko 1. VTN-verkon elementit

VTN-verkkoelementti		Kuvaus
Virtuaalinen verkkoelementti	vBridge	Looginen L2-tason kytkin
	vRouter	Looginen L3-tason reititin
	vTep	Looginen tunnelin päätepiste (Tunnel End Point, eli TEP)
	vTunnel	Looginen tunneli
	vBypass	Looginen yhteys kahden kontrolloidun verkon välillä
Virtuaalinen rajapinta	rajapinta	Looginen päätepiste virtuaalisella verkkoelementillä
Virtuaalinen linkki	vLink	Looginen L1-tason yhteys virtuaalisten rajapintojen välillä

3.5.2 Verkon kartoitus

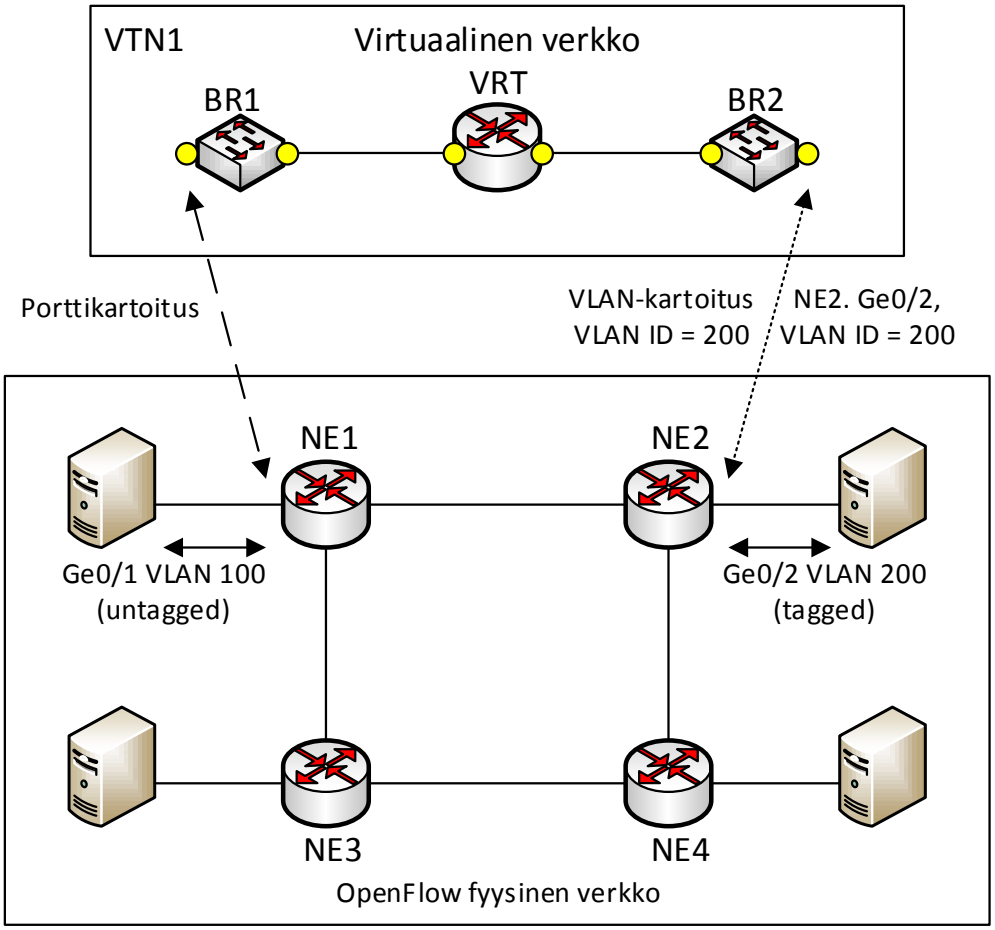
Verkon kartoituksella VTN-verkoissa tarkoitetaan virtuaalisen VTN-verkon sovittamista fyysiseen verkkoinfrastruktuuriin. Kartoitus tunnistaa mihin virtuaaliseen verkkoon kunkin kytkimen lähettämä tai vastaanottama paketti kuuluu. Lisäksi kartoituksessa tunnistetaan verkkolaitteen käyttämät rajapinnat. (OpenDaylight Virtual Tenant Network (VTN):Overview n.d.)

Verkon kartoitus voi tapahtua kahdella tavalla, porttikartoituksella ja VLAN-kartoituksella. Kartoitettaessa rajapintoja vBridge-elementille, käydään ensin läpi porttikartoitus ja sitten VLAN-kartoitus. Kartoitus tapahtuu pakettien saapuessa verkkolaitteelle. (OpenDaylight Virtual Tenant Network (VTN):Overview n.d.)

Porttikartoituksessa kartoitetaan fyysisen verkkoinfrastruktuurin resurssit vBridgen rajapinnalle käyttämällä saapuvan paketin Switch ID:tä, portti ID:tä ja VLAN ID:tä. Kartoitus onnistuu myös ilman VLAN ID:tä saapuvien pakettien kanssa. (OpenDaylight Virtual Tenant Network (VTN):Overview n.d.)

VLAN-kartoituksessa kartoitetaan fyysisen verkkoinfrastruktuurin resurssit vBridgelle käyttäen saapuvan paketin VLAN ID:tä. Tällöin kartoitukseen käytetään switch ID:tä ja VLAN ID:tä. (OpenDaylight Virtual Tenant Network (VTN):Overview n.d.)

Kuviossa 9 on osoitettuna verkonkartoitusta VTN-verkosta fyysiseen verkkoon. Kuvion 9 porttikartoituksessa laitteen NE1 portti Ge0/1 kartoitetaan kuulumaan vBridgeen BR1 ja VLAN-kartoituksessa laitteen NE2 portti Ge0/2 kartoitetaan kuulumaan vBridgeen BR2 käyttäen VLAN-tunnistetta 200.



Kuvio 9. Verkonkartoitus

3.5.3 Flow filter functions

VTN-verkot sisältävät vastaavan toiminnon kuin access control list (ACL). Tätä kutsutaan termillä Flow filter functions. Tällä toiminnolla voidaan tehdä samaa liikenteen hallintaa ja rajoitusta kuin perinteisillä ACL-listoilla. Paketteja voidaan siis kontrolloida muun muassa MAC-osoitteiden, IP-osoitteiden ja DSCP-merkintöjen mukaisesti. Flow filter, eli vuosuodatin, voidaan asettaa minkä tahansa vNoden mihin tahansa rajapintaan. (OpenFaylight Virtual Tenant Network (VTN):Overview n.d.)

Vuosuodatin voi tehdä kolmea toimintoa verrattaville paketeille. Paketti voidaan päästää läpi (Pass), paketti voidaan tiputtaa (Drop) tai paketti voidaan uudelleenohjata (Redirect) toiseen rajapintaan. (OpenFaylight Virtual Tenant Network (VTN):Overview n.d.)

3.5.4 Usean SDN-kontrollerin VTN

Yksittäinen SDN-kontrolleri voi hallita useita VTN-verkkoja. On kuitenkin mahdollista levittää yksittäinen VTN-verkko useamman kontrollerin hallittavaksi. Näin mahdollistetaan suurempi skaalautuvuus. (OpenFaylight Virtual Tenant Network (VTN):Overview n.d.)

Useamman SDN-kontrollerin hallinnoima VTN-verkko mahdollistaa VTN-verkon ulottumisen useampaan maantieteelliseen sijaintiin. Tällöin yhtenäistä politiikkaa noudattava verkko voidaan ulottaa esimerkiksi yrityksen erillisiin toimipisteisiin. Lisäksi VTN-verkkoon on mahdollista lisätä ja poistaa SDN-kontrollereita helpommin. (OpenFaylight Virtual Tenant Network (VTN):Overview n.d.)

3.6 Käyttökohteet SDN-tekniikalle

Operaattorit ja konesalityyppisten palveluiden tarjoajat joutuvat vastaamaan jatkuvaan kasvuun ja kaistan kysyntään. Samalla operaattorien ja palveluntarjoajien tulee pitää kulut mahdollisimman matalina. Tällöin haetaan ratkaisua SDN-tekniikasta. Operaattorit ja palveluntarjoajat voivat yksinkertaistaa verkkoratkaisuja ja virtaviivaistaa operaatioita keskittämällä hallinnan ja tarjoamalla end-to-end palveluita. Automaation ja palvelukeskeisten informaatiomallien sekä API-rajapintojen avulla palveluntarjoajat ja operaattorit voivat lisätä palveluita nopeammin ja vähemmillä virheillä. Lisäksi palveluntarjoajat sekä operaattorit voivat muokata nykyisiä palveluita paremmin. (Carrier Ethernet and SDN 2014, 4; OpenFlow-Enabled Hybrid Cloud Services Connect Enterprise and Service Provider Data Centers 2012, 2-3.)

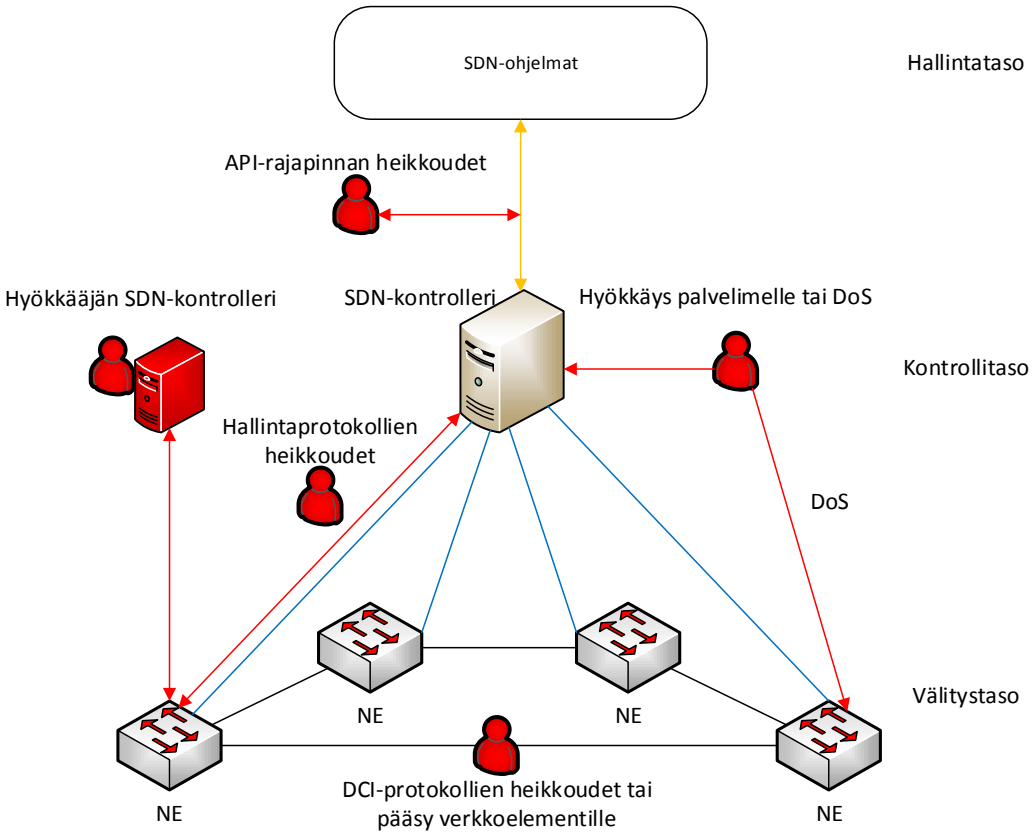
Yksi tulevista SDN-tekniikan käyttökohteista voi olla Network as a Service (NaaS). SDN mahdollistaisi operaattorien tarjota useita virtuaalisia verkkoja eri asiakkaille käyttäen samaa jaettua fyysistä verkkoinfrastruktuuria. Tällöin operaattoritason verkot muuntuisivat samalla tavalla kuin nykyiset konesaliverkot, jossa fyysisiä verkkoresursseja siirretään virtuaalisiksi. Tällöin mahdollistetaan pilvipohjainen ympäristö, jossa asiakkailla voi olla itsepalvelutyyppinen oma verkko. Tällöin operaattori tarjoaisi VTN-tyyppisen verkon, jota asiakas voisi itse muokata omiin tarpeisiinsa. Hubbard 2012, 5.)

3.7 Tietoturva SDN-tekniikassa

3.7.1 Hyökkäysvektorit

SDN-tekniikkaa otettaessa käyttöön nousee pinnalle kysymys sen tietoturvasta. Yritykselle on tärkeää tietää kuinka SDN-tekniikka voi taata, että ohjelmat, data ja infrastruktuuri eivät ole haavoittuvaisia. SDN-tekniikassa tarvitaankin siis uusia tietoturvastrategioita vahvistamaan erityisesti kontrollitasoa. (Hogg 2014.)

SDN-tekniikkaa käyttävät verkot jakautuvat usein kolmeen tasoon. Hyökkäysvektorit voidaan jakaa karkeasti näille kolmelle tasolle. Kuviossa 10 on esitettynä eri hyökkäysvektoreita, joita voidaan käyttää SDN-verkkoja vastaan, sekä niiden sijoittuminen eri verkkotasoille. Lisäksi kuviossa 10 on esitettynä punaisella hyökkääjä sekä hyökkääjän käyttämät laitteet ja hyökkäysvektorit. Vastaavasti verkkoelementtien välinen yhteys SDN-kontrollerille on esitettynä sinisellä ja SDN-kontrollerin yhteys SDN-ohjelmiin oranssilla. (Hogg 2014.)



Kuvio 10. Hyökkäysvektorit

3.7.2 Välitystason hyökkäysvektorit

Ensimmäisenä välitystason hyökkäysvektorina on hyökkääjän pääsy verkkoelementiin joko virtuaalisesti tai fyysisesti. On myös mahdollista, että hyökkääjä saastuttaa verkkoon kuuluvan päätelaitteen. Tällöin hyökkääjän on mahdollista horjuttaa verkon vakauttaa käyttämällä esimerkiksi DoS –palvelunestohyökkäystä. (Hogg 2014.)

Välitystasolla toisena hyökkäysvektorina ovat protokollat, joilla verkkoelementtejä ohjataan (Esimerkiksi OpenFlow, XMPP tai NETCONF). Hyökkääjä voi hyödyntää näitä protokollia ja asettaa verkkoelementeille omia asetuksia. Esimerkiksi OpenFlow-protokollaa käytettäessä hyökkääjä voisi asettaa verkkoelementille omia vuomerkintöjä. Tällöin hyökkääjällä olisi mahdollisuus asettaa verkkoelementti välittämään ei haluttua tietoliikennettä tai kuunnella tietoliikennettä ohjaamalla se haluamallaan tavalla. (Hogg 2014.)

Kolmantena hyökkäysvektorina välitystasolla voidaan pitää konesaleissa tai verkoissa käytettäviä erillisiä Data Center Interconnect (DCI) –protokollia. Näitä ovat muun muassa Virtual Extensible LAN (VXLAN) ja TRILL-pohjaiset protokollat, kuten Juniper Qfabric. Näissä protokollissa voi olla erilaisia haavoittuvuuksia, joita hyökkääjä voi käyttää hyväkseen. (Hogg 2014.)

3.7.3 Kontrollitason hyökkäysvektorit

Kontrollitasolla hyökkääjä voi kohdistaa hyökkäykset SDN-kontrolleriin. Saadessaan haltuunsa SDN-kontrollerin, hyökkääjä voi lisätä omat vuomerkintänsä verkkoelementeille. Tällöin hyökkääjä kiertää tietoliikenteelle asetetut politiikat ja tietoturvan. (Hogg 2014.)

Hyökkääjän on myös mahdollista suorittaa palvelunestohyökkäys SDN-kontrolleria vastaan. Tällöin SDN-kontrolleri hidastuu tai lakkaa vastaamasta verkkoelementeille. Hyökkääjä voi myös kuluttaa SDN-kontrollerin resursseja, täten hidastaen SDN-kontrollerin kykyä käsitellä Packet-In ja Packet-Out viestejä. (Hogg 2014.)

SDN-kontrolleri voi olla hyökkäyksille haavoittuvainen sen oman käyttöjärjestelmän osalta. Yleisesti SDN-kontrolleri voi toimia Linux-käyttöjärjestelmässä, jolloin käyttö-

järjestelmää koskevat haavoittuvuudet ovat myös SDN-kontrollerin haavoittuvuuksia. (Hogg 2014.)

Viimeisenä keinona hyökkääjä voi kontrollitasolla yrittää asettaa verkkoon ulkopuolisen SDN-kontrollerin ja saada verkkoelementit ottamaan vuomerkinnät vastaan omalta kontrolleriltaan. Tällöin hyökkääjä voi asettaa verkkoelementeille omat vuomerkinnät ilman, että vuomerkinnät olisivat havaittavissa varsinaisessa SDN-kontrollerissa. Hyökkääjällä olisi täysi valta tietoverkkoon. (Hogg 2014.)

3.7.4 Hallintatason hyökkäysvektorit

Hallinta- tai SDN-tasolla, hyökkääjä voi käyttää hyväkseen SDN-kontrollerien API-rajapintoja. Näitä ovat muun muassa Java, Python ja REST. Mikäli hyökkääjän on mahdollista käyttää hyväkseen näiden rajapintojen haavoittuvuuksia, voi hyökkääjä saada hallintaansa SDN-verkon ohjaamalla SDN-kontrolleria API-rajapinnan kautta. (Hogg 2014.)

3.7.5 Suojautuminen välitystasolla

Suojautuminen välitystasolla voidaan toteuttaa käyttämällä TLS-protokollaa tunnistautumiseen ja tietoliikenteen salaukseen SDN-kontrollerin ja verkkoelementeissä olevien agenttien välillä. Joillain SDN-kontrollerin ja verkkoelementtien välisillä protokollilla on omat valmistajakohtaiset tavat varmistaa tietoturva. (Hogg 2014.)

DCI-protokollissa voidaan käyttää erillistä tunnelien päätepisteiden tunnistautumista ja tunnelien tietoliikenteen salausta. (Hogg 2014.)

3.7.6 Suojautuminen kontrollitasolla

Kontrollitasolla pääasiallinen hyökkäyskohde on itse SDN-kontrolleri, joten sen tietoturvaa täytyy koventaa. Tyypillisesti tietoturvan koventaminen tapahtuu varmistamalla SDN-kontrollerin käyttöjärjestelmän tietoturva. Vaikka käytössä olisikin best practice -käytänteet, on silti suositeltavaa monitoroida SDN-kontrollereita. (Hogg 2014.)

Kontrolliverkkoon pääsevien henkilöiden määrä tulisi rajoittaa. Tähän voidaan käyttää esimerkiksi roolipohjaista tunnistautumista. Lisäksi tapahtumat tulisivat auditoida ja kirjata logeihin, jolloin voidaan tarkistaa sallimattomat muutokset verkossa. (Hogg 2014.)

Palvelunestohyökkäyksen varalta on hyvä varmistaa SDN-kontrollerin saatavuus. Tämä voidaan toteuttaa esimerkiksi kahdentamalla SDN-kontrolleri, jolloin yhden kontrollerin hajoaminen ei myöskään haittaisi verkon toimintaa. (Hogg 2014.)

3.7.7 Suojautuminen hallintatasolla

Konesaleihin voidaan helposti rakentaa erillinen kontrolliverkko, jolla on yhteys ainoastaan verkkoelementteihin ja kontrolleriin. Tällainen Out-of-band –verkko voi helpottaa SDN-kontrollerin hallintaan käytettävien protokollien suojaamista.

SDN-kontrollerin hallintaan tulisi käyttää TLS-tekniikalla suojattuja yhteyksiä tai SSH-yhteyttä. Tietoliikenne ohjelmilta, jotka pyytävät resursseja ja muutoksia SDN-kontrollerilta, tulisi suojata käyttäen autentikaatio- ja salauskäytänteitä. (Hogg 2014.)

Lisäksi huomiota tulisi kiinnittää SDN-kontrollerille yhteydessä olevien ohjelmien koodiin. Ohjelmien koodi tulisi tarkistaa ja muuttaa tietoturvalliseksi. (Hogg 2014.)

4 OpenFlow

4.1 OpenFlow-komponentit

OpenFlow on ensimmäinen standardi, joka on kehitetty kontrollitason ja välitystason väliseksi rajapinnaksi. OpenFlow mahdollistaa suoran välitystason muokkauksen SDN-verkkolaitteella. (OpenFlow n.d.)

OpenFlow käsittelee verkkolaitteita termillä OpenFlow-switch, eli OpenFlow-kytkin. OpenFlow-kytkin koostuu yhdestä tai useammasta vuotaulusta (flow table) ja ryhmätaulusta (group table). Näiden taulujen ja kontrolleriyhteyden avulla OpenFlow-kytkin suorittaa pakettien välityksen. (OpenFlow Switch Specification 2014, 8.)

Vuotaulut koostuvat vuomerkinnöistä, joita voidaan muokata, lisätä tai poistaa käyttämällä OpenFlow-protokollaa. Muutokset vuomerkintöihin voidaan tehdä reaktiivisesti tai ennakoivasti. Reaktiivisesti tapahtuvat muutokset tehdään saapuvan paketin mukaisesti. Ennakoivassa vuotaulujen asetuksissa on vuotaulujen sisältö lisätty ennen pakettien saapumista. Jokainen vuomerkintä taas sisältää vertailukentät, laskurit ja ohjeet siitä, kuinka paketti tulee käsitellä. (OpenFlow Switch Specification 2014, 8.)

Vertailu alkaa ensimmäisestä vuotaulusta ja voi jatkua useampiin vuotauluihin. Tätä kutsutaan putkistoprosessiksi. Putkistoprosessi on tarkemmin käsitelty tämän opinnäytetyön kappaleessa 4.2. Paketteja vertaillaan vuomerkintöihin prioriteettijärjestyksessä, jossa käytetään ensimmäistä vastaavuuden saanutta vuomerkintää. Mikäli vastaavuuden saanut vuomerkintä löytyy, toteutetaan vuomerkinnän mukaiset toiminnot paketille. Mikäli vastaavuutta ei löydy, on toiminto riippuvainen kytkimen konfiguraatiosta. Paketti voidaan välittää kontrollerille, pudottaa tai välittää seuraavaan vuotauluun. (OpenFlow Switch Specification 2014, 8.)

Kuviossa 11 on esitettynä yksittäinen vuotaulu, jota käytetään pakettien ohjaamiseen putkistoprosessissa.

OpenFlow verkkoelementti
Vuotaulu 0

MAC src	MAC dst	IP src	IP dst	TCP dport	...	Action	Count
*	10:20:.	*	*	*	*	port 1	250
*	*	*	7.7.4.2	*	*	port 2	300
*	*	*	*	25	*	drop	960
*	*	*	192.*	*	*	local	120
*	*	*	*	*	*	controller	14

Kuvio 11. Vuotaulu

Toimintaohjeet vuotauluissa voivat sisältää toimintoja tai muokata putkistoprosessia. Toiminnot sisältävät paketin välityksen, muokkauksen tai ryhmäkäsittelyn. Putkistoprosessin ohjeet sallivat paketin välityksen seuraaville vuotauluille lisäkäsittelyä varten. Lisäksi sallitaan metadatan välitys vuotaulujen välillä. Vuotaulujen putkistoprosessi päättyy kun ohjeet vastaavuuden saaneessa vuomerkinnässä eivät määritä seuraavaa vuotaulua. Tässä vaiheessa paketti on tyypillisesti muokattu ja välitetty. (OpenFlow Switch Specification 2014, 9.)

Vuomerkinnät voivat välittää paketin porttiin. Tämä on yleisesti fyysinen portti, mutta se voi olla myös looginen portti. Lisäksi kolmantena porttityyppinä voi olla varattu portti (reserved port). Varattujen porttien kautta voidaan suorittaa normaaleja välitystoimintoja. Tällaisia välitystoimintoja ovat kontrollerille lähetys, tulvittaminen tai välitys ilman OpenFlow-metodeja. (OpenFlow Switch Specification 2014, 9.)

Vuomerkintöjen toiminnot voivat välittää paketin myös ryhmään, joka määrittää lisää käsittelyä paketille. Ryhmät voivat sisältää toimintoja tulvittamiseen ja muita monimutkaisempia välitystoimintoja, kuten useamman reitin välitys, nopea uudelleen reititys (fast reroute) tai linkkien yhdistämistä (link aggregation). Ryhmät sallivat myös useiden vuomerkintöjen välittää tietoliikennettä yhteen tunnisteeseen. (OpenFlow Switch Specification 2014, 9.)

Ryhmätaulut koostuvat ryhmämerkinnöistä. Jokainen ryhmämerkintä sisältää listan toimintolistoista, jotka riippuvat ryhmän tyypistä. Toiminnot yhdestä tai useammasta toimintolistasta sovelletaan pakettiin, joka on lähetetty ryhmään. (OpenFlow Switch Specification 2014, 9.)

4.2 Putkistoprosessi

OpenFlow-yhteensopivia kytkimiä on kahdenlaisia. OpenFlow-only kytkimet tukevat ainoastaan OpenFlow-protokollan mukaisia toimintoja ja kaikki paketit käsitellään OpenFlow-putkistoprosessin mukaisesti. Näin ollen paketteja ei voida käsitellä muulla tavalla. (OpenFlow Switch Specification 2014, 15.)

Toinen kytkintyyppi on OpenFlow-hybrid. Näissä kytkimissä nimensä mukaan voidaan toimittaa paketin välitystä hybridinä, eli kahdella eri tavalla. Paketit voidaan käsitellä OpenFlow-protokollan mukaisesti tai perinteisen L2-, L3-tason, VLAN, ACL ja QoS menettelyjen mukaisesti. Tällaisissa kytkimissä voidaan määrittää kuinka paketti käsitellään. Esimerkiksi VLAN-leimalla varustetut paketit voidaan ohjata perinteiseen paketin käsittelyyn tai kaikki paketit voidaan ohjata OpenFlow-putkistoprosessiin. (OpenFlow Switch Specification 2014, 15.)

OpenFlow-putkistossa jokainen looginen kytkin sisältää yhden tai useamman vuotaulun ja jokainen vuotaulu usean vuomerkinnän. OpenFlow-putkistoprosessissa määritetään kuinka paketti käsitellään vuotaulujen mukaisesti. OpenFlow-kytkimellä on oltava vähintään yksi vuotaulu. (OpenFlow Switch Specification 2014, 15.)

OpenFlow-kytkimien vuotaulut ovat numeroitu järjestyksessä, alkaen numerosta 0. Putkistoprosessi alkaa aina ensimmäisestä taulusta. Täten siis pakettia verrataan aina vuotaulun 0 vuomerkintöihin. (OpenFlow Switch Specification 2014, 16.)

Putkistoprosessissa pakettia verrataan vuotaulun vuomerkintöihin. Mikäli pakettia vastaava vuomerkintä löytyy vuotaulusta, suoritetaan kyseisen vuomerkinnän mukaiset toiminnot. Nämä toiminnot voivat ohjata paketin käsiteltäväksi toiseen vuotauluun. Putkistoprosessi voi edetä ainoastaan vuotaulujen numeroinnin mukaisesti kasvavaan suuntaan. Tällöin putkistoprosessissa ei voida palata edellisiin vuotauluihin, mutta voidaan hypätä vuotaulujen yli. Näin ollen viimeisen vuotaulun vuomerkinnät eivät voi sisältää ohjeita, jotka ohjaisivat paketin toiseen vuotauluun. (OpenFlow Switch Specification 2014, 16.)

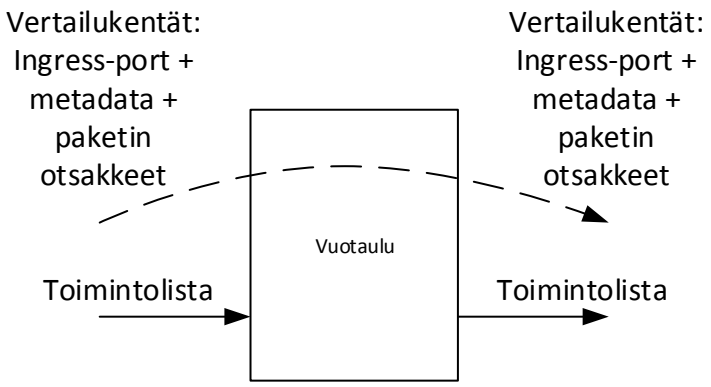
Tapaukset, joissa paketille ei löydy vastaavuutta vuotaulusta, ovat table-miss tilanteita. Tällöin paketille tehtävät toimenpiteet riippuvat OpenFlow-kytkimen konfiguraa-

tiosta. Nämä konfiguraatiot voivat olla muun muassa paketin pudottaminen, lähettäminen toisen vuotaulun käsiteltäväksi tai paketin välitys kontrollerille käyttäen Packet-In viestiä. Packet-In ja muut olennaisimmat OpenFlow-viestit käsitellään tämän opinnäytetyön kappaleessa 4.3. (OpenFlow Switch Specification 2014, 16.)

Muutamassa tapauksessa pakettia ei käsitellä kokonaan vuomerkinnän mukaan ja putkistoprosessi päättyy suorittamatta paketin toimintoja tai välittämättä pakettia toiseen vuotauluun. Mikäli table-miss tilanteita varten ei ole luotu erillistä vuomerkintää, paketti pudotetaan tai se voidaan lähettää kontrollerille. (OpenFlow Switch Specification 2014, 16.)

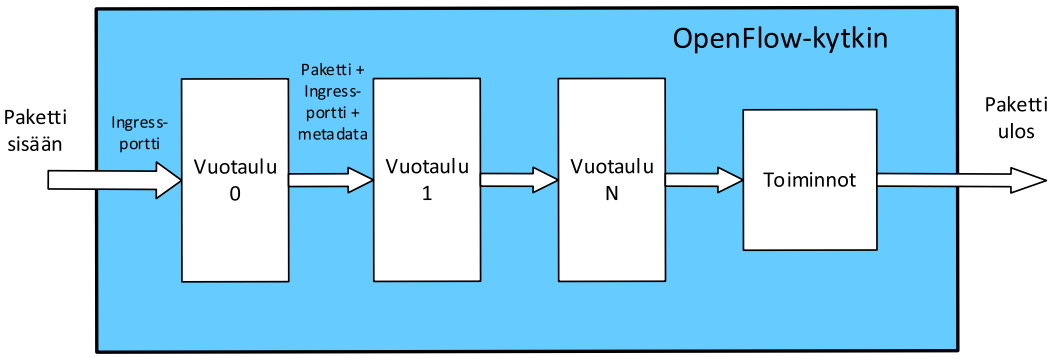
OpenFlow-putkisto ja osa OpenFlow-toiminnoista käsittelee tietyn tyyppisiä paketteja noudattamalla pakettityypin määritelmiä. Esimerkiksi OpenFlow-protokollan käyttämien Ethernet-otsakkeiden on noudatettava IEEE:n määritelmiä sekä TCP/IP-otsakkeiden on noudatettava RFC:n määritelmiä. (OpenFlow Switch Specification 2014, 16.)

Kuviossa 12 on esitettynä kuinka putkistoprosessi etenee yhden vuotaulun tapauksessa.



Kuvio 12. Yhden vuotaulun putkistoprosessi

Kuviossa 13 on esitettynä kuinka putkistoprosessi etenee useamman vuotaulun tapauksessa.



Kuvio 13. Useamman vuotaulun putkistoprosessi

4.3 OpenFlow-perusteet

OpenFlow-protokollassa kontrolleri ja OpenFlow-kytkin keskustelee yhteisen OpenFlow-kanavan yli. Tämän kanavan kautta kontrolleri konfiguroi ja hallinnoi kytkintä. Lisäksi kyseisen kanavan kautta kontrolleri saa tiedon kytkimen tapahtumista ja voi välittää paketteja kytkimelle. OpenFlow-kytkimen kontrollikanava voi tukea yhtä tai useampaa OpenFlow-kanavaa. Useamman kanavan tapauksissa OpenFlow-kytkin voi yhdistää useampaan kontrolleriin yhtä aikaa. OpenFlow-kanava on yleensä salattu käyttämällä TLS-protokollaa, mutta se voi toimia myös suojaamattoman TCP-yhteyden kautta. (OpenFlow Switch Specification 2014, 29.)

OpenFlow-protokolla tukee kolmea viestityyppiä, jotka ovat kontrollerilta kytkimelle, asynkroninen ja symmetrinen. Jokaisessa viestityypissä on useita alityyppejä. Kontrollerilta kytkimelle viestit ovat kontrollerin kytkimelle lähettämiä viestejä, joita pääasiallisesti käytetään OpenFlow-kytkimen hallinnointiin tai kytkimen tilan tarkistukseen. Asynkroniset viestit ovat kytkimen lähettämiä. Näillä viesteillä kytkin päivittää kontrollerille tiedot verkon tapahtumista ja kytkimen tilan muutoksista. Symmetriset viestit voivat olla sekä kontrollerin että kytkimen lähettämiä. (OpenFlow Switch Specification 2014, 29.)

4.3.1 Viestit kontrollerilta kytkimelle

Kontrolleri voi pyytää kytkimeltä identiteetin ja kytkimen ominaisuudet käyttämällä Features-viestiä. Kytkin vastaa pyyntöön lähettämällä pyydetyt tiedot. Features-viestiä käytetään yleensä OpenFlow-kanavan muodostuksessa. (OpenFlow Switch Specification 2014, 30.)

Configuration-viestillä kontrolleri voi asettaa ja pyytää tietoa kytkimen konfiguraatiosta. (OpenFlow Switch Specification 2014, 30.)

Modify-state –viesteillä kontrolleri voi hallita kytkimien tilaa. Näiden viestien pääasiallinen tarkoitus on lisätä, poistaa ja muokata vuo- tai ryhmämerkintöjä sekä muokata kytkimen porttien ominaisuuksia. (OpenFlow Switch Specification 2014, 30.)

Read-state –viesteillä kontrolleri voi kerätä tietoa kytkimeltä. Kerättävä tieto voi olla esimerkiksi kytkimen konfiguraatio, statistiikka tai ominaisuudet. (OpenFlow Switch Specification 2014, 30.)

Packet-Out -viestillä kontrolleri voi välittää paketteja määrittelemäänsä kytkimen porttiin. Tätä viestiä käytetään välittämään Packet-In viestillä kontrollerille saapuneet paketit. Packet-Out –viestien on sisällettävä kokonainen paketti tai puskuritunniste (buffer ID). Puskuritunnisteella voidaan merkitä ja säilöä paketti kytkimelle Packet-Out -viestiä odotettaessa. Packet-Out –viestin on myös sisällettävä järjestyksessä suoritettava lista toiminnoista. Mikäli toimintolista on tyhjä, paketti pudotetaan. (OpenFlow Switch Specification 2014, 30.)

Barrier-viesteistä käytetään sekä pyyntöä että vastausta. Näillä viesteillä varmistetaan, että toiminnot on suoritettu. (OpenFlow Switch Specification 2014, 30.)

Role-request –viestillä kontrolleri voi asettaa oman OpenFlow-kanavansa roolin tai pyytää roolia. Näitä viestejä käytetään yleisesti kun kytkin on yhteydessä useampaan kontrolleriin. (OpenFlow Switch Specification 2014, 30.)

Asynchronous-Configuration –viesteillä kontrolleri voi asettaa suodatuksen asynkronisille viesteille, joita kontrolleri haluaa vastaanottaa OpenFlow-kanavan kautta. (OpenFlow Switch Specification 2014, 30.)

4.3.2 Asynkroniset viestit

Asynkroniset viestit koostuvat pääasiallisesti neljästä erilaisesta viestistä. Näitä viestejä kytkin lähettää kontrollerille, ilman erillisiä pyyntöjä. (OpenFlow Switch Specification 2014, 30-31.)

Packet-In -viesteillä kytkin voi siirtää paketin hallinnan kontrollerille. Pääasiallisesti Packet-In -viesti luodaan table-miss tilanteissa. Muita Packet-In –viestiin johtavia tilanteita on muun muassa TTL-kentän tarkastus. (OpenFlow Switch Specification, 2014 30-31.)

Packet-In tilanteet voidaan konfiguroida puskuroimaan paketteja. Puskuroitaessa paketteja kytkin tarvitsee riittävän määrän muistia. Tällöin Packet-In –viestit sisältävät vain osan paketin tiedosta sekä puskuri tunnisteen (buffer ID), jota kontrolleri voi käyttää Packet-Out –viestien kanssa. Puskurointia tukemattomien kytkinten tulee lähettää koko paketti kontrollerille Packet-In –viestissä. Puskuroidut viestit käsitellään yleensä kontrollerin lähettämällä Packet-Out tai Flow-mod viestillä. Puskuroidut paketit voivat myös vanhentua määritetyn ajan kuluttua. (OpenFlow Switch Specification 2014, 30-31.)

Flow-Removed -viestillä kytkin voi tiedottaa kontrollerille poistaneensa vuomerkinnän vuotaulusta. Flow-Removed –viestit lähetetään kontrollerin toimesta tehdyn vuomerkinnän poiston varmistamiseksi ja vuomerkinnän poistuessa vanhetessaan. (OpenFlow Switch Specification 2014, 30-31.)

Port-status –viestillä kytkin voi ilmoittaa kontrollerille porttien muutoksista. Kytkin lähettää Port-status –viestejä aina porttien tilan muuttuessa. (OpenFlow Switch Specification 2014, 30-31.)

Error-viestillä kytkin pystyy ilmoittamaan kontrollerille mahdollisista ongelmista. (OpenFlow Switch Specification 2014, 30-31.)

4.3.3 Symmetriset viestit

Symmetrisiä viestejä voi lähettää sekä kytkin että kontrolleri ilman erillistä pyyntöä. Hello-viestejä vaihdetaan kytkimen ja kontrollerin välillä niiden muodostaessa yhteyttä. Echo-viesteissä on pyyntö (request) ja vastaus (reply). Näillä viesteillä kontrolleri ja kytkin varmistavat toisen olevan toiminnassa (live). Echo-viesteillä voidaan myös mitata verkon viivettä ja kaistanleveyttä. Experimenter-viestit ovat tarkoitettu tulevillle ominaisuuksille OpenFlow-protokollassa. (OpenFlow Switch Specification 2014, 31.)

5 Evaluointi

5.1 Kontrollerien valinta

SDN-kontrollereita on tarjolla sekä avoimeen lähdekoodiin pohjautuvia että kaupallisia ratkaisuja. Avoimen lähdekoodin kontrollereista ensimmäisenä on NOX, jonka kehitti Nicira Networks. Myöhemmässä vaiheessa Nicira yhdessä NTT:n ja Googlen kanssa kehittivät ONIX kontrollerin. (What are SDN Controllers? n.d.)

Lisäksi avoimen lähdekoodin kontrollereita ovat POX, Beacon, Tream ja Ryu. Myöhemmässä kehitysvaiheessa Beacon kontrollerista haarautui nykyisin tunnetumpi Floodlight. Floodlight on pohjana Big Switch Networksin kaupalliselle SDN-kontrollerille. (What are SDN Controllers? n.d.)

Ensimmäisenä kaupallisena SDN-kontrollerina, jota ei kehitetty avoimen lähdekoodin kontrollereista, oli NEC:n ProgrammableFlow kontrolleri. Myöhemmin markkinoille toivat omat kontrollerinsa myös suuret verkkofirmat kuten Cisco, HP, IBM ja Juniper. HP, Cisco ja IBM kontrollerit pohjautuivat Beacon kontrolleriin, mutta ovat myöhemmin siirtyneet kehityksessä OpenDaylight kontrollerin suuntaan. Juniper muodosti oman kontrollerinsa ostamalla Contrailin. Nykyisin Contrail on saatavilla sekä kaupallisena että avoimen lähdekoodin ratkaisuna. (What are SDN Controllers? n.d.)

2013 OpenDaylight julkaisi oman kontrollerinsa. Nykyisin OpenDaylight on osa Linux Foundationia. OpenDaylight pohjautuu Beacon kontrolleriin. Syyskuussa 2014 julkaistiin OpenDaylightin Helium-versio. Useat yhtiöt toimittavat omia OpenDaylightiin perustuvia kontrollereitaan. Näihin yhtiöihin kuuluvat muun muassa Cisco, Brocade ja Extreme networks. (What are SDN Controllers? n.d.)

5.2 SDN-kontrollerit

Pääasiallisena vaatimuksena valittaville SDN-kontrollereille oli avoimeen lähdekoodiin perustuminen. Muita varsinaisia vaatimuksia ei ollut.

Työntilaajalle aiemmin tehdyn työharjoittelun aikana suoritettiin tutkimusta SDN-tekniikan osalta. Tällöin toteutuksissa päädyttiin käyttämään kahta kontrolleria. Nä-

mä kontrollerit valittiin vertailtavaksi tässä opinnäytetyössä. Näiden kahden kontrollerin osalta voitiin todeta, että saatavilla on hyvä dokumentaatio ja kontrollerit ovat edenneet kehityksessä tarpeeksi pitkälle vastatakseen kaupallisia kontrollereita. Nämä kaksi kontrolleria ovat OpenDaylight ja OpenContrail.

5.2.1 OpenDaylight

OpenDaylight on avoimenlähdekoodin Java-pohjainen SDN-kontrolleri. Kontrolleri on implementoitu täysin ohjelmistoon ja toimii oman Java Virtual Machinen sisällä. OpenDaylight on tällöin mahdollista ottaa käyttöön millä tahansa Javaa tukevalla laitteistolla ja käyttöjärjestelmällä. Kontrolleri on modulaarinen, joustava ja tukee lisäosia. (Technical overview n.d.)

OpenDaylight tukee OSGi-runkoa ja kaksisuuntaista REST-rajapintaa. OSGi-runkoa käyttää kontrollerin sisäiset ohjelmat. REST-rajapinnalla kontrolleri voi kommunikoida kontrollerin ulkopuolisten ohjelmien kanssa. Tällöin ohjelmien ei tarvitse olla edes samalla palvelimella kuin kontrollerin. Logiikka ja algoritmit sijaitsevat ohjelmien sisällä. Ohjelmat käyttävät kontrolleria kerätäkseen tietoa verkosta ja analysoidakseen verkkoa, jotta kontrolleri voi asettaa tarvittavat asetukset verkkolaitteille. (Technical overview n.d.)

OpenDaylight-kontrolleri itsessään koostuu useista dynaamisesti lisättävistä moduuleista, jotka suorittavat verkon toimintoja. Moduulit koostuvat perusmoduuleista, jotka takaavat verkon perustoiminnot. Lisäksi kontrolleriin voidaan asettaa liitännäisiä, joilla kontrollerin toimintoja voidaan lisätä. (Technical overview n.d.)

OpenDaylight-kontrolleri tukee useita protokollia, joilla se kykenee keskustelemaan fyysisten verkkoelementtien kanssa. Näitä protokollia ovat muun muassa OpenFlow 1.0, OpenFlow 1.3 ja BGP-LS. (Technical overview n.d.)

OpenDaylight kontrollerin viimeisin julkaisuversio on nimeltään Helium, jonka ensimmäinen versio julkaistiin 29.7.2014. Aiempi julkaistu versio OpenDaylight kontrollerista on nimeltään Hydrogen, joka julkaistiin 4.2.2014. Hydrogen julkaisusta on tarjolla kolme eri versiota, base, virtualization ja service provider. (Downloads archive n.d.)

Helium julkaisussa kontrolleri sisältää vain perustoiminnot. Näihin toimintoihin kuuluu verkkolaitteiden hallinta, L2-tason paketin välitys, graafinen käyttöliittymä, ja klusterointimahdollisuudet. Samoja ominaisuuksia tukee Hydrogen julkaisun base versio. (OpenDaylight Controller:Architectural Framework 2013; Release/Hydrogen/Base 2014.)

Hydrogen julkaisun virtualization versio tukee perustoimintojen lisäksi VTN-tekniikka, Defense4All DDoS-hyökkäyksen torjuntajärjestelmää ja OpenStack palveluita. Virtualization versio on suunnattu konesaleja varten. (Release/Helium/Virtualization/User Guide 2014.)

Service provider versio Hydrogen julkaisusta tukee perustoimintojen lisäksi useampia protokollia välitystasoa varten. Näitä protokollia ovat BGP-LS, PCEP, LISP ja SNMP. Lisäksi tuettuna on myös Defense4All järjestelmä. (Release/Hydrogen/Service Provider/User Guide 2014.)

5.2.2 OpenContrail

OpenContrail on avoimen lähdekoodin verkkovirtualisointialusta pilvipalveluille. Juniper Networks tarjoaa OpenContrail:ista myös kaupallista versiota. OpenContrail rakentuu useista toiminnallisista osista, jotka voivat sijaita virtuaalisella tai fyysisellä palvelimella. Toiminnallisia osia voidaan toteuttaa useampia, jotta voidaan taata redundanttisuus. (OpenContrail – Quick Start Guide n.d.; Singla & Rijsman n.d.)

OpenContrail voidaan jakaa kahteen pääasialliseen komponenttiin, kontrolleriin ja OpenContrail vRouter:iin. Kontrolleri vastaa verkon hallinta- ja analyysitoiminnoista. vRouter toimii järjestelmän välitystasona. vRouter:in avulla voidaan konesalien fyysiset verkot jatkaa osaksi virtuaalisia verkkoja. vRouter tarjoaa L3-tason verkkomahdollisuudet. (Singla & Rijsman n.d.)

Kontrolleriosa koostuu pääasiallisesti kolmesta toiminnallisesta osasta, jotka ovat konfiguraatio-osa (configuration node), analyysiosa (analytics node) ja kontrolliosasta (control node). Konfiguraatio-osat ylläpitää jatkuvaa tietoa verkolle tarkoitetusta konfiguraatiotilasta ja muuttavat korkeamman tason datamallit sopiviksi, jotta ne voidaan välittää verkkoelementeille. Kontrolliosa pitää yllä hetkellistä verkontilaa.

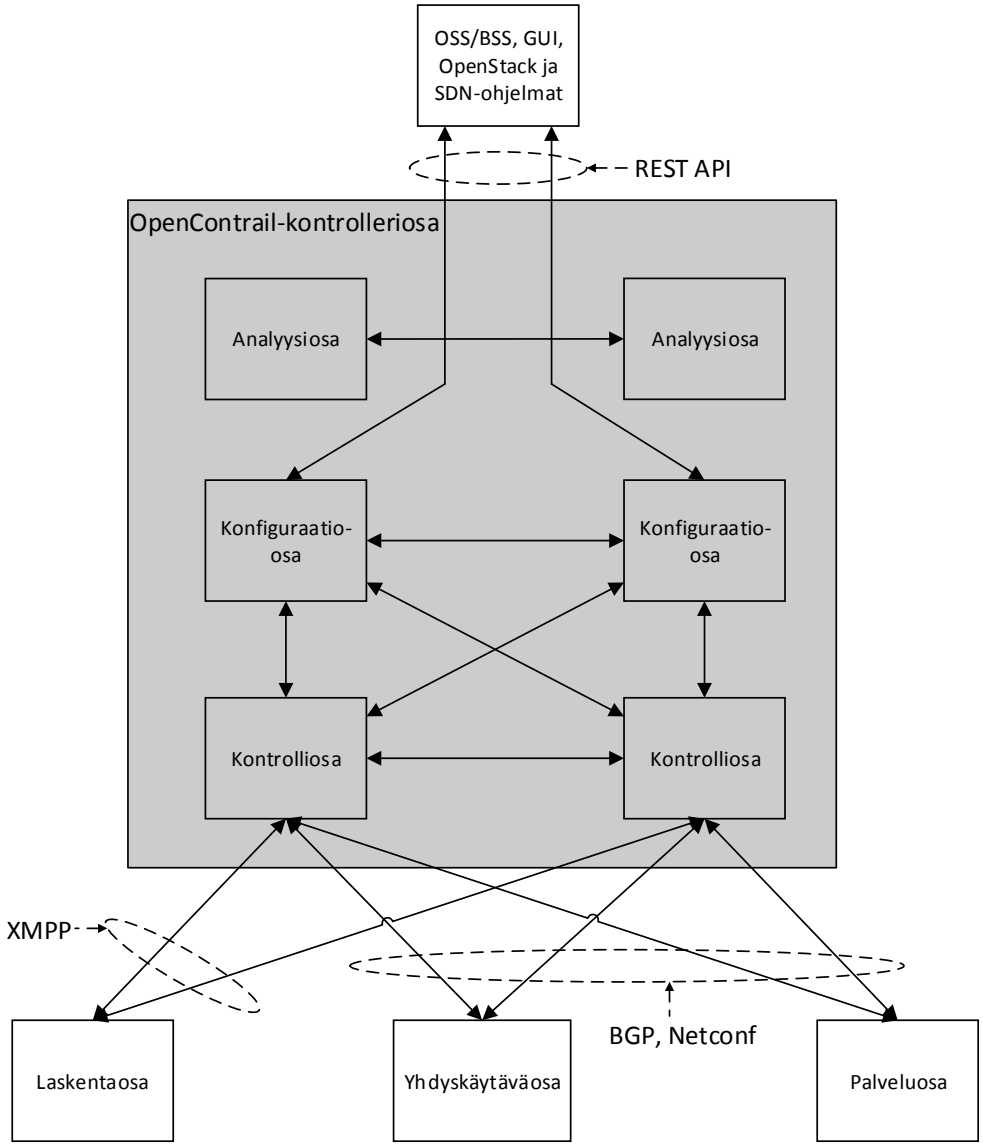
Kontrolliosat ovat vuorovaikutuksissa toistensa sekä verkkoelementtien kanssa, jotta voidaan taata ylläpidettävälle verkolle yhdenmukainen verkontila. Analyysiosat keräävät reaaliaikaista tietoa verkkoelementeiltä ja analysoivat, varastoivat nämä tiedot. Näitä tietoja ovat muun muassa statistiikka, logit, tapahtumat (event) ja virheet. (Singla & Rijsman n.d.)

Kontrolleriosa tarjoaa REST API –rajapinnan, jota erilliset ohjelmat sekä OSS/BSS voivat käyttää kontrollerin kanssa kommunikointiin. Kyseistä rajapintaa käytetään pilvipalveluiden kanssa. OpenContrail järjestelmä tukee integraatiota OpenStack- ja OpenCloud-pilvipalvelujärjestelmien kanssa. Samaa REST API –rajapintaa käyttämällä OpenContrail tukee myös graafista web-pohjaista käyttöliittymää. Kontrolleriosa käyttää XMPP-protokollaa keskustellakseen vRouter-komponenttien kanssa. Lisäksi kontrolleriosa tukee BGP- ja Netconf-protokollia, joita voidaan käyttää kommunikointiin fyysisten kytkimien ja reitittimien kanssa. (Singla & Rijsman n.d.)

XMPP eli Extensible Messaging and Presence Protocol on XML:ään pohjautuva viestipohjainen keskusteluprotokolla. OpenContrail käyttää XMPP:tä laskentaosien ja kontrolliosien välillä, kun osat vaihtavat tietoa reiteistä, konfiguraatiosta, tilasta ja statistiikasta. (Singla & Rijsman n.d.)

OpenContrail-kontrolliosaan voidaan laskea myös kolme erillistä, fyysistä, toiminnallisuusosaa. Nämä osat ovat laskentaosa (compute node), yhdyskäytäväosa (gateway node) ja palveluosa (service node). Laskentaosat ovat virtualisoituja palvelimia, jotka ylläpitävät virtuaalitietokoneita. Kyseiset virtuaalitietokoneet voivat olla pitää sisällään erilaisia ohjelmia tai palveluita kuten virtuaalista palomuuria. Jokainen laskentaosa sisältää vRouter-komponentin. Yhdyskäytäväosat fyysisiä reitittimiä tai kytkimiä, jotka yhdistävät virtuaaliset verkot fyysisiin verkkoihin. Palveluosat ovat fyysisiä verkkoelementtejä, jotka tarjoavat erilaisia palveluja esimerkiksi tietoturvaa. Tällaisia verkkoelementtejä ovat muun muassa fyysiset palomuurit, kuormantasaajat ja IPS/IDS järjestelmät. (Singla & Rijsman n.d.)

Kuviossa 14 on esitettynä OpenContrail-järjestelmän rakenne ja sen sisäiset yhteydet. Lisäksi kuviossa 14 on esitettynä protokollat, joita OpenContrail-järjestelmä käyttää kommunikointiin.



Kuvio 14. OpenContrail-järjestelmä

6 Toteutus

6.1 Toteutuksen ympäristö

Toteutusta varten tarvittavat palvelimet, verkot ja verkkolaitteet virtualisoitiin käyttäen JYVSECTEC-hankkeen VMware vCloud -pilvipalvelua. Toteutuksessa asennetaan sekä OpenDaylight että OpenContrail ja todennetaan niiden toimintaa sekä ominaisuuksia.

Varsinainen toteutus tapahtui tekemällä kaksi erillistä toteutusta, OpenDaylight-kontrolleria ja Mininet-ohjelmaa käyttäen sekä OpenContrail-kontrolleria käyttäen. Molemmissa toteutuksissa esiteltiin kontrollerien perusominaisuuksia ja toimintamahdollisuuksia.

6.1.1 Mininet

Mininet on verkkoemulaattori. Mininetin avulla voidaan emuloida päätelaitteita, kytkimiä, reitittimiä ja linkkejä käyttäen Linux käyttöjärjestelmän kerneliä. Emuloidut päätelaitteet käyttäytyvät kuin oikeat tietokoneet ja niillä voidaan suorittaa erillisiä ohjelmia emuloidussa verkossa. Emuloitavassa verkossa olevien verkkoelementtien ominaisuuksia voidaan muokata halutuiksi. Lisäksi Mininetillä emuloitavia verkkoja voidaan muokata tarpeen mukaan ja siksi se toimii hyvänä testiympäristönä. (Handigol, Heller, Jeyakumar & Lantz n.d.)

Mininet käyttää kytkinten emulointiin Open vSwitchiä. Open vSwitch on ohjelmistopohjainen kytkin, joka tukee ohjelmaliitännäisiä. Open vSwitch tukee paljon erilaisia ominaisuuksia perus kytkintoimintojen lisäksi. Näitä ovat muun muassa OpenFlow, VLAN, QoS ja VXLAN. (Open vSwitch n.d.)

6.1.2 OpenStack

OpenStack on avoimen lähdekoodin pilvipalvelujärjestelmä. OpenStack koostuu pääasiallisesti neljästä osasta, laskenta- (compute), levytila- (storage) ja tietoverkko-osasta (networking) sekä hallintapaneelista (dashboard). (OpenStack: The Open Source Cloud Operating System n.d.)

Laskentaosa vastaa fyysisten resurssien jakamisesta virtualisoitaville päätelaitteille. Näitä fyysisiä resursseja ovat muun muassa prosessorit, keskusmuisti, levytila ja verkkorajapinnat. Lisäksi laskentaosa vastaa virtualisoitavien päätelaitteiden ja niiden levykuvien hallinnasta. (OpenStack Comute n.d.)

Levytilaosa vastaa pilvipalvelun tallennustilasta. Levytilaosan avulla voidaan liittää ja yhdistää uusia tallennusmedioita laskentaosaan. Tällöin levytilaosa itsessään vastaa tallennettavan tiedon redundanttisuudesta sekä tallennusmedioiden hallinnasta. (Open Stack Storage n.d.)

Verkko-osa tarjoaa käyttäjille mahdollisuuden luoda omia tietoverkkoja, hallita tietoliikennettä ja yhdistää päätelaitteita yhteen tai useampaan tietoverkkoon. Verkko-osa tarjoaa lisäksi toiminnot, joilla käyttäjäryhmät voivat luoda ja käyttää omia verkkomalleja. (OpenStack Networking n.d.)

Hallintapaneelin avulla käyttäjien on mahdollista hallita omaa OpenStack-ympäristöään web-pohjaisen käyttöliittymän avulla. Käyttöliittymän kautta käyttäjät voivat hallita ja automatisoida pilvipalvelun resursseja. (Open Stack Dashboard n.d.)

6.2 Todennus

Todennusta varten luodaan testiympäristöt, joissa voidaan todentaa OpenDaylight-kontrollerin ja OpenContrail-järjestelmän toimintaa. Todennuksessa asennetaan sekä OpenDaylight että OpenContrail ja dokumentoidaan niiden toimintaa.

6.2.1 OpenDaylight-ympäristö

OpenDaylight-kontrollerin todennuksessa luotiin ympäristö, jolla voitiin testata OpenDaylight-kontrollerin toiminta, Mininet-verkkoemulaattorin toiminta ja VTN-verkkojen luonti. Lisäksi todennettiin OpenFlow-protokollan liikennettä OpenDaylight-kontrollerin ja emuloidun kytkimen välillä.

Ympäristöä lähdettiin toteuttamaan aluksi OpenDaylight-kontrollerin uusimmalla julkaisulla, Heliumilla. Asentamisen jälkeen havaittiin, ettei kyseinen versio toiminut

täysin halutulla tavalla. Tämän takia päädyttiin käyttämään vanhempaa Hydrogen julkaisua.

Hydrogen versiosta haluttiin VTN-tekniikkaa varten asentaa virtualization versio. Kyseinen versio toimi aluksi hyvin, mutta lopulta lakkasi toimimasta ennen kuin sitä saatiin dokumentoitua. Tämän takia päädyttiin lopulta käyttämään kesällä 2014 toteuttamaani testiympäristöä. Liitteessä 1 on esitetty Hydrogen version OpenDaylight-kontrollerin asentaminen. Kyseinen kontrolleri toimi aluksi, mutta uudelleenkäynnistyksen jälkeen sitä ei saatu enää toimimaan edes uudelleen asentamalla.

Kesällä 2014 toteuttamassani ympäristössä voidaan todentaa OpenDaylight-kontrolleria viidellä virtualisoidulla laitteella, jotka ovat yhteydessä toisiinsa sisäisen verkon kautta sekä Internettiin toisen verkkorajapinnan kautta. Taulukossa 2 on esitettynä virtualisoidut palvelimet, sisäisen verkon IP-osoitteet, käyttöjärjestelmät ja käyttötarkoitukset.

Taulukko 2. OpenDaylight-ympäristön laitteet

Laite	Tarkoitus	Käyttöjärjestelmä	IP-osoite
Centos	Graafinen käyttöliittymä todentamista varten	CentOS 6.5 64-bit minimal	192.168.0.2/24
ODL	OpenDaylight-kontrolleri	Ubuntu 14.04 64-bit server	192.168.0.3/24
ODL2	OpenDaylight-kontrolleri	Ubuntu 14.04 64-bit server	192.168.0.5/24
Mininet-VM	Mininet-verkkoemulaattori	Ubuntu 14.04 server	192.168.0.4/24
Coordinator	Opendaylight koordinaatoori	CentOS 6.5 64-bit minimal	192.168.0.6/24

OpenDaylight ei esitä varsinaisia laitevaatimuksia, joten virtualisoitaville laitteille ei annettu suuria resursseja. Taulukossa 3 on esitettynä virtualisoitujen laitteiden käytössä olevat resurssit.

Taulukko 3. Laitteiden resurssit

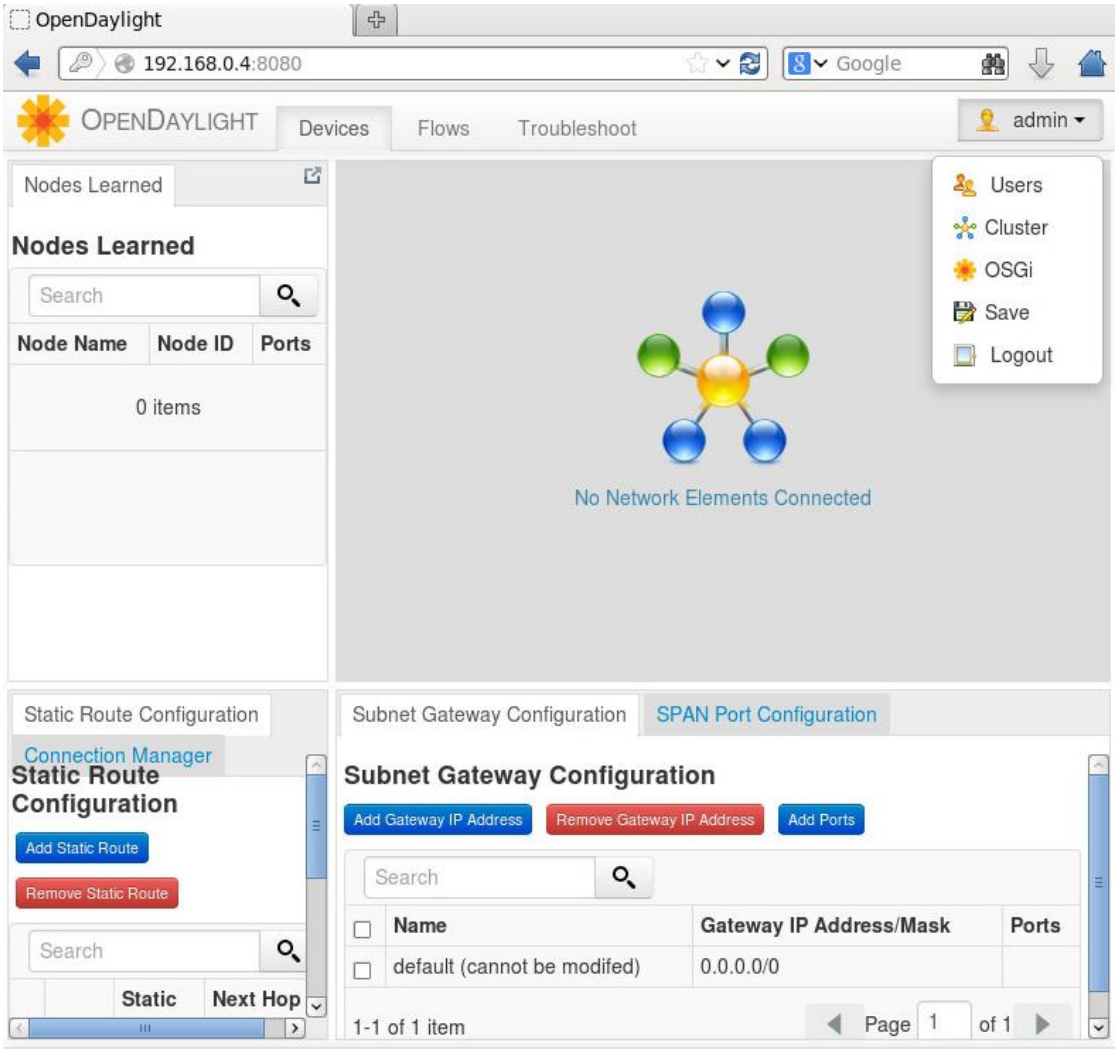
Laite	Levytila	Keskusmuisti	Prosessoriytimet
Centos	10Gb	2Gb	1
ODL	10Gb	4Gb	1
ODL2	10Gb	4Gb	1
Mininet-VM	8Gb	1Gb	1
Coordinator	8Gb	2Gb	1

Kyseisen vanhemman testiympäristön asentaminen on esitettynä liitteessä 2, sillä tässä vaiheessa opinnäytetyötä halutaan todentaa OpenDaylight-kontrollerin toimintaa ja perusominaisuuksia eikä keskittyä uudempien versioiden ongelmien ratkaisemiseen.

Kontrollerin ja Mininetin ollessa asennettuna liitteen 2 mukaisesti voidaan todentaa kontrollerin toimintaa emuloimalla Mininet-VM –laitteella kahden kytkimen verkko ja neljän päätelaitteen verkko. Molempiin kytkimiin yhdistetään kaksi päätelaitetta ja kytkimet osoitetaan OpenDaylight kontrollerin hallintaan.

Aluksi käynnistetään OpenDaylight-kontrolleri liitteen 2 mukaisesti ja otetaan Centos-laitteen webselaimella yhteys OpenDaylight-kontrollerin hallintasivuun kirjoittamalla selaimen osoitekenttään *192.168.0.4:8080*. Tämän jälkeen kirjaudutaan oletuskäyttäjätunnuksella *admin* ja salasanalla *admin*. Kuviossa 15 on esitetty avautuva hallintasivu. Kyseisen *Devices*-hallintasivun kautta käyttäjä voi tutkia hallittua verkkoa, asettaa staattisia reittejä ja hallita oletusyhdyskätäväasetuksia. Oikeassa laidassa olevan pudotusvalikon kautta käyttäjä voi tarkastella klusteroituja OpenDaylight-kontrollereja ja moduuleja. Lisäksi järjestelmänvalvojalla on mahdollisuus hallita käyttäjiä.

Kuviossa 15 näkyvät *Flows*- ja *Troubleshoot*-välilehdet käsitellään myöhemmin tässä luvussa.



Kuvio 15. OpenDaylight hallintasivu

OpenFlow-protokollan toimintaa on helpointa tarkastella kaappaamalla OpenFlow-liikennettä Mininet-VM –laitteella olevan Wireshark-ohjelman avulla. Wireshark käynnistetään Centos-laitteella, jotta voidaan käyttää graafista käyttöliittymää. Tämä tapahtuu ottamalla Centos-laitteella SSH-yhteys Mininet-VM –laitteeseen lisäoptiolla *–X*. Kyseinen optio mahdollistaa X11-tunneloinnin, joka tarvitaan graafisten käyttöliittymien käyttöön etänä SSH-yhteyden yli. Seuraavilla komennoilla otetaan tarvittava SSH-yhteys Mininet-VM –laitteeseen ja käynnistetään Mininet-VM –laitteella Wireshark-ohjelma käytettäväksi Centos-laitteella. Komennot syötetään Centos-laitteella avattavaan terminaaliin.

```
[root@Centos ~]# ssh –X mininet@192.168.0.3
mininet@mininet-vm:~$ sudo wireshark
```

Kun Wireshark-ohjelma on käynnistynyt, valitaan kaapattavaksi verkkorajapinnaksi sisäverkkoon kytkettynä oleva verkkorajapinta, joka tässä tapauksessa on *eth1*. Lii-

kenteen kaappaus voidaan aloittaa heti. Koska verkossa voi liikkua muutakin tietoliikennettä, karsitaan näytettävät paketit OpenFlow-protokollaan asettamalla Filter-kohtaan ”of”.

Tämän jälkeen luodaan Mininet-VM –laitteella haluttu verkko. Verkko voidaan muodostaa erillisellä python-skriptillä tai käynnistää suoraan komennolla. Python-skriptit mahdollistavat Mininet-emulaattorilla monimutkaisempien verkkojen muodostamisen. Tässä todennuksessa käynnistetään haluttu verkko antamalla Mininet-VM laitteella seuraava komento.

```
mininet@mininet-vm:~$sudo mn --topo=tree,2,2 --controller=remote,ip=192.168.0.4
```

Edellisellä komennolla käynnistetään lineaarinen kahden kytkimen ja kahden päätelaitteen verkko. Mininet itsessään sisältää SDN-kontrollerin, jota voidaan käyttää halutessa. Tässä todennuksessa halutaan käyttää OpenDaylight-kontrolleria, joka sijaitsee toisella laitteella. Tämän takia komennossa on määritetty, että kontrolleri on ulkoinen ja sen IP-osoite on 192.168.0.4. Kuviossa 16 on esitettynä verkon käynnistys Mininet-vm –laitteella. Verkkoon muodostetaan kytkimistä s1 ja s2 sekä päätelaitteista h1s1, h1s2, h2s1 ja h2s2.

```
mininet@mininet-vm:~$ sudo mn --topo=linear,2,2 --controller=remote,ip=192.168.0.4
*** Creating network
*** Adding controller
*** Adding hosts:
h1s1 h1s2 h2s1 h2s2
*** Adding switches:
s1 s2
*** Adding links:
(h1s1, s1) (h1s2, s2) (h2s1, s1) (h2s2, s2) (s1, s2)
*** Configuring hosts
h1s1 h1s2 h2s1 h2s2
*** Starting controller
*** Starting 2 switches
s1 s2
*** Starting CLI:
mininet> _
```

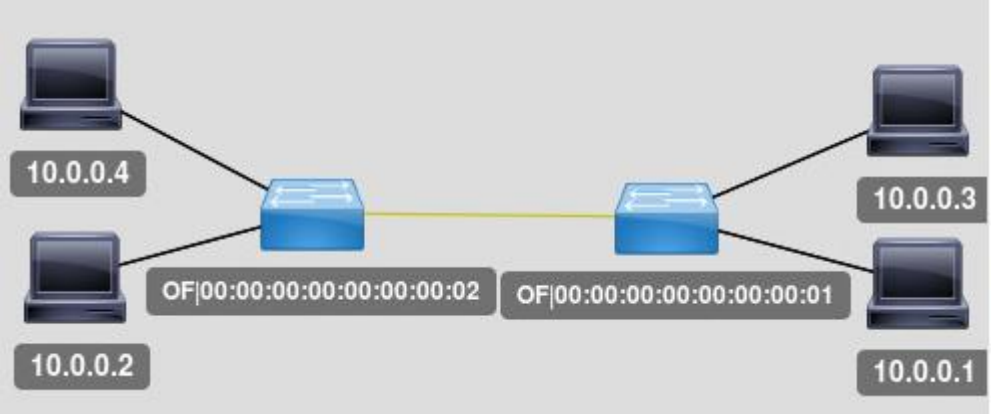
Kuvio 16. Verkon käynnistys Mininet-VM -laitteella

Tämän jälkeen käynnistettyä topologiaa voidaan testata antamalla Mininet-VM –laitteella kuvion 17 mukaisesti *pingall*-komennolla. Kyseinen komento suorittaa *ping*-komennon jokaiselta päätelaitteelta toisille päätelaitteille. Kuviosta 17 käy ilmi, että kaikki päätelaitteet kykenevät liikennöimään keskenään.

```
mininet> pingall
*** Ping: testing ping reachability
h1s1 -> h1s2 h2s1 h2s2
h1s2 -> h1s1 h2s1 h2s2
h2s1 -> h1s1 h1s2 h2s2
h2s2 -> h1s1 h1s2 h2s1
*** Results: 0% dropped (12/12 received)
mininet> _
```

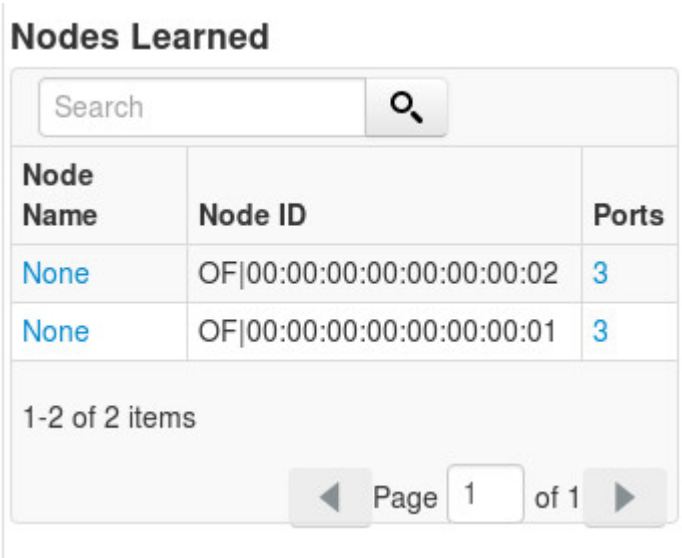
Kuvio 17. Mininet pingall-komento

Nyt on mahdollista tarkastella verkkoa OpenDaylight-hallintasivulta. Kuviossa 18 on esitettynä käynnistetyn verkon näkyminen OpenDaylight-kontrollerilla *pingall*-komennon jälkeen.



Kuvio 18. Näkymä OpenDaylight-kontrollerilla

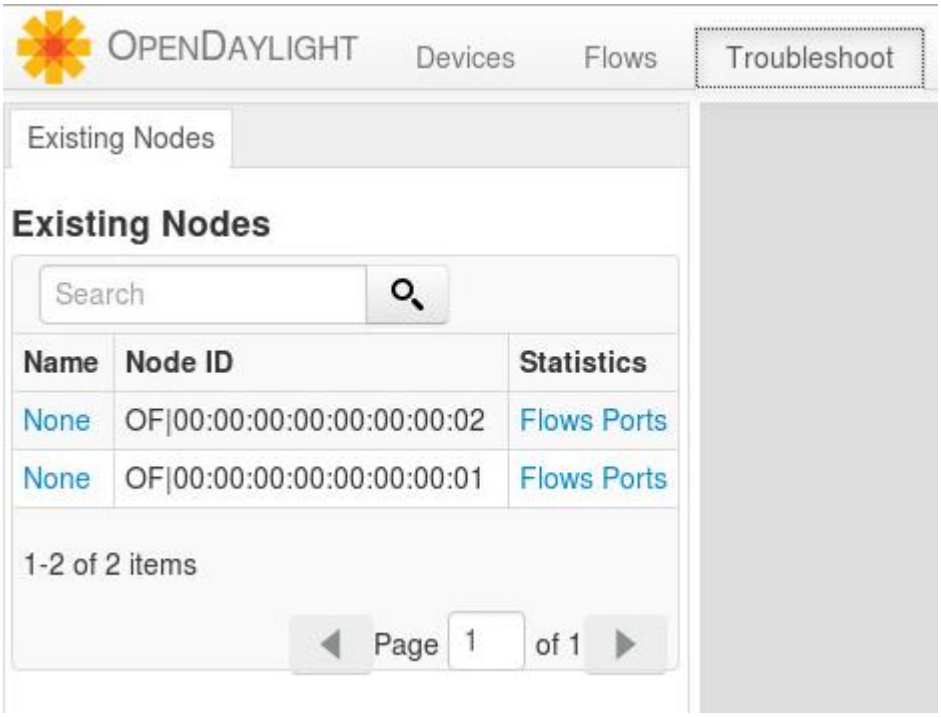
Hallintasivulta voidaan myös nähdä kontrollerin havaitsemat verkkoelementit listana. Lisäksi listasta käy ilmi verkkoelementin toiminnassa olevat portit. Kuviossa 19 on esitetty havaittujen verkkoelementtien näkyminen valikossa.



Kuvio 19. Havaitut verkkoelementit

Edellisessä kuviossa esiintyvää *None*-painiketta painamalla voidaan lisätä verkkoelementille lisätietoja. *None*-painikkeesta aukeaa valikko, josta voidaan määrittää verkkoelementin nimi, hierarkkinen taso (Access, Distributin, Core) ja toimintatapa. Toimintatavalla tarkoitetaan reaktiivista tai ennakoivaa tapaa lisätä vuomerkinnät verkkoelementille.

Hallintasivulta voidaan siirtyä *Troubleshoot*-välilehdelle. Tämän välilehden kautta voidaan tarkastella kytkimillä olevia vuomerkintöjä ja porttien statistiikkaa. Kuviossa 20 on esitettynä *Troubleshoot*-välilehti ja OpenDaylight-kontrollerin havaitsemat kytkimet, joista voidaan tutkia vuomerkintöjä ja portteja painamalla *Flows*- tai *Ports*-painiketta.



Kuvio 20. Troubleshoot-välilehti

Flows-painiketta painamalla voidaan tutkia tarkemmin kytkimelle asetettuja vuomerkintöjä. Liitteessä 3 on esitetty OF|00:00:00:00:00:00:00:02 kytkimen vuomerkinnät Mininet-VM –laitteella suoritetun *pingall*-komennon jälkeen. Vuomerkinnöistä voidaan helposti havaita muun muassa, että mistä tahansa lähetetyt, IP-osoitteeseen 10.0.0.2 tarkoitetut IPv4-paketit, välitetään portista 1 (OUTPUT = OF|1).

Kuviossa 21 on esitetty OF|00:00:00:00:00:00:00:02 kytkimen porttikohtaiset tiedot, jotka saadaan esille *Ports*-painiketta painamalla. Näistä tiedoista voidaan havaita

muun muassa lähetetyn ja vastaanotetun tietoliikenteen määrä sekä mahdolliset virheet ja pakettien pudotukset.

Ports

Node Connector	Rx Pkts	Tx Pkts	Rx Bytes	Tx Bytes	Rx Drops	Tx Drops	Rx Errs	Tx Errs	Rx Frame Errs	Rx OverRun Errs	Rx CRC Errs	Collisions
OF|3@00:00:00:00:00:00:00:02	40	43	3920	4214	0	0	0	0	0	0	0	0
OF|1@00:00:00:00:00:00:00:02	32	66	1896	5068	0	0	0	0	0	0	0	0
OF|2@00:00:00:00:00:00:00:02	34	66	1992	5068	0	0	0	0	0	0	0	0
SW|0@00:00:00:00:00:00:00:02	0	35	0	3500	0	0	0	0	0	0	0	0

Page 1 of 1

Kuvio 21. Kytkimen porttistatiikka

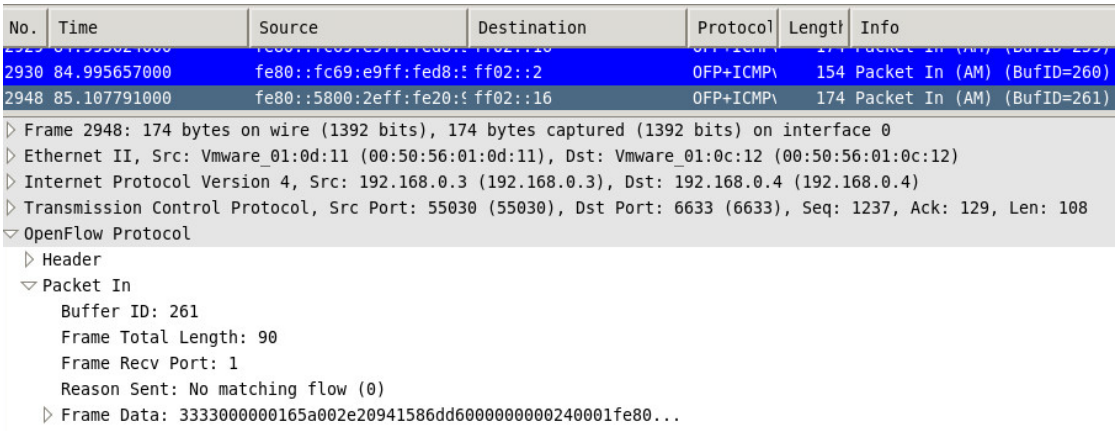
Heti verkon käynnistyksen jälkeen Wireshark-ohjelmassa havaittiin OpenFlow-paketteja. Kuviossa 22 on esitettynä kaapattua liikennettä.

Source	Destination	Protocol	Length	Info
192.168.0.3	192.168.0.4	OFP	74	Hello (SM) (8B)
192.168.0.4	192.168.0.3	OFP	74	Hello (SM) (8B)
192.168.0.4	192.168.0.3	OFP	74	Features Request (CSM) (8B)
192.168.0.3	192.168.0.4	OFP	290	Features Reply (CSM) (224B)
192.168.0.4	192.168.0.3	OFP	78	Set Config (CSM) (12B)
192.168.0.4	192.168.0.3	OFP	74	Get Config Request (CSM) (8B)
192.168.0.3	192.168.0.4	OFP	78	Get Config Reply (CSM) (12B)
192.168.0.4	192.168.0.3	OFP	138	Flow Mod (CSM) (72B)
192.168.0.3	192.168.0.4	OFP	130	Port Status (AM) (64B)
192.168.0.4	192.168.0.3	OFP	86	Stats Request (CSM) (20B)
192.168.0.3	192.168.0.4	OFP	494	Stats Reply (CSM) (428B)

Kuvio 22. OpenFlow-paketteja

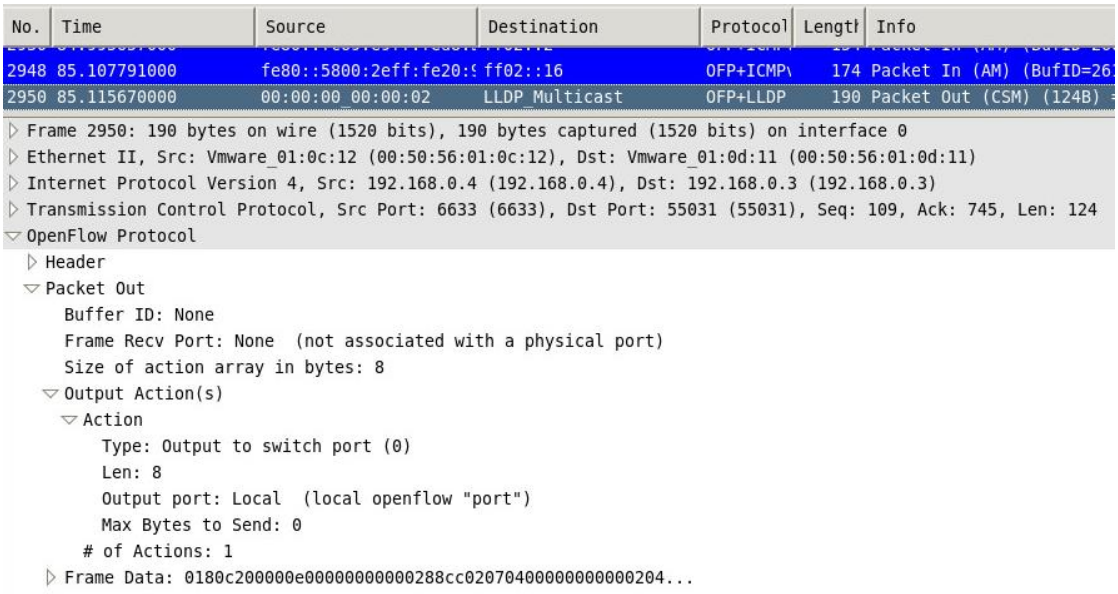
Kuvion 22 mukaisesti kytkimet ja OpenDaylight-kontrolleri muodostavat yhteyden käyttämällä *Hello*-viestejä. Tämän jälkeen OpenDaylight-kontrolleri pyytää kytkintä lähettämään tiedot kytkimen tukemista ominaisuuksista *Features Request* –viestillä. Tähän kytkin vastaa lähettämällä tiedon tukemistaan ominaisuuksista *Features Reply* –viestillä. Seuraavana OpenDaylight-kontrolleri asettaa konfiguraation kytkimelle *Set Config* –viestillä ja pyytää heti tiedot kytkimen konfiguraatiosta *Get Config Request* –viestillä. Tähän kytkin vastaa lähettämällä tiedot konfiguraatiosta *Get Config Reply* –viestillä. Viimeisenä OpenDaylight-kontrolleri muokkaa kytkimen vuomerkintoja *Flow Mod* –viestillä, johon kytkin vastaa porttitilan muutoksesta syntyvällä Port Status –viestillä. Lopuksi OpenDaylight-kontrolleri pyytää tiedon kytkimen statistiikasta *Stats Request* –viestillä ja kytkin vastaa lähettämällä statistiikkatiedot *Stats Reply* –viestillä.

Tässä vaiheessa Mininet-VM –laitteella suoritettiin *pingall*-komento, joka aiheuttaa OpenDaylight-kontrollerilla topologiamuutoksen kontrollerin havaitessa uusia laitteita. Tämä on havainnoitavissa *Packet In*- ja *Packet Out* –viesteinä Wireshark-ohjelmassa. Kuvioissa 23 ja 24 ovat esitettyinä *Packet-In*- ja *Packet Out* –viestit.



Kuvio 23. Packet In -viesti

Packet In –viestistä voidaan todeta, että paketti on vastaanotettu kytkimen portista 1 ja syy *Packet In* –viestille on *table-miss* tilanne eli paketille ei löytynyt vastaavuutta vuotaulusta.



Kuvio 24. Packet Out -viesti

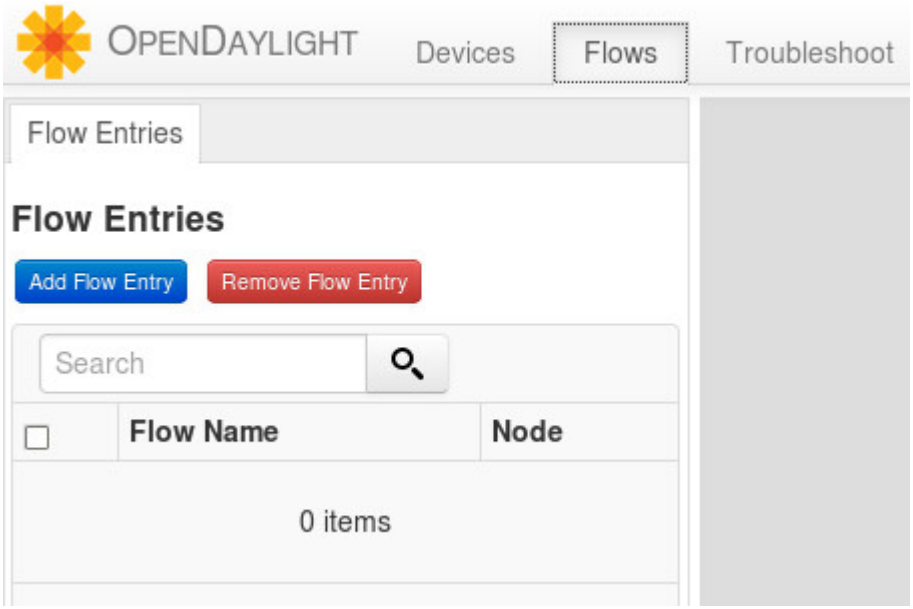
Packet Out –viestistä voidaan todeta, että kytkin ohjeistetaan lähettämään kyseinen paketti paikallisesta openflow-portista 0.

Kaapatusta liikenteestä voidaan myös havaita tilanne, jolloin verkossa ei liiku tietoliikennettä. Tällöin OpenDaylight-kontrolleri ja kytkin vaihtavat *Echo Request*- ja *Echo Reply* –viestejä pitääkseen yllä tietoa toistensa toiminnassa olosta. Kuviossa 25 on esitettynä kaapatut *Echo Request*- ja *Echo Reply* –viestit.

Source	Destination	Protocol	Length	Info
192.168.0.3	192.168.0.4	OFP	74	Echo Request (SM) (8B)
192.168.0.4	192.168.0.3	OFP	74	Echo Reply (SM) (8B)

Kuvio 25. Echo-viestit

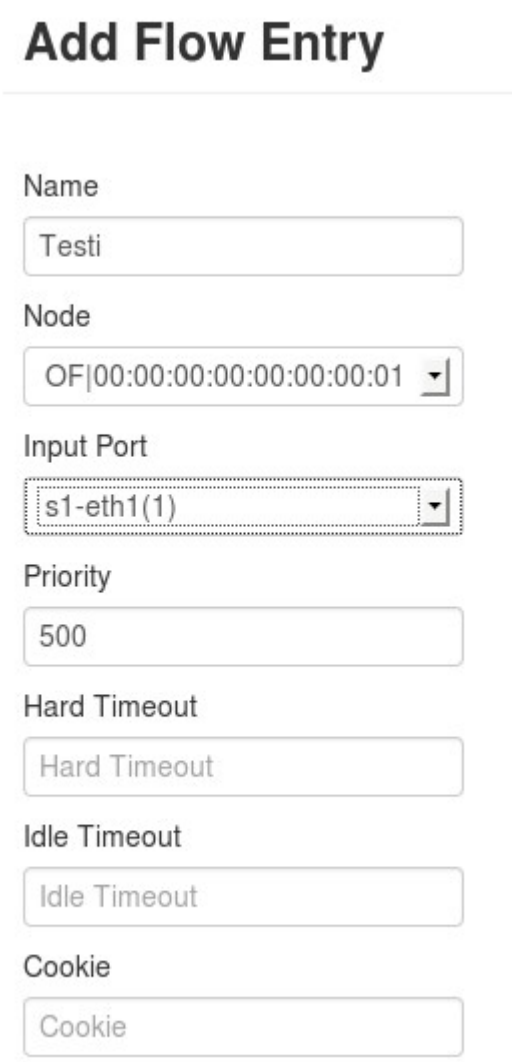
OpenDaylight-kontrollerin hallintasivulta voi käyttäjä luoda ja poistaa omia vuomerkintöjä *Flows*-välilehden alta. Kuviossa 26 on esitettynä *Flows*-välilehti ja vuomerkintöjen tekemiseen ja poistamiseen tarkoitettu valikko. Kuviosta 26 huomataan myös, ettei listassa näy OpenDaylight-kontrollerin itsensä tekemiä ja kytkimille asettamia vuomerkintöjä. Tähän listaan tulevat ainoastaan käyttäjän omat vuomerkinnät.



Kuvio 26. Flows-välilehti

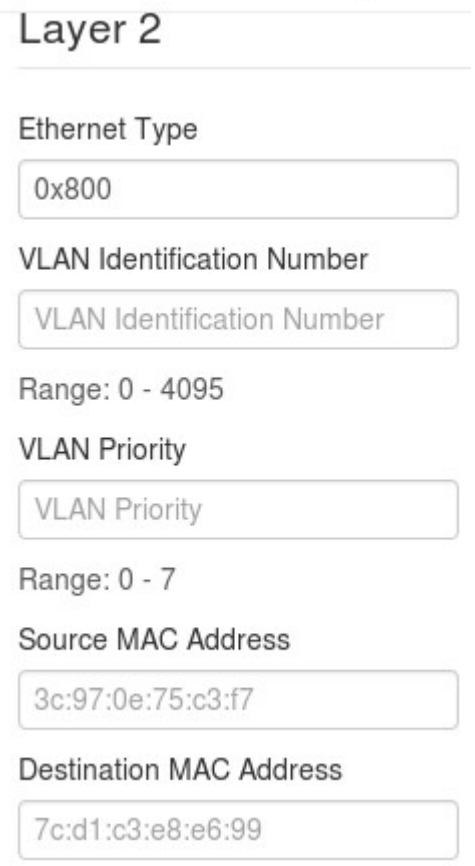
Vuomerkintöjen tekemistä päätettiin testata toteuttamalla yksinkertainen vuomerkintä, joka pudottaa kaikki IP-paketit, jotka vastaanotetaan OF|00:00:00:00:00:00:00:01 portista 1. Kyseiseen porttiin on kytkettynä päätelaite h1s1, jolloin kyseisen päätelaitteelta ei voida onnistuneesti suorittaa *ping*-komentoa muille päätelaitteille.

Painamalla *Add Flow Entry* –painiketta aukeaa valikko, josta vuomerkinnät luodaan. Vuomerkinnät voidaan keskittää hienojakoisesti yleisen OSI-mallin mukaisille verkkotasoille 2-4. Kuviossa 27 on esitettynä aukeavan valikon ensimmäinen osa, johon luotiin *Testi*-niminen vuomerkintä. Himmennetyt tekstit eivät ole asetettuja vaan ovat esimerkkejä mahdollisista asetuksista.



Kuvio 27. Vuomerkinnän lisääminen 1/4

Kuvioissa 28 ja 29 on esitettynä OSI-mallin tason 2 (L2-taso) ja 3 (L3-taso) mahdolliset asetukset, joita ei testaukseen käytettävän vuomerkinnän puitteissa tarvinnut muuttaa, sillä *Ether Type* oli valmiiksi *0x800*, joka tarkoittaa kaikkea IP-liikennettä. Näiden valikoiden kautta olisi mahdollista tehdä vuomerkintöjä esimerkiksi VLAN, MAC-osoitteen tai IP-osoitteen perusteella. Kuviossa 28 nähdään vuomerkintään mahdollisesti asetettavat L2-tason asetukset.



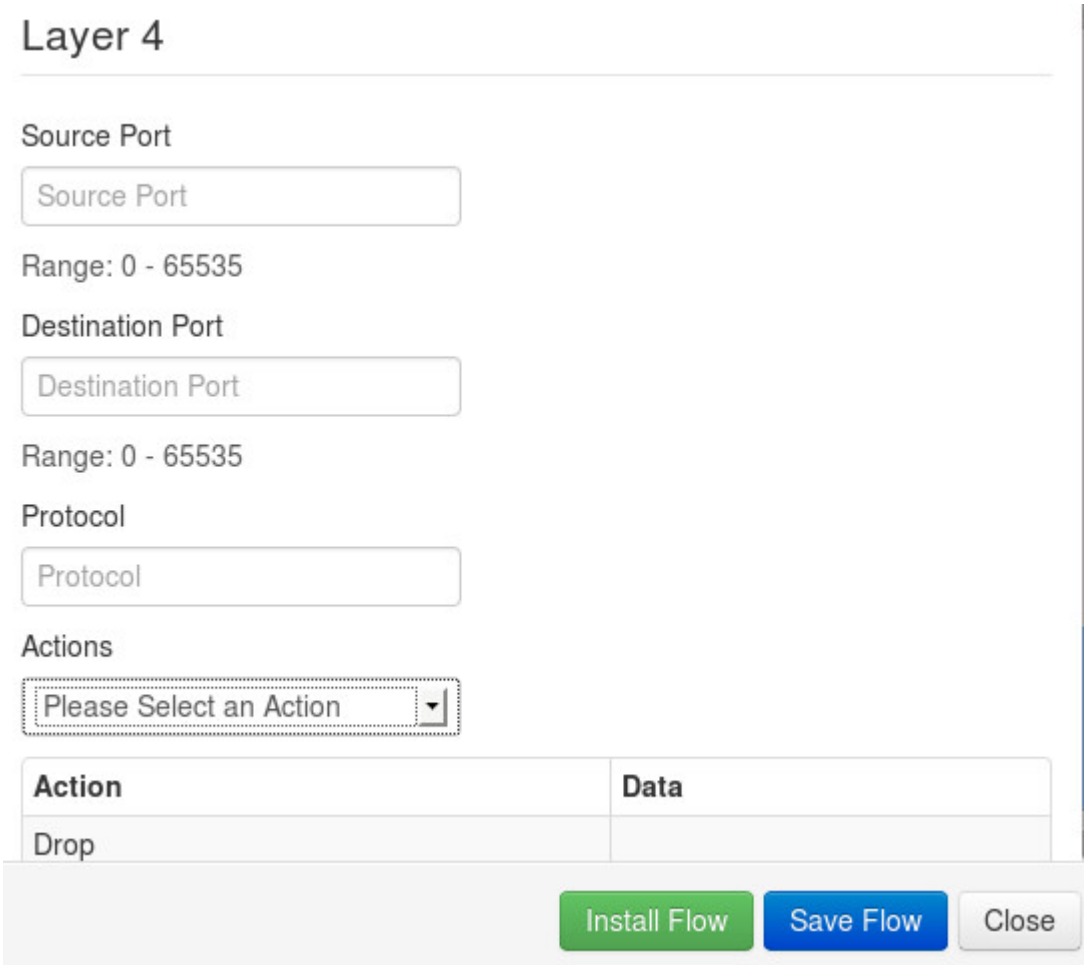
Kuvio 28. Vuomerkinnän lisääminen 2/4

Kuviossa 29 nähdään vuomerkintään mahdollisesti asetettavat L3-tason asetukset.

Kuvio 29. Vuomerkinnän lisääminen 3/4

Viimeisenä voidaan asettaa OSI-mallin tason 4 asetukset ja toiminnot, jotka vuomerkintää vastaaville paketille toteutetaan. Lopuksi vuomerkintä voidaan asettaa suoraan kytkimelle painamalla *Install Flow* –painiketta. *Save*-painikkeella voidaan tallen-

taa luotu vuomerkintä, jolloin sitä ei aseteta kytkimelle. Kuviossa 30 on esitettynä tason 4 asetukset ja valittu *Drop*-toiminto sekä *Install Flow* –painike.

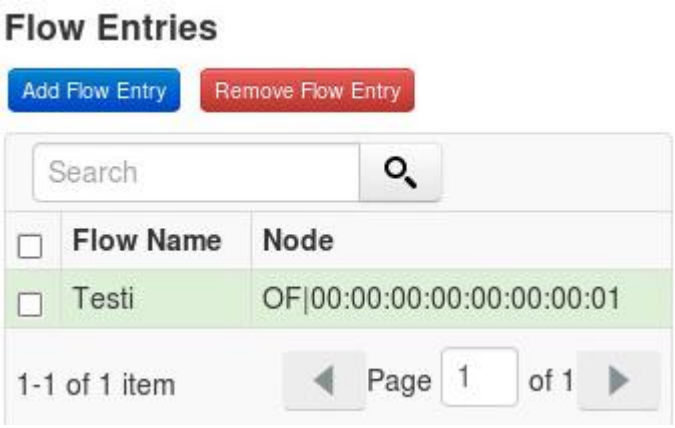


Kuvio 30. Vuomerkinnän lisääminen 4/4

Vuomerkintään voidaan asettaa useita toimintoja. Toiminnot, jotka on mahdollista asettaa *Actions*-valikosta, on esitettynä kuviossa 31.

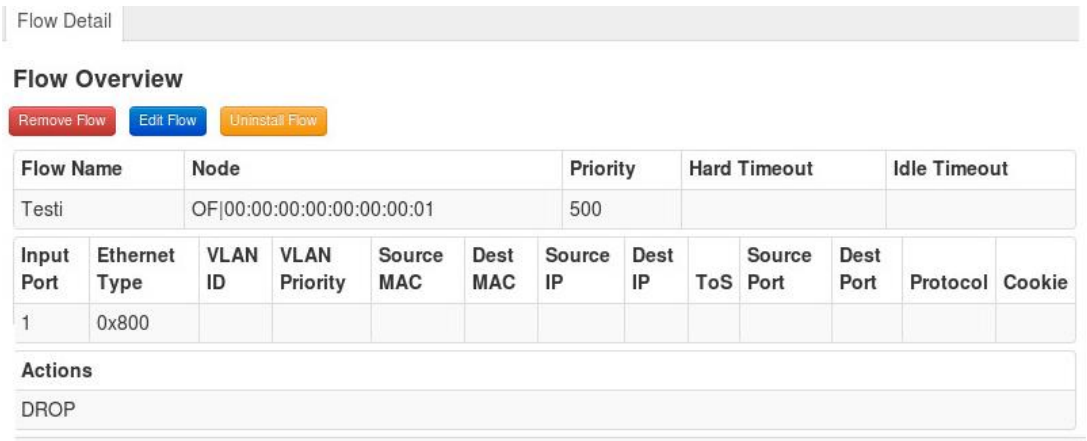
Kuvio 31. Vuomerkinnän mahdolliset toiminnot

Vuomerkinnän asetuksen jälkeen näkyy listassa *Testi*-niminen vuomerkintä. Kuviossa 32 on esitetty vuomerkinnän näkyminen.



Kuvio 32. Vuomerkintä

Painamalla vuomerkinnän nimestä avautuu tarkempi näkymä vuomerkinnästä. Tämä on kätevä ominaisuus vuomerkintöjen monimutkaistuessa. Tämän näkymän kautta voidaan myös poistaa, muokata ja ottaa pois käytöstä vuomerkintöjä. Kuviossa 33 on esitettynä *Testi*-vuomerkinnän tarkempi näkymä.



Kuvio 33. Vuomerkintä tarkemmin

Nyt vuomerkinnän toimintaa voidaan todentaa Mininet-VM -laittella seuraavalla komennolla.

```
mininet>pingall
```

Tuloksena on kuvion 34 mukainen tilanne, jossa päätelaite h1s1 ei saavuta muita päätelaitteita, mutta muut päätelaitteet kykenevät liikennöimään keskenään.

```
mininet> pingall
*** Ping: testing ping reachability
h1s1 -> X X X
h1s2 -> X h2s1 h2s2
h2s1 -> X h1s2 h2s2
h2s2 -> X h1s2 h2s1
*** Results: 50% dropped (6/12 received)
mininet> _
```

Kuvio 34. Vuomerkinnällä IP-liikenteen esto

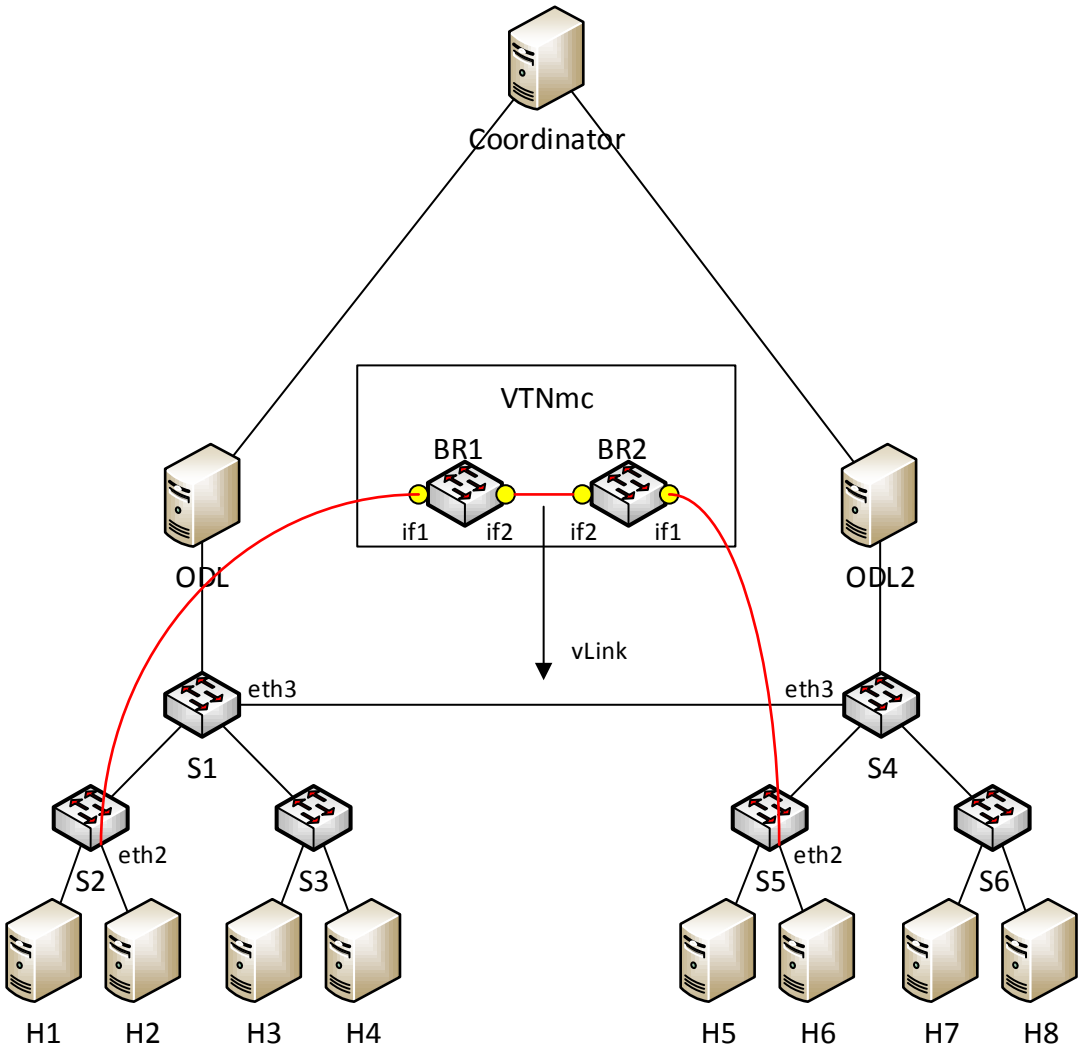
Vuomerkinnän toiminnasta voitiin varmistua ottamalla vuomerkintä pois käytöstä painamalla kuviossa 33 näkyvää *Uninstall Flow* –painiketta. Tällöin liikennöinti päätelaitteiden välillä tapahtui normaalisti, kuten kuviossa 35 on esitetty.

```
mininet> pingall
*** Ping: testing ping reachability
h1s1 -> h1s2 h2s1 h2s2
h1s2 -> h1s1 h2s1 h2s2
h2s1 -> h1s1 h1s2 h2s2
h2s2 -> h1s1 h1s2 h2s1
*** Results: 0% dropped (12/12 received)
mininet> _
```

Kuvio 35. Vuomerkintä poistettu

6.2.2 VTN-tekniikan todennus

OpenDaylight-ympäristössä todennettiin lisäksi VTN-verkkoja. Tämä tapahtui muodostamalla VTN-verkko, joka ulottuu kahden OpenDaylight-kontrollerin alueelle. Tätä toteutusta varten käytetään kaikkia OpenDaylight-ympäristön laitteita. OpenDaylight-kontrolleri vaihdetaan tukemaan VTN-tekniikkaa liitteen 2 mukaisesti. Kuvissa 36 on esitettynä toteutettava ympäristö, jossa päätelaitteet H2 ja H6 kykenevät keskustelemaan keskenään, mutta muilla päätelaitteilla ei ole pääsyä mihinkään. Todentaminen tapahtuu muodostamalla VTN-verkko päätelaitteiden H2 ja H6 välille. Koordinaattorin ohjatessa OpenDaylight-kontrollereilta, ei kytkimille muodosteta automaattisesti vuomerkintöjä, joten päätteillä ei ole yhteyksiä toisiinsa.



Kuvio 36. VTNmc-topologia

Toteutus aloitetaan luomalla Mininet-VM –laitteelle python-skripti. Kyseisellä skriptillä muodostetaan kuvion 36 mukaisesti kytkimet S1-S6 ja päätelaitteet H1-H4 sekä

linkit niiden ja kontrollerien välille. Kytkimet S1-S3 asetetaan ODL-laittella olevan kontrolelrin hallintaan ja kytkimet S4-S6 asetetaan ODL2-laitteella olevan kontrollerin hallintaan. Skriptin luominen ja avaaminen tapahtuvat seuraavasti Mininet-VM –laitteella.

```
mininet@mininet-vm:~$ sudo touch multitree.py
mininet@mininet-vm:~$ sudo chmod +x multitree.py
mininet@mininet-vm:~$ sudo nano multitree.py
```

Tässä vaiheessa avautuvaan tiedostoon lisätään liitteen 4 mukainen skripti. Nyt verkko on mahdollista käynnistää käyttämällä luotua skriptiä. Ympäristön käynnistys aloitetaan koordinaattorista, joka käynnistetään liitteen 2 mukaisesti seuraavalla komennolla.

```
[root@coordinator ~]#/usr/local/vtn/sbin/vtn_start
```

Seuraavaksi käynnistetään OpenDaylight-kontrollerin VTN-tekniikkaa tukeva versio sekä *ODL*- että *ODL2*-laitteella seuraavilla komennoilla.

```
opendaylight@ODL:~$ cd /vtn/manager/dist/target/distribution.vtn-manager-0.2.0-
SNAPSHOT-osgipackage/opendaylight
opendaylight@ODL:~$./run.sh
```

Viimeisenä käynnistetään ohjattava verkko Mininet-VM –laitteella ja testataan etteivät päätelaitteet kykene keskustelemaan toistensa kanssa suorittamalla *pingall*-komento. Tämä tapahtuu seuraavasti Mininet-VM –laitteella.

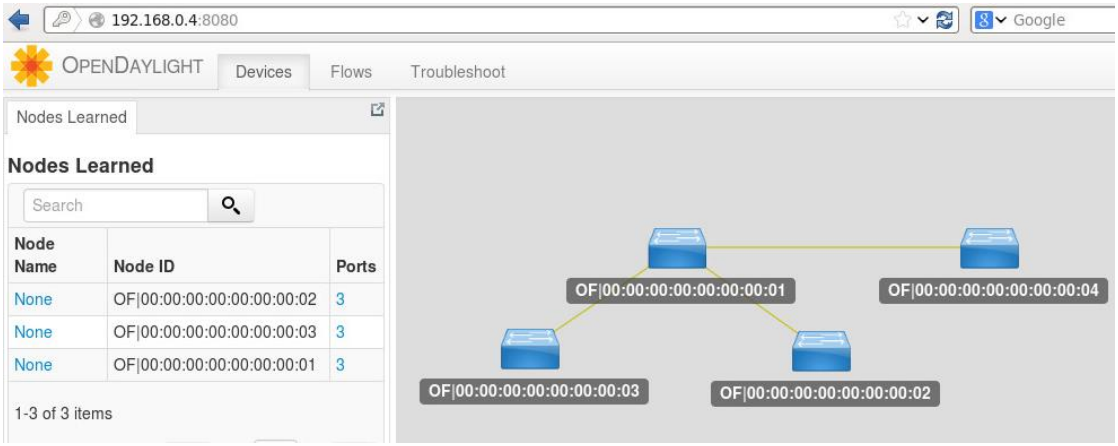
```
mininet@mininet-vm:~$ sudo ./multitree.py
mininet>pingall
```

Kuviossa 37 on esitettynä, kuinka päätelaitteet eivät kykene keskustelemaan toistensa kanssa.

```
mininet> pingall
*** Ping: testing ping reachability
h1 -> X X X X X X X
h2 -> X X X X X X X
h3 -> X X X X X X X
h4 -> X X X X X X X
h5 -> X X X X X X X
h6 -> X X X X X X X
h7 -> X X X X X X X
h8 -> X X X X X X X
*** Results: 100% dropped (0/56 received)
mininet>
```

Kuvio 37. VTNmc ilman konfiguraatiota

Centos-laitteella voidaan tarkastaa, että OpenDaylight-kontrollerit ovat havainneet niille osoitetut kytkimen menemällä niiden hallintasivulle. Kuviossa 38 on esitettynä ODL-laitteen OpenDaylight-kontrollerin näkymä verkosta. Kuviosta 38 voidaan havaita myös, että kontrolleri on havainnut neljännen laitteen, muttei hallinnoi sitä, sillä *Nodes Learned* –listalla on näkyvissä ainoastaan kolme verkkolaitetta.



Kuvio 38. ODL-laitteen näkymä VTNmc-verkosta

Varsinainen VTN-verkko muodostetaan antamalla komennot koordinaattorille HTTP-pyyntöinä koordinaatorin REST API –rajapintaan. Pyynnöt voidaan antaa suoraan Coordinator-laitteen komentorivillä. Ensimmäisenä luodaan VTN, jolle annetaan nimeksi *vtnmc*. Tämä tapahtuu seuraavalla komennolla.

```
[root@coordinator ~]# curl -v --user admin:adminpass -H 'content-type: application/json' -H 'ipaddr:127.0.0.1' -X POST -d '{"vtn" : {"vtn_name":"vtnmc"}}' http://127.0.0.1:8083/vtn-webapi/vtns.json
```

Edellisessä komennossa asetetaan pyyntö *verbose* muotoon optiolla –v, jolloin voidaan nähdä koordinaattorin vastaus pyyntöön. Tällöin voidaan varmistaa pyyntöjen toiminta. Edellisessä komennossa asetetaan käyttäjäksi *admin* ja salasanaksi *adminpass*. Lisäksi komento on *POST*-muotoinen, eli sillä asetetaan tietoa kohteeseen. Seuraavaksi annetaan koordinaattorille tieto OpenDaylight-kontrollereista seuraavilla komennoilla.

```
[root@coordinator ~]# curl -v --user admin:adminpass -H 'content-type: application/json' -H 'ipaddr:127.0.0.1' -X POST -d '{"controller": {"controller_id": "odc1", "ipaddr":"192.168.0.4", "type": "odc", "version": "1.0", "auditstatus":"enable"}}' http://127.0.0.1:8083/vtn-webapi/controllers.json
[root@coordinator ~]# curl -v --user admin:adminpass -H 'content-type: application/json' -H 'ipaddr:127.0.0.1' -X POST -d '{"controller": {"controller_id":
```

```
"odc2", "ipaddr":"192.168.0.5", "type": "odc", "version": "1.0", "auditsta-
tus":"enable"}}' http://127.0.0.1:8083/vtn-webapi/controllers.json
```

Ensimmäisessä komennossa asetetaan kontrollerin tunnus, nimi, IP-osoite, tyyppi ja kontrollerin versio. Tässä tapauksessa annettiin ODL-laitteen OpenDaylight-kontrollerin tunnukseksi *odc1*, osoitteeksi *182.168.0.4*, tyypiksi *odc* ja versioksi *1.0*. Sama toistettiin ODL2-laitteen OpenDaylight-kontrollerille.

Tämän jälkeen luodaan vBridge BR1 ja BR2. BR1 saa nimekseen *vbr1* ja BR2 *vbr2*. BR1 asetetaan ODL-laitteen Opendaylight-kontrollerille ja BR2 ODL2-laitteen Open-Daylight-kontrollerille. Nämä tapahtuvat seuraavilla komennoilla.

```
[root@coordinator ~]# curl -v --user admin:adminpass -H 'content-type: applica-
tion/json' -H 'ipaddr:127.0.0.1' -X POST -d '{"vbridge" :
{"vbr_name":"vbr1","controller_id":"odc1","domain_id":"(DEFAULT)" }}'
http://127.0.0.1:8083/vtn-webapi/vtns/vtnmc/vbridges.json
[root@coordinator ~]# curl -v --user admin:adminpass -H 'content-type: applica-
tion/json' -H 'ipaddr:127.0.0.1' -X POST -d '{"vbridge" :
{"vbr_name":"vbr2","controller_id":"odc2","domain_id":"(DEFAULT)" }}'
http://127.0.0.1:8083/vtn-webapi/vtns/vtnmc/vbridges.json
```

Tämän jälkeen luodaan molemmille vBridgeille kaksi virtuaalista rajapintaa *if1* ja *if2* seuraavilla komennoilla.

```
[root@coordinator ~]# curl -v --user admin:adminpass -H 'content-type:
application/json' -H 'ipaddr:127.0.0.1' -X POST -d '{"interface":
{"if_name": "if1"}}' http://127.0.0.1:8083/vtn-
webapi/vtns/vtnmc/vbridges/vbr1/interfaces.json
[root@coordinator ~]# curl -v --user admin:adminpass -H 'content-type:
application/json' -H 'ipaddr:127.0.0.1' -X POST -d '{"interface":
{"if_name": "if2"}}' http://127.0.0.1:8083/vtn-
webapi/vtns/vtnmc/vbridges/vbr1/interfaces.json
[root@coordinator ~]# curl -v --user admin:adminpass -H 'content-type:
application/json' -H 'ipaddr:127.0.0.1' -X POST -d '{"interface":
{"if_name": "if1"}}' http://127.0.0.1:8083/vtn-
webapi/vtns/vtnmc/vbridges/vbr2/interfaces.json
[root@coordinator ~]# curl -v --user admin:adminpass -H 'content-type:
application/json' -H 'ipaddr:127.0.0.1' -X POST -d '{"interface":
{"if_name": "if2"}}' http://127.0.0.1:8083/vtn-
webapi/vtns/vtnmc/vbridges/vbr2/interfaces.json
```

Seuraavaksi muodostetaan reunakohta kahden kontrollerin hallinnoiman verkon välille. Reunakohdalle annetaan nimeksi *b1* ja siihen sidotaan loogisina portteina kytkimen S1 portti eth3 ja kytkimen S4 portti eth3. Tätä varten tarvitaan kytkinten MAC-

osoitteet, jotka voi katsoa esimerkiksi OpenDaylight-kontrollerin hallintasivulta. Reunakohdan luominen tapahtuu seuraavalla komennolla.

```
[root@coordinator ~]# curl -v --user admin:adminpass -H 'content-type: application/json' -H 'ipaddr:127.0.0.1' -X POST -d '{"boundary": {"boundary_id": "b1", "link": {"controller1_id": "odc1", "domain1_id": "(DEFAULT)", "logical_port1_id": "PP-OF:00:00:00:00:00:00:00:01-s1-eth3", "controller2_id": "odc2", "domain2_id": "(DEFAULT)", "logical_port2_id": "PP-OF:00:00:00:00:00:00:00:04-s4-eth3"}}}' http://127.0.0.1:8083/vtn-webapi/boundaries.json
```

Reunakohdan ollessa luotuna, voidaan siihen liittää virtuaalinen linkki eli vLink. Seuraavalla komennolla lisätään vLink nimeltä *vlink1*, joka sidotaan vBridge BR1 virtuaaliseen rajapintaan *if2* ja vBridge BR2 virtuaaliseen rajapintaan *if2*. Lisäksi asetetaan, ettei käytössä ole VLAN-tunnusta.

```
[root@coordinator ~]# curl -v --user admin:adminpass -H 'content-type: application/json' -H 'ipaddr:127.0.0.1' -X POST -d '{"vlink": {"vlk_name": "vlink1" , "vnode1_name": "vbr1", "if1_name":"if2", "vnode2_name": "vbr2", "if2_name": "if2", "boundary_map": {"boundary_id":"b1","no_vlan_id": "true"}}}' http://127.0.0.1:8083/vtn-webapi/vtns/vtnmc/vlinks.json
```

Tämän jälkeen kartoitetaan kytkimen S2 portti eth2 kuuluvaksi vBridge BR1 porttiin *if1* sekä kytkimen S5 portti eth2 kuuluvaksi vBridge BR2 porttiin *if1*. Nämä tapahtuvat seuraavilla komennoilla.

```
curl -v --user admin:adminpass -H 'content-type: application/json' -H 'ipaddr:127.0.0.1' -X PUT -d '{"portmap":{"logical_port_id": "PP-OF:00:00:00:00:00:00:00:02-s2-eth2"}}' http://127.0.0.1:8083/vtn-webapi/vtns/vtnmc/vbridges/vbr1/interfaces/if1/portmap.json
curl -v --user admin:adminpass -H 'content-type: application/json' -H 'ipaddr:127.0.0.1' -X PUT -d '{"portmap":{"logical_port_id": "PP-OF:00:00:00:00:00:00:00:05-s5-eth2"}}' http://127.0.0.1:8083/vtn-webapi/vtns/vtnmc/vbridges/vbr2/interfaces/if1/portmap.json
```

VTN-verkko on nyt toiminnassa. Tämä voidaan havaita *ODL*-laitteen OpenDaylight-kontrollerilta kuvion 39 mukaisesti.

```
2015-04-14 01:05:22.675 EEST [VTN Task Thread: default] INFO  o.o.v.m.internal.VTNManagerImpl - defa
ult:vtnmc.vbr1.if1: Port mapping changed: PortMap[config=PortMapConfig[node=OF|00:00:00:00:00:00:00:
02,port=SwitchPort[name=s2-eth2],vlan=0],connector=OF|2@OF|00:00:00:00:00:00:00:02]
2015-04-14 01:05:22.679 EEST [VTN Task Thread: default] INFO  o.o.v.m.internal.VTNManagerImpl - defa
ult:vtnmc.vbr1.if1: Bridge interface changed: VInterface[name=if1,enabled,state=UP,entityState=UP]
2015-04-14 01:05:22.702 EEST [VTN Task Thread: default] INFO  o.o.v.m.internal.VTNManagerImpl - defa
ult:vtnmc.vbr1.if2: Port mapping changed: PortMap[config=PortMapConfig[node=OF|00:00:00:00:00:00:00:
01,port=SwitchPort[name=s1-eth3],vlan=0],connector=OF|3@OF|00:00:00:00:00:00:00:01]
2015-04-14 01:05:22.712 EEST [VTN Task Thread: default] INFO  o.o.v.m.internal.VTNManagerImpl - defa
ult:vtnmc.vbr1.if2: Bridge interface changed: VInterface[name=if2,enabled,state=UP,entityState=UP]
2015-04-14 01:05:22.715 EEST [VTN Task Thread: default] INFO  o.o.v.m.internal.VTNManagerImpl - defa
ult:vtnmc.vbr1: Bridge changed: VBridge[name=vbr1,ageInterval=600,faults=0,state=UP]
```

Kuvio 39. VTN-verkon muodostuminen OpenDaylight-kontrollerilla

Lisäksi toiminta testataan *ping*-komennolla. Päätteeltä H2 suoritetaan *ping*-komento päätteelle H6. Kuviossa 40 on osoitettuna kuinka päätelaitteelta H2 on yhteys päätelaitteelle H6. Lisäksi kuviosta 40 käy ilmi, että yhteydet muilla päätelaitteilla ei ole toiminnassa, mikä testattiin *ping*-komennolla päätelaitteiden H1 ja H3 välillä.

```
mininet> h2 ping h6
PING 10.0.0.6 (10.0.0.6) 56(84) bytes of data.
64 bytes from 10.0.0.6: icmp_seq=1 ttl=64 time=94.9 ms
64 bytes from 10.0.0.6: icmp_seq=2 ttl=64 time=4.70 ms
64 bytes from 10.0.0.6: icmp_seq=3 ttl=64 time=5.82 ms
64 bytes from 10.0.0.6: icmp_seq=4 ttl=64 time=14.6 ms
^C
--- 10.0.0.6 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 4.704/30.012/94.922/37.671 ms
mininet> h1 ping h3
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.
From 10.0.0.1 icmp_seq=1 Destination Host Unreachable
From 10.0.0.1 icmp_seq=2 Destination Host Unreachable
From 10.0.0.1 icmp_seq=3 Destination Host Unreachable
^C
--- 10.0.0.3 ping statistics ---
5 packets transmitted, 0 received, +3 errors, 100% packet loss, time 4000ms
pipe 4
mininet>
```

Kuvio 40. Toimiva VTN-yhteys

6.2.3 OpenContrail-ympäristö

OpenContrail-ympäristön todennuksessa luotiin OpenContrail-palvelin, jolla kartoitettiin OpenContrail-järjestelmän ominaisuuksia ja todennettiin niitä luomalla testiverkko. Lisäksi järjestelmään kuuluvasta OpenStack-pilvipalvelusta kartoitettiin ja testattiin perustoiminnot luomalla testattava virtuaalitietokone, testiverkko ja testireititin.

OpenContrail-järjestelmä on tarkoitettu hajautettavaksi vähintään viidelle tai useammalle palvelimelle, mutta järjestelmä tarjoaa myös yhden palvelimen testiratkai-

sun. Tässä kaikki järjestelmän osat asennetaan yhdelle palvelimelle. Tätä kyseistä testiratkaisua käytettiin tämän opinnäytetyön OpenContrail-ympäristössä.

Taulukossa 4 on esitettynä OpenContrail-ympäristön virtualisoitujen palvelimet ja tietokoneet sekä niiden käyttötarkoitus. Taulukossa 4 on esitettynä IP-osoitteet sisäisessä verkossa, jossa laitteet kommunikoivat keskenään. Laitteilla on kuitenkin oltava pääsy Internettiin tai niille on kyettävä siirtämään asennuspaketteja muuta kautta, sillä niille joudutaan asentamaan paketteja ja lataamaan tiedostoja. Tämä voidaan toteuttaa toisella verkkorajapinnalla, toteuttaa pääsy Internettiin sisäisen verkon kautta tai siirtämällä tarvittavat tiedostot toiselta tietokoneelta esimerkiksi käyttäen SCP:tä.

Taulukko 4. OpenContrail-järjestelmän laitteet

Laite	Käyttöjärjestelmä	Hostname	Käyttötarkoitus	IP-osoite ja verkko
CentOS-GUI	CentOS 7 minimal	centos-gui	OpenContrail-järjestelmän graafisen käyttöliittymän tarkasteluun	192.168.0.2/24
OpenContrail	CentOS 6.5 minimal	contrail	OpenContrail-järjestelmän palvelin	192.168.0.3/24

OpenContrail-järjestelmälle ilmoitetaan minimi vaatimuksiksi neljä (4) prosessoriydintä, 32Gb keskusmuistia ja 128Gb levytilaa. Kyseiset vaatimukset ovat kuitenkin suuret ja siksi kokeilu tehtiin huomattavasti alemmilla resursseilla. OpenContrail-palvelimelle asetettiin yksi prosessoriydin, 8Gb keskusmuistia ja 32Gb levytilaa, sillä tarkoituksena oli todentaa ainoastaan järjestelmän toimivuutta minimitasolla, eikä luoda täysimittaista pilvipalvelua. CentOS-GUI -laitteelle asetettiin yksi prosessoriydin, 2Gb keskusmuistia ja 8Gb levytilaa. Lisäksi kaikista laitteista on kytketty pois päältä tietoliikennettä mahdollisesti estävät palvelut, kuten palomuurit.

Ympäristö aloitetaan asentamalla CentOS-GUI laitteelle graafinen käyttöliittymä. Seuraavilla komennoilla asennetaan graafinen käyttöliittymä ja järjestelmänvalvojan työkalut ja *ifconfig*-komentoihin tarvittava *net-tools*.

```
[root@centos-gui ~] #yum install net-tools
[root@centos-gui ~] #yum groupinstall "GNOME Desktop" "Graph-
ical Administration Tools"
```

Tämän jälkeen on mahdollista käynnistää graafinen käyttöliittymä uudelleenkäynnistyksen jälkeen seuraavalla komennolla.

```
[root@centos-gui ~] #startx
```

Seuraavaksi asennetaan OpenContrail-laitteella varsinainen OpenContrail-järjestelmä. Tässä työssä asennuspaketit haettiin Juniper Networksin sivustolta. Lataukset löytyvät osoitteesta https://www.juniper.net/support/downloads/?p=contrail. OpenContrail-järjestelmästä otettiin käyttöön versio 1.10. Tämä versio valittiin siksi, että todennuksessa kokeiltiin useita versioita, useilla käyttöjärjestelmillä, mutta vain tämä kyseinen järjestelmä asentui kunnolla. Ladattu asennuspaketti on nimeltään *contrail-install-packages-1.10-32~centos65havana.el6.noarch.rpm*, joka sijaitsee root-käyttäjän kotikansiossa ja se asennetaan seuraavalla komennolla.

```
[root@contrail ~]#yum localinstall contrail-install-packages-
1.10-32~centos65havana.el6.noarch.rpm
```

Tämän jälkeen ajetaan skripti, joka luo tarvittavat *OpenContrail* -paketit ja verkonluontiin tarvittavat apuohjelmat. Tämä tapahtuu seuraavalla komennolla

```
[root@contrail ~]# cd /opt/contrail/contrail_packages
[root@contrail contrail_packages]# ./setup.sh
```

Seuraavaksi muokataan OpenContrail-järjestelmälle tiedot luotavasta ympäristöstä. Tämä tapahtuu kopioimalla valmis skripti ja muokkaamalla se omaa ympäristöä vastaavaksi. Järjestelmä tarjoaa oletuksena sekä useammalle palvelimelle että yhdelle palvelimelle tarkoitetun skriptin. Seuraavilla komennoilla kopioidaan yhdelle palvelimelle tarkoitettu skripti, nimetään se oikeanlaiseksi järjestelmää varten ja avataan kyseinen skripti muokkausta varten.

```
[root@contrail ~]# cd /opt/contrail/utils/fabfile/testbeds
[root@contrail testbeds]# cp testbed_singlebox_example.py
testbed.py
[root@contrail ~]#vi testbed.py
```

Avautuvasta skriptistä muokataan osia vastaamaan seuraavaa.

```
#Management ip addresses of hosts in the cluster
host1 =’root@192.168.0.3’
#Host form which the fab command are triggered to install and
provision
host_build = ‘root@192.168.0.3’
#Hostnames
env.hostnames ={
        ‘all’: [‘contrail’]
}
```

```
env.password = '<root-käyttäjän salasana>'
#Passwords of each hos
env.passwords = {
        host1: '<root-käyttäjän salasana>'
        host_build: ' '<root-käyttäjän salasana>',
}
#For reimage purpose
env.ostypes = {
        host1:'centos' ,
}
```

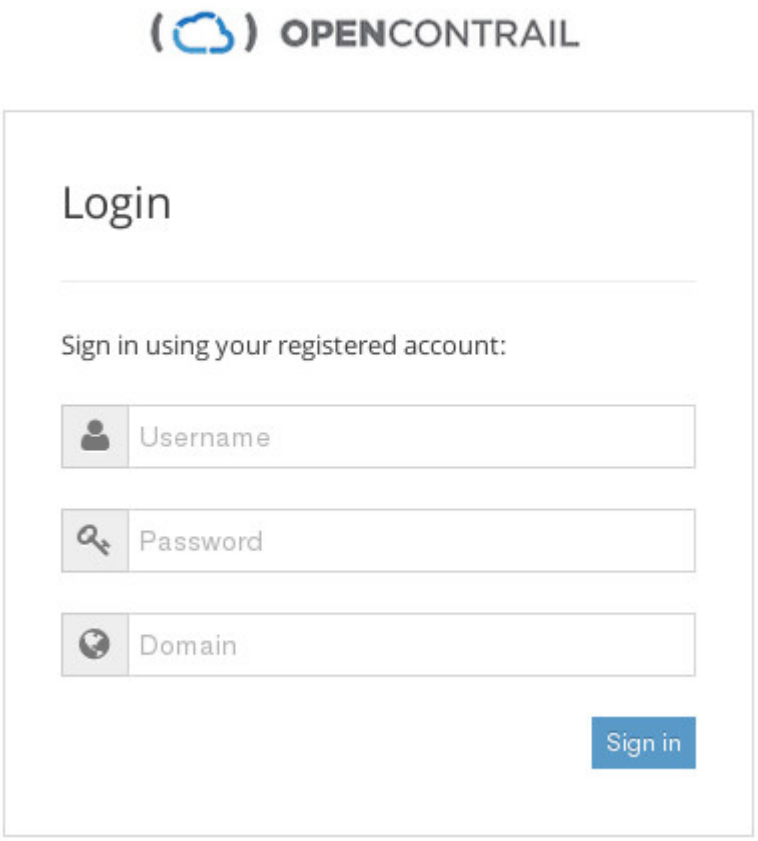
Tämän jälkeen voidaan suorittaa varsinainen OpenContrail-järjestelmän asennus seuraavasti.

```
[root@contrail testbeds]# cd /opt/contrail/utils
[root@contrail utils]#fab install_contrail
```

Tämän jälkeen laite käynnistyy uudelleen. Verkkorajapinnat ovat tässä vaiheessa nimettynä uudelleen ja IP-osoite on annettava oikealle verkkorajapinnalle uudestaan ennen asennuksen jatkamista. IP-osoitteen antaminen ja asennuksen jatkaminen tapahtuu seuraavasti.

```
[root@contrail ~]# ifconfig p2p0p0 192.168.0.3 netmask
255.255.255.0
[root@contrail ~]# cd /opt/contrail/utils
[root@contrail utils]# fab setup_all
```

Järjestelmä jatkaa käynnistymällä uudelleen, jonka jälkeen OpenContrail-järjestelmä on asennettu. Asennuksen ja järjestelmän toimintaa voidaan todentaa käyttämällä CentOS-GUI –laitetta. Kyseisellä laitteella käynnistetään graafinen käyttöliittymä ja kirjaudutaan sisään. Tämän jälkeen avataan selain ja asetetaan osoitekenttään OpenContrail-järjestelmän osoite ja portti 8080. Tässä tapauksessa osoiteriville kirjoitetaan *192.168.0.3:8080*. Avautuva sivu tulisi olla kuvion 41 mukainen, josta voidaan kirjautua sisään käyttämällä järjestelmänvalvojan tunnuksia tai demo-käyttäjän tunnuksia. Järjestelmänvalvojan käyttäjätunnus on oletuksena *admin* ja salasana *secret123*. Demo-käyttäjän oletustunnukset ovat *demo* ja *secret123*. *Domain*-kohta kirjautumisessa voidaan molemmilla käyttäjillä jättää tyhjäksi, jolloin käytetään oletustoimialuetta (Default-domain).



Kuvio 41. OpenContrail-kirjautumisruutu

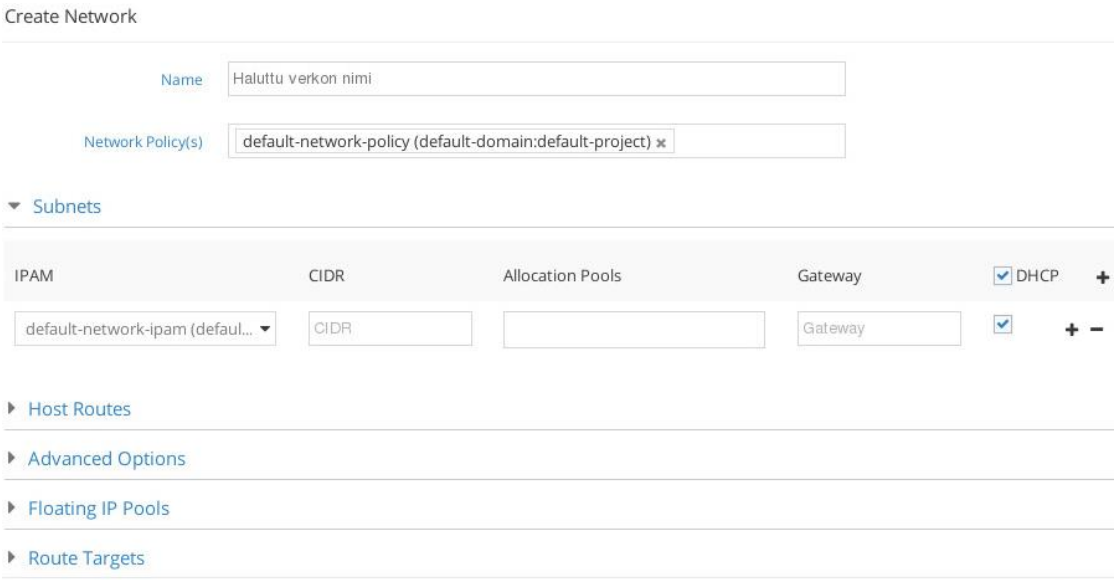
Järjestelmänvalvojan tunnuksilla kirjautumisen jälkeen käyttäjälle avautuu OpenContrai-järjestelmän hallinnan käyttöliittymä. Käyttöliittymän kautta on mahdollista tarkastella järjestelmän yleistilaa, sen laitteita ja toiminnallisia osia. Liitteessä 5 on esitettynä pääikkuna, jonka kautta voidaan valita ylhäältä monitorointi tai konfigurointi. Tässä tapauksessa valittuna on *Monitor*-välilehti, jonka kautta voidaan valvoa järjestelmän osien tilaa ja liikennemääriä.

Monitoroinnista voidaan valita *Networking*-valikko, josta voidaan tarkastella tarkemmin verkon tilaa ja liikennemääriä. Samaisen valikon alta voidaan valita *Networks*-välilehti, jonka alta voidaan tarkastella luotuja verkkoja. Liitteessä 6 on kuvattuna kyseinen välilehti, josta voidaan havaita luodut verkot. Tässä tapauksessa ovat *Contrail*, *Contrail-dash-testi* ja *Testaus*, jotka ovat luotu oletustoimialueelle.

Configure-välilehden alta voidaan tarkastella järjestelmän ja verkon fyysistä rakennetta. *Configure*-välilehden alta löytyy neljä välilehteä, joiden kautta voidaan hallita verkon rakennetta (Infrastructure), virtuaalisia verkkoja (Networking), palveluita

(Services) ja DNS-palvelimia (DNS). Oletuksena avautuvan *Infrastructure*-välilehden alta voidaan hallita verkon sisäisiä välitystoimintoja Forwarding Options –välilehden kautta. *BGP-peers* -välilehden alta voidaan lisätä fyysisiä BGP-naapureita. Liitteessä 7 on esitettynä edellä mainitut valikot ja *BGP-peers* -valikko.

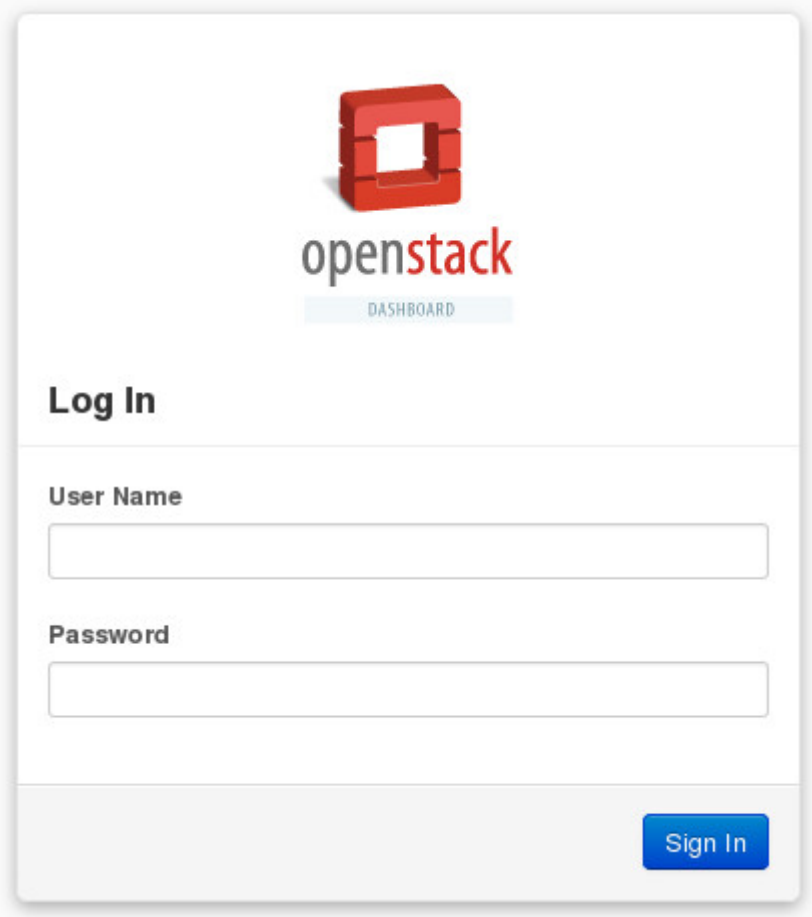
Konfiguraatiopuolella Networking-välilehden alta voidaan hallita verkkoja kokonaisvaltaisesti. Tätä kautta voidaan luoda ja muokata verkkoja (Networks), aliverkkoja, politiikkoja (Policies), IP-osoitteita (IP Address Management ja Manage Floating IPs) sekä projektikohtaisia resursseja (Project Quotas), kuten verkkojen määriä tai reititimien määriä. Liitteessä 8 on kuvattuna *Networks*-välilehti, jonka kautta on luotu *Contrail-dash-testi* ja *Contrail* verkot. Liitteessä 8 on esitettynä myös valikot, joiden kautta voidaan hallita verkkoja. Kuviossa 42 on esitettynä valikko, joka voidaan avata painamalla *plus*-merkkiä liitteen 8 mukaisen käyttöliittymän oikeasta ylälaidasta. Tämä valikon kautta voidaan lisätä verkkoja ja aliverkkoja sekä niiden IP-osoitteita. Todennusta varten tätä kautta luotiin *Contrail-dash-testi* –verkko, joka oli havaittavissa jo aiemmin monitoroinnin puolen *Networks*-välilehden kautta.



Kuvio 42. Verkon luonnin valikot

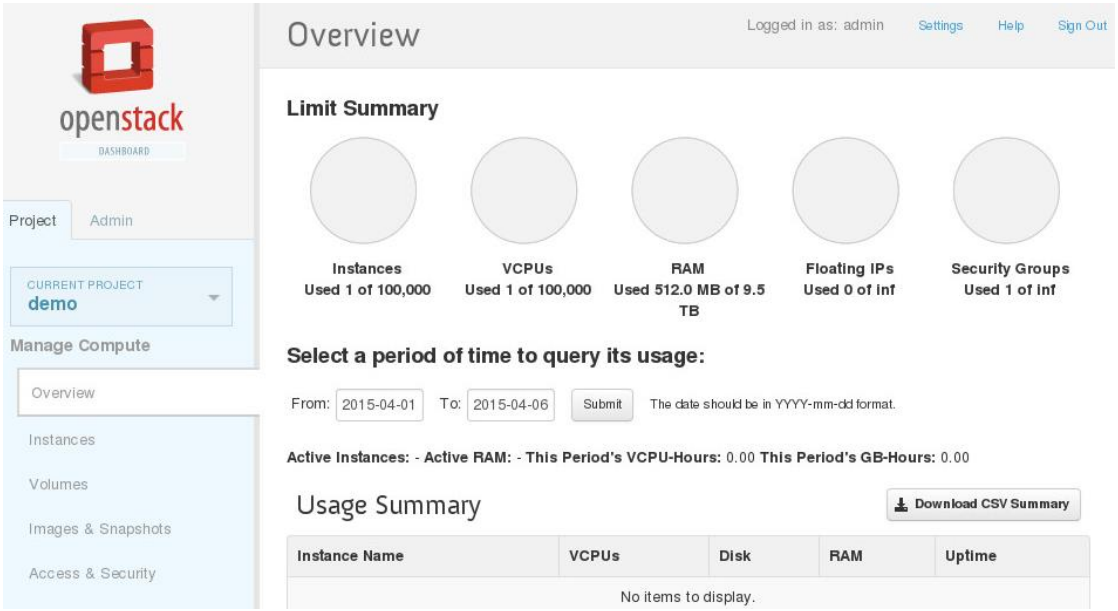
OpenContrail-ympäristön hallinta on tarkoitettu oikeissa ympäristöissä operaattorin hallintapaneeliksi. OpenContrailiin on oletuksena integroitu OpenStack pilvipalvelu, joka on tarkoitettu asiakaskäyttöön. OpenStack käyttää OpenContrailin kautta luotuja resursseja. OpenStack mahdollistaa asiakkaan luoda omia virtuaalisia palvelimia ja palveluita, jotka sijoitetaan OpenContrail-järjestelmän virtuaalisiin verkkoihin. Tässä

toteutuksessa voidaan todentaa OpenStack-pilvipalvelun toiminta avaamalla yhteys sen graafiseen käyttöliittymään, Horizoniin (dashboard). Käyttöliittymään saa avattua *CentOS-GUI* -laitteen web-selaimella kirjoittamalla osoitekenttään *192.168.0.3/dashboard*. Kuviossa 43 on esitettynä OpenStack-kirjautumisruutu.



Kuvio 43. OpenStack-kirjautumisruutu

Kirjautuminen tapahtuu käyttäen samoja oletustunnuksia kuin OpenContrail-hallintapaneelissa. Kirjautumisen jälkeen avautuu OpenStack-järjestelmän yleisnäkymä. Kuviossa 44 on esitettynä OpenStack-järjestelmän yleisnäkymä.



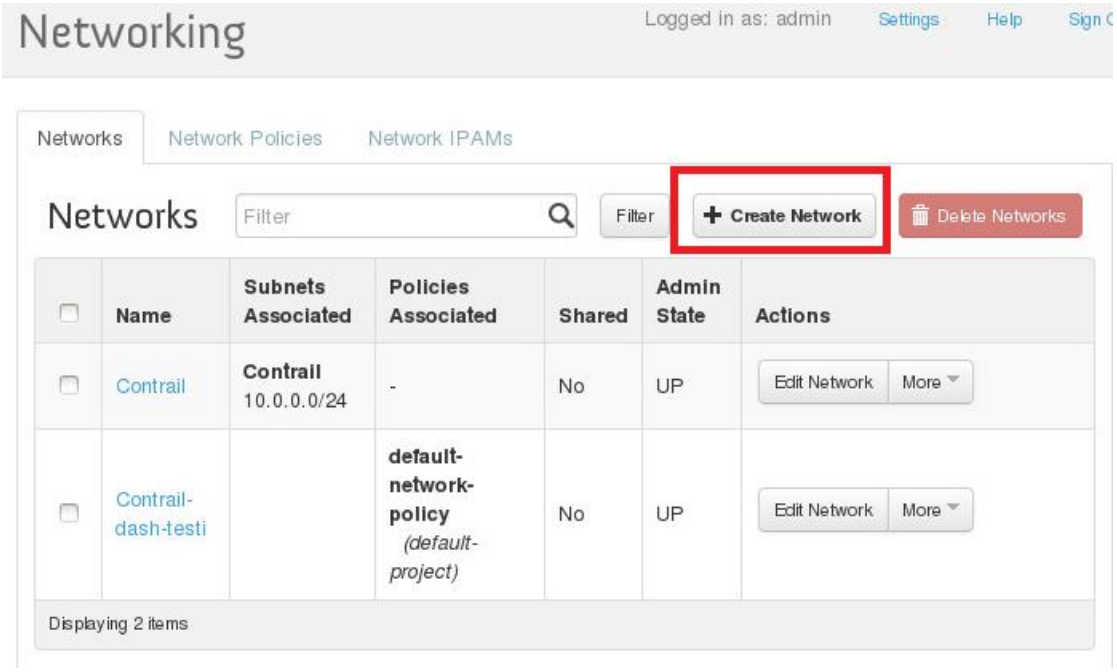
Kuvio 44. OpenStack-järjestelmän yleisnäkymä

OpenStack-käyttöliittymän kautta asiakas voi hallita omia projektejaan ja järjestelmänvalvoja (Admin) voi hallita kaikkia projekteja. Kuviossa 45 on esitettynä valikko, jonka kautta asiakas, tässä tapauksessa *demo*-käyttäjä, voi valita hallintavälilehdet. Kuviossa 45 on käytettynä demo-käyttäjän projektia. Käyttäjä voi katsoa yleisnäkymän (Overview), hallita virtualisoituja tietokoneita (Instances), luoda virtualisoiduille tietokoneille levytiloja (Volumes), luoda ja muokata levykuvia (Images & Snapshots), luoda tietoturvapolitiikkoja (Access & Security), hallita verkkoja (Networking), hallita reitittimiä (Routers) ja tutkia verkkotopologiaa (Network topology).

Kuvio 45. OpenStack-järjestelmän peruskäyttäjän hallintavälilehdet

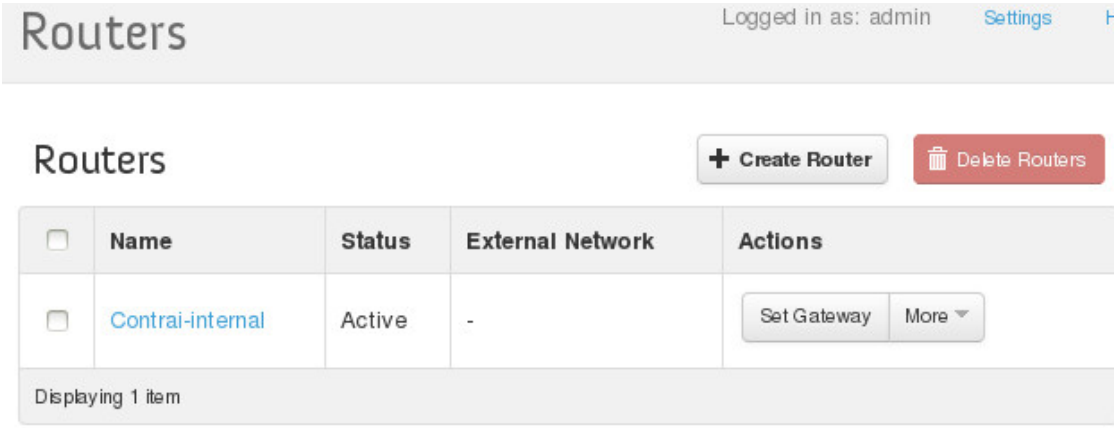
Järjestelmänvalvoja voi suorittaa samoja toimintoja kuin peruskäyttäjä. Lisäksi järjestelmänvalvojalla on mahdollisuus hallita käyttäjiä (Users), käyttäjien oikeuksia, ryhmiä (Groups), projekteja (Projects), toimialueita (Domains), rooleja (Roles) ja virtuaalitietokoneisiin asetettavia resursseja (Flavors). Liitteessä 9 on virtualitietokoneisiin asetettavien resurssien luontiin käytettävä *Flavors*-valikko. Tätä kautta järjestelmänvalvoja voi määrittää millaisia virtuaalisia tietokoneita käyttäjät voivat käynnistää.

Peruskäyttäjänä verkkojen luonti ja hallinta tapahtuu valikoiden kautta. Kuviossa 46 on esitettynä *demo*-käyttäjän *Networking*-välilehden alta löytyvä verkkojen hallinnan valikko. Tämän valikon kautta käyttäjä voi luoda verkkoja, liittää verkkoon politiikkoja ja suorittaa IP-osoitteiden hallintaa. Testiverkon luominen tapahtuu kuviossa 46 korostetun *Create Network* –painikkeen kautta. Ohjelma kysyy tämän jälkeen luotavan verkon nimen, aliverkon tiedot ja verkkoon liitettävät politiikat. Kuviossa 46 voidaan nähdä painikkeen kautta luotu testiverkko *Contrail* sekä *OpenContrail*-hallintapaneelin kautta luotu *Contrail-dash-testi* -verkko. *Contrail*-verkolle luotiin aliverkko 10.0.0.0/24 eikä siihen liitetty erillisiä politiikkoja.



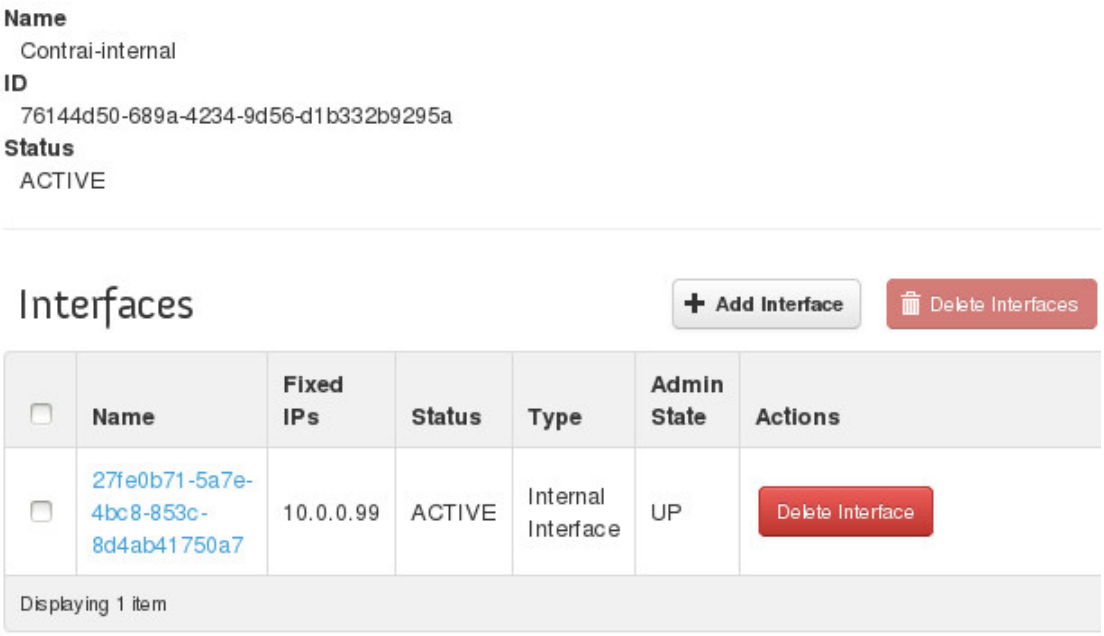
Kuvio 46. Verkonluonnin hallintapaneeli

Todennusta varten luodaan reitin *Routers*-välilehden kautta löytyvästä valikosta, painamalla *Create Router* -nappia. Tällöin ohjelma kysyy luotavan reitittimen nimen, joka tässä todennuksessa on *Contrail-internal*. *Routers*-valikko ja Contrail-internal -reititin on esitetty kuviossa 47.



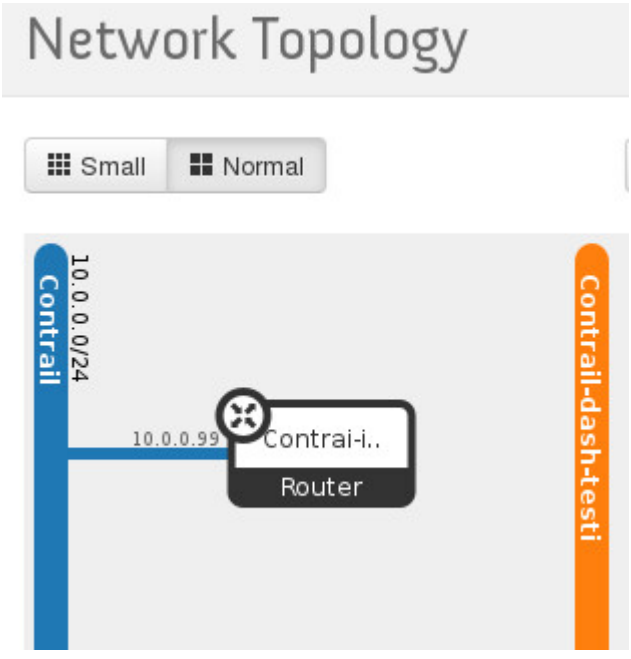
Kuvio 47. Routers-hallintapaneeli

Painamalla luodun reitittimen nimeä voidaan hallita reitittimen verkkorajapintoja. Avautuvan valikon kautta voidaan luoda reitittimelle verkkorajapinta ja liittää se aiemmin luotuun verkkoon. Kuviossa 48 on esitettynä reitittimen rajapintojen hallinnan valikko ja *Contrail-internal* –reitittimelle annetun rajapinnan IP-osoite 10.0.0.99.



Kuvio 48. Reitittimen verkkorajapinnat

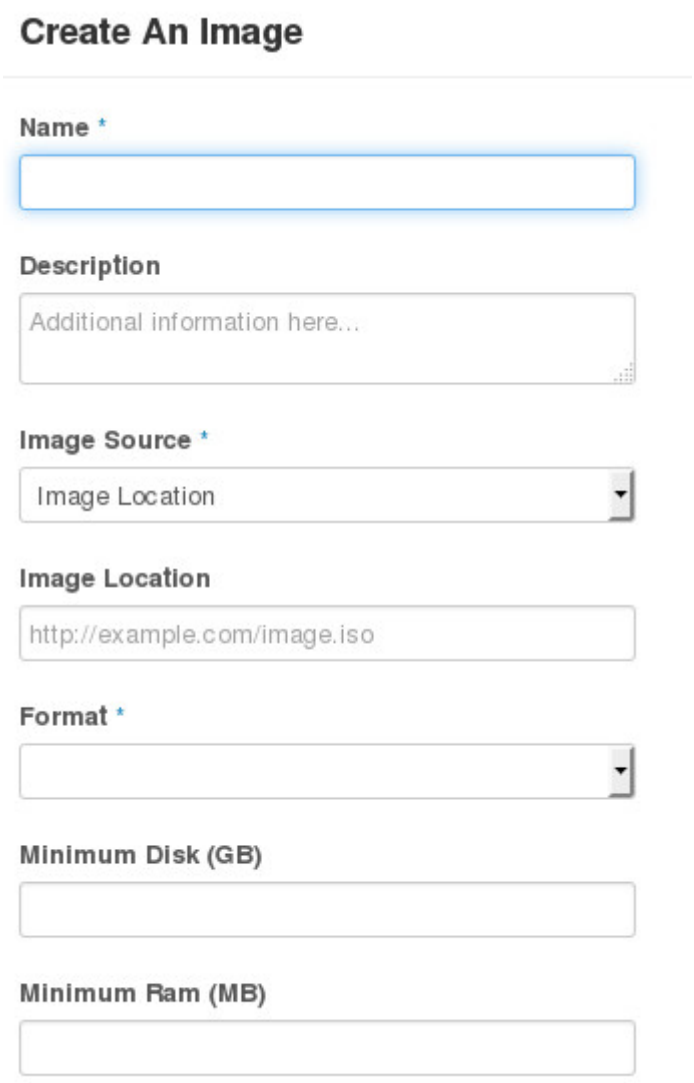
Tämän jälkeen kytkentä voidaan tarkastaa *Network Topology* –välilehden alta. Kuvissa 49 on osoitettuna luotujen *Contrail-* ja *Contrail-dash-testi* –verkkojen sekä *Contrail-internal* –reitittimen näkyminen topologiakuvassa.



Kuvio 49. Verkkotopologia

Seuraavaksi voidaan virtualisoida tietokone OpenStack-järjestelmällä. Testivirtuaalikonetta varten tulee OpenStack-järjestelmään luoda levykuva käynnistettävästä virtuaalitietokoneesta tai virtuaalitietokoneelle asetettavasta käyttöjärjestel-

mästä. Tämä voidaan toteuttaa kahdella tavalla. Ensimmäinen tapa on *Images & Snapshots* –välilehden alta painamalla *Create Image* –painiketta. Kuviossa 50 on esitetty *Create Image* –painikkeesta avautuva valikko, jonka kautta voidaan luoda levykuva.



Kuvio 50. Levykuvan luonnin valikko

Toinen tapa luoda levykuva on ladata haluttu levykuva OpenContrail-palvelimelle ja suorittaa levykuvan lisääminen komentorivin kautta. Tässä todennuksessa käytettiin levykuvan lisäämistä komentorivin kautta. Levykuvan lisääminen aloitetaan lataamalla haluttu levykuva. Seuraavalla komennolla ladataan *CirrOS*-käyttöjärjestelmän levykuva OpenContrail-palvelimelle.

```
[root@contrail ~]# wget download.cirros-
cloud.net/0.3.3/cirros-0.3.3-x86_64-disk.img
```

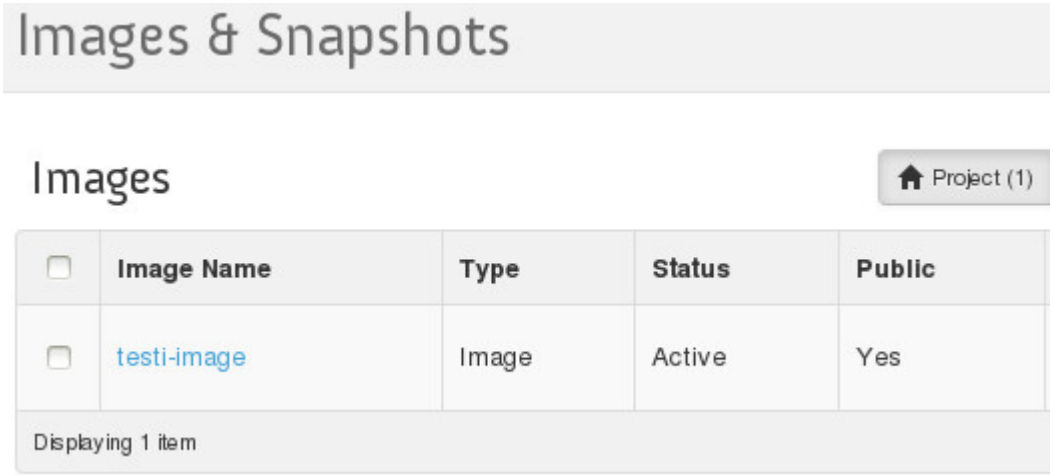
Tämän jälkeen annetaan OpenStack-palvelulle käyttäjätiedot levykuvan lisäämistä varten seuraavasti.

```
[root@contrail ~]# export OS_USERNAME=admin
[root@contrail ~]# export OS_PASSWORD=secret123
[root@contrail ~]# export OS_PROJECT_NAME=demo
[root@contrail ~]# export OS_AUTH_URL=http://192.168.0.3/v2.0
[root@contrail ~]# export OS_TENANT_NAME=demo
```

Tämän jälkeen luodaan *testi-image* –niminen levykuva, joka on julkinen ja formaatti on qcow2 seuraavalla komennolla.

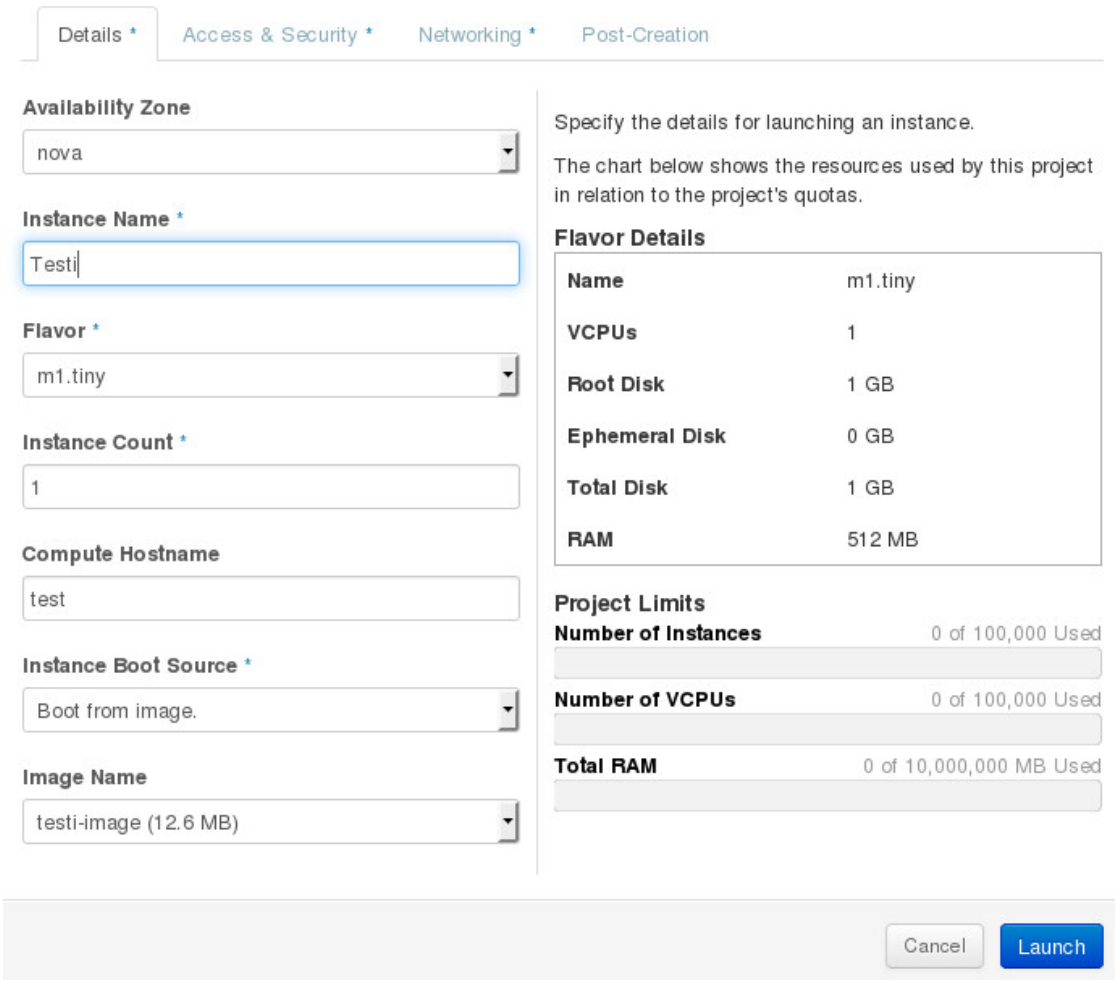
```
[root@contrail ~]# glance image-create --name testi-image --
disk-format qcow2 --container-format-bare --is-public TRUE –
file ./cirros-0.3.3-x86_64-disk.img
```

Tämän jälkeen levykuva voidaan todentaa *Images & Snapshots* –välilehden alta. Kuviossa 51 on esitettynä *testi-image* kyseisessä välilehdessä.



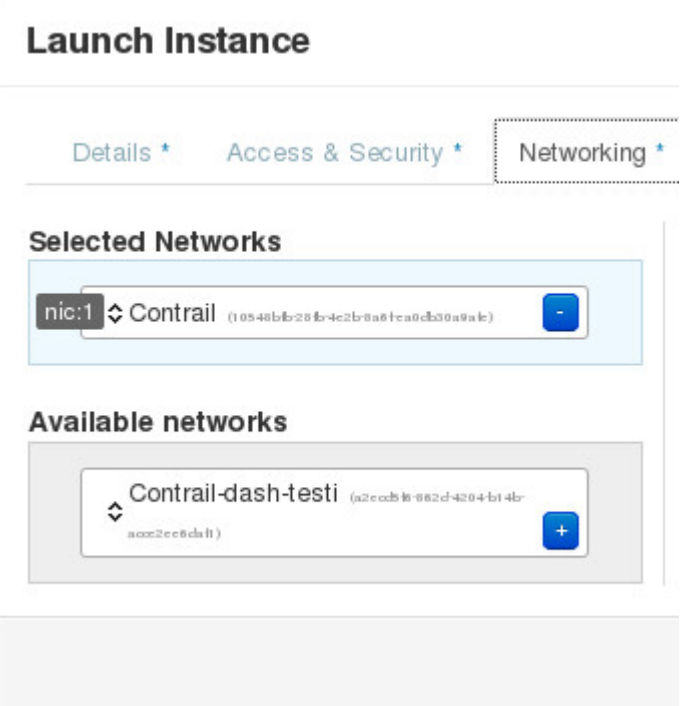
Kuvio 51. Valmis levykuva

Luodun levykuvan avulla voidaan käynnistää virtuaalitietokone. Tämä tapahtuu *Instances* –välilehden alta painamalla *Create Instance* –painiketta. Todennusta varten luodaan *Testi*-niminen virtuaalitietokone, joka kytketään *Contrail*-verkkoon. Kuviossa 52 on esitettynä Testi-virtuaalitietokoneelle annettavat resurssit.



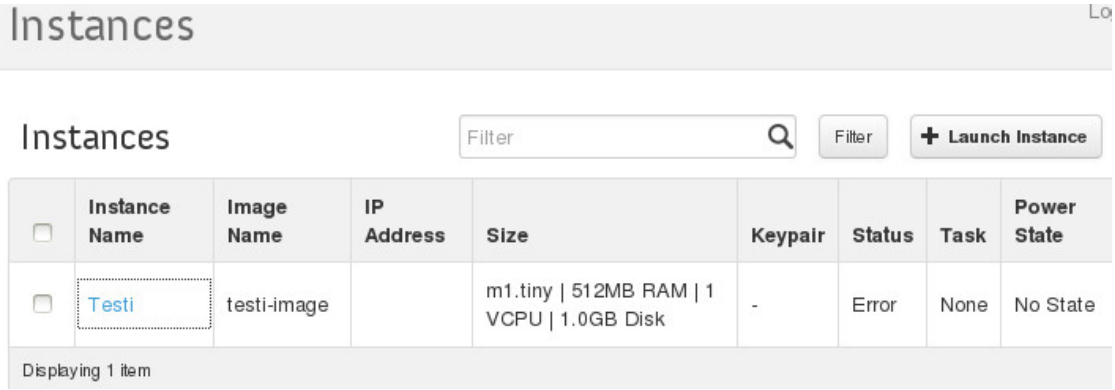
Kuvio 52. Virtuaalitietokoneen resurssit

Testi-virtuaalitietokoneen lisääminen *Contrail* verkkoon tapahtuu kuviossa 52 näkyvän *Networking*-välilehden alta. Kuviossa 53 on kuvattuna kyseinen välilehti ja asetukset *Contrail*-verkkoon liittämistä varten.



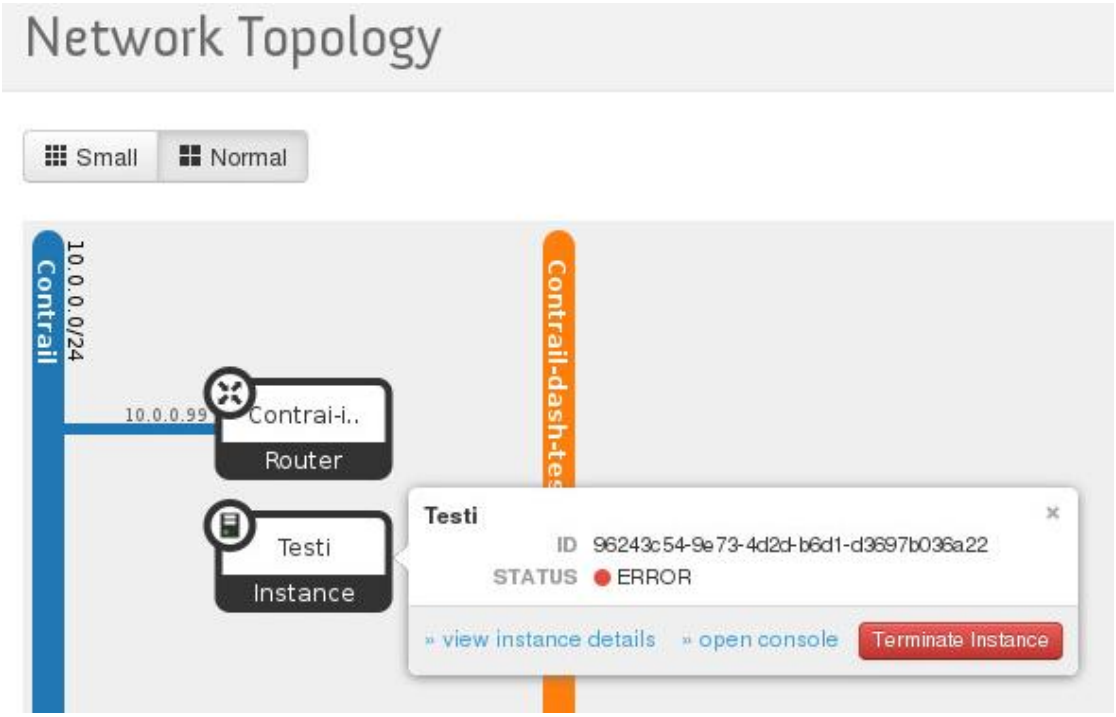
Kuvio 53. Virtuaalitietokoneen verkko

Tämän jälkeen painetaan *Launch*-painiketta, jolloin virtuaalitietokone käynnistetään. Virtuaalitietokone näkyy nyt *Instances & Snapshots* –välilehden kautta saatavassa valikossa kuvion 54 mukaisesti.



Kuvio 54. Virtuaalitietokone

Lisäksi *Testi*-virtuaalitietokone on havaittavissa verkkotopologiakuvassa. Testi-virtuaalitietokonetta ei saatu todennuksessa toimimaan täydellisesti. Kuviossa 55 on esitettynä verkkotopologiakuva Testi-virtuaalitietokoneen kanssa ja ilmenevä virheti-la.



Kuvio 55. Virtuaalitietokone verkossa

Yhteyttä virtuaalitietokoneen päätteelle ei saatu muodostettua. Yritettäessä avata virtuaalitietokoneen päätettä painamalla kuviossa 55 näkyvää *open console* –painiketta, tulee vastaan virheilmoitus ”Unable to get VNC console for instance…”. Ongelmaa ei saatu korjattua ja siitä johtuen virtuaalitietokoneiden toimintaa ei voitu todentaa enempää.

7 Tulokset

7.1 SDN-tekniikan hyödyt ja haitat

SDN-tekniikka osoittaa suurta potentiaalia ja on selkeästi vahvimmillaan suurissa verkoissa. SDN-tekniikan hyötyjä ovat selkeästi valmistajariippumattomuus, hienojakoisemmat hallintamahdollisuudet tietoverkoille ja muokattavuus. Erityisesti ohjelmapohjainen muokattavuus antaa verkoille erilaisia käyttömahdollisuuksia ja mahdollistaa verkkoresurssien optimoinnin.

SDN-tekniikan haittana voidaan mieltää sen keskeneräisyys. SDN-tekniikka on kehitysvaiheessa ja vaatii siksi tällä hetkellä resursseja suunnitteluun, käyttöönottoon ja

ylläpitoon. Monet avoimen lähdekoodin SDN-kontrollereista on tarkoitettu yritykselle pohjaksi, josta voidaan muokata yrityksen tarpeita vastaava SDN-kontrolleri.

7.2 OpenDaylight ja OpenContrail

Todennuksen myötä voidaan huomata, että OpenDaylight ja OpenContrail eroavat selkeästi käyttötarkoituksiltaan. OpenDaylight on suunniteltu fyysisen tietoliikenneverkon hallintaan kun taas OpenContrail on selkeästi suunnattu kokonaisratkaisuna pilvipalveluksi.

OpenDaylight on perusteiltaan selkeä käyttää, mutta tarjoaa silti monipuolisia laajennusmahdollisuuksia. Näillä laajennuksilla OpenDaylight voidaan muokata halutuun muotoon SDN-verkkoja varten. Lisäksi OpenDaylight tukee uusien ohjelmien lisäämistä ja useita rajapintoja. Täten OpenDaylight toimii erinomaisena pohjana esimerkiksi yrityksille, jotka ovat halukkaita luomaan oman SDN-kontrollerin. Yritys voi ottaa OpenDaylight-mallisen ratkaisun ja muokata sen omanlaisekseen.

OpenContrail on selkeästi suunniteltu konesaleja ja pilvipalveluja varten. Hajautettuna järjestelmänä OpenContrail sopii erityisesti konesaleihin. Konesalin resurssit voidaan asettaa OpenContrail-järjestelmän hallintaan, sillä se kykenee toimimaan myös ulkoisten verkkolaitteiden kanssa. Tällöin yhdellä järjestelmällä voidaan hallita sekä verkko- että palvelininfrastruktuuria.

Kumpikaan SDN-kontrolleri ei toiminut täydellisesti, mutta pääasialliset toiminnot sekä toimintomahdollisuudet saatiin kartoitettua todennuksen yhteydessä. OpenDaylight saattoi päivittyä useasti kuukauden aikana. Päivitykset sekä korjasivat että toivat mukanaan ongelmia. On kuitenkin mahdollista, että sopivilla resursseilla molemmista SDN-kontrollereista voidaan toteuttaa toimiva ratkaisu, jota voidaan soveltaa opetukseen, tutkimiseen ja mahdollisesti toimintaan sekä JAMKissa että JYVSECTEC-hankkeessa.

Tällä hetkellä JAMK ei omista laitteita, jotka suoraan tukisivat OpenFlow-protokollaa, joten OpenDaylight-kontrolleria ei voida hyödyntää toiminnassa. Lisäksi OpenDaylight-kontrolleri osoittautui tässä vaiheessa liian epävakaaksi, että sitä kannattaisi soveltaa JAMK:in verkkoinfrastruktuuriin, muuten kuin tutkimuksellisesti.

7.3 Näkökulmia SDN-tekniikan koulutuksesta

SDN on nopeasti kehittyvä tekniikka, joka kiinnostaa erityisesti konesalipalveluiden tarjoajia sekä suuria verkko-operaattoreita. Näin ollen SDN-tekniikan koulutusta tulisi suunnitella JAMKissa. Opiskelijoille voitaisiin luoda mahdollisuus valita kurssi SDN-tekniikan perusteista, jossa voidaan käsitellä erityisesti OpenFlow-protokollaa käyttäviä SDN-ratkaisuja. Tällä hetkellä OpenFlow-protokolla on laajimmin käytössä oleva ja tunnetuin verkkoelementtien hallintaprotokolla SDN-tekniikassa.

Toimivalla OpenDaylight-kontrollerilla opiskelijoille voitaisiin osoittaa käytännössä SDN-tekniikan toiminta. Lisäksi opetusta voidaan kohdentaa OpenDaylight-kontrollerin modulaarisuuden avulla. Esimerkiksi tietoturvaa käsitellessä voitaisiin ottaa käyttöön Defense4all-toteutus ja suojata liikenne TLS-protokollalla SDN-kontrollerin sekä verkkoelementtien välillä. Olisi mahdollista myös tehdä erilaisia kokonaisuuksia VTN-verkkoja käyttäen. Tällöin perusharjoitusten ja teorian jälkeen opetuksessa voidaan tehdä erimuotoisia toteutuksia. OpenDaylight ja SDN-tekniikka voitaisiin siis hyvin sisällyttää QoS- ja tietoturvakurssien kokonaisuuksiin JAMK:issa.

OpenContrail-järjestelmää voitaisiin soveltaa konesaleja ja palveluita käsittelevillä kursseilla. OpenContrail-järjestelmän avulla opiskelijat voitaisiin perehdyttää avoimen lähdekoodin pilvipalveluun sekä tietoliikennemalleihin ja tietoverkkoihin pilvipalveluiden sisällä.

7.4 Näkökulmia tekniikan jatkotutkimuksesta

Voidaan todeta, että SDN-tekniikka on hyvin laaja alue. SDN-tekniikka ei rajoitu ainoastaan valmiisiin SDN-kontrollereihin vaan jatkuu myös ohjelmistopuolelle, jossa voidaan luoda tarvittavat automatisoinnit ja verkonohjausohjelmat SDN-kontrollerin lisäksi.

Molemmat SDN-kontrollerit omina kokonaisuuksinaan ja erityisesti OpenDaylight-kontrollerin moduulit, kuten Defense4all, tarjoavat paljon jatkotutkimusaiheita. Lisäksi SDN-tekniikkaan vahvasti liittyvää NFV-tekniikkaa ja sen tuomia mahdollisuuksia olisi syytä tutkia tarkemmin.

SDN-tekniikassa on vahvasti mukana avoimen lähdekoodin ratkaisut, mutta myös kaupallisia ratkaisuja on saatavilla. Useat valmistajat ovat tuoneet markkinoille omat kaupalliset SDN-kontrollerinsa tai SDN-pilvipalvelujärjestelmät, joten SDN-tekniikan tutkimista ei tulisi rajoittaa ainoastaan avoimenlähdekoodin puolelle.

Nykypäivänä tietoturva on suuressa roolissa tietoliikennetekniikassa. Erityisesti suuret yritykset panostavat tietoturvaan omissa verkoissaan. JYVSECTEC-hanke voisi tutkia SDN-tekniikan vaikutusta tietoturvan kannalta.

8 Pohdinta

8.1 Työn tuloksien arviointi

Työ aloitettiin tutkimalla SDN-tekniikkaa ja sen toimintaa teoriatasolla. Seuraava askel oli rajata työn aihe laajasta kokonaisuudesta. Aiheen rajaaminen oli hankalaa, sillä SDN-tekniikka on hyvin monimuotoista ja suuri osa siitä koostuu useiden asioiden kokonaisuuksista.

Tämän jälkeen selvitettiin OpenContrail- ja OpenDaylight-kontrollerien toimintaa. Olin aikaisemmin käsitellyt kyseisiä SDN-kontrollereita ja osittain todentanut niiden toimintaa, mutta niiden versiot olivat päivittyneet huomattavasti opinnäytetyön toteutusta aloittaessa. Toteutus toivottiin tehtävän uusimpien versioiden mukaisesti, mikäli mahdollista. Päivittyneet ohjelmat eivät kuitenkaan toimineet ja sen takia jouduttiin osittain palaamaan vanhoihin versioihin.

Toteutuksessa huomattiin, etteivät kaikki halutut ominaisuudet toimi ja resurssien puitteissa tälle ei mahdettu mitään. Toteutusta jatkettiin niin pitkälle kuin mahdollista, että saatiin havainnollistettua OpenDaylight- ja OpenContrail-kontrollerien toiminta.

Mielestäni työ ei onnistunut täysin halutulla tavalla. Vaatimukset saatiin pääosin toteutettua. Aihetta ja toteutusta jouduttiin muokkaamaan kesken toteutuksen, sillä SDN-kontrollerien ja alkuperäisen toteutuksen kanssa havaittiin niin suuria ongelmia, ettei niitä voitu toteuttaa. Mukana alkuperäisessä suunnitelmassa ja vielä toteutus-

vaiheessakin oli Juniper vMX -virtuaalireititttimen testiversio, joka myöhemmin jätettiin pois.

Erityisesti nopea kehitys heijastui OpenDaylight-kontrolleriin. Kyseinen kontrolleri koostuu useista projekteista, joilla kaikilla on omat kehittäjänsä. Tällöin yhden projektin päivittyessä, voi kyseinen projekti lakata toimimasta muiden kanssa. Lisäksi kaikki ohjelmat tuntuivat olevan melko epäluotettavia. Esimerkiksi ohjelma saattoi toimia aluksi moitteetta, mutta palvelimen uudelleenkäynnistyksen jälkeen ohjelma ei välttämättä enää käynnistynytkään.

Opinnäytetyö SDN-tekniikasta oli erittäin haastava kokonaisuus. Materiaalia oli tarjolla niukasti ja siitäkin suuri osa voitiin tulkita kaupalliseksi myyntitekstiksi. Paikoitellen ainoastaan puoli vuotta vanha lähde saattoi olla nopeasti kehittyvälle SDN-tekniikalle vanhentunutta tietoa johtuen SDN-tekniikan ja ohjelmien nopeasta kehityksestä.

Kokonaisuutena opin paljon opinnäytetyön aikana. Erityisesti työn haasteellisuus kasvatti paljon omaa osaamista, sillä työnteko oli itsenäistä ja sisälsi paljon ongelmanratkaisua. SDN-tekniikan lisäksi opin työn aikana käyttämään monipuolisesti erilaisia Linux-käyttöjärjestelmiä sekä ymmärtämään perusteet Python-ohjelmointikielestä. Työ itsessään oli mielekästä, koska mielestäni SDN-tekniikkaa tulisi käsitellä tietoverkkotekniikan koulutuksessa. Lisäksi työ tarjosi paljon uusia näkemyksiä tietoliikennetekniikasta, sen kehityksestä ja tulevaisuudesta.

8.2 Kehittämisideat

Tässä opinnäytetyössä käytetyt kontrollerit voitaisiin saattaa vakaaseen toimintaan, mikäli resursseja olisi käytettävissä tarpeeksi. Molempien kontrollerien kohdalla voitaisiin todeta, että vakaina ja toimivina kokonaisuuksina ne on mahdollista sovittaa Jyväskylän ammattikorkeakoulun sekä JYVSECTEC-hankkeen koulutuksiin.

OpenFlow-protokollan vuopohjainen liikennöinti ja sen myötä vuomerkinnät ja niiden muutokset tuovat lisää tietoliikennettä tietoverkkoon. Olisi siis mahdollista tutkia kuinka suuret määrät vuomerkintöjä tai niiden muutoksia vaikuttaa verkkoelementteihin sekä SDN-kontrollereihin.

SDN-tekniikassa riittää paljon tutkimista sekä ohjelmistopuolelle että tietoliikennepuolelle. Tietoliikennetekniikan osalta erityisesti Defense4All-järjestelmä, Mininet-verkkoemulaattorin laajempi käyttö ja SDN-kontrollerien käyttö eri verkkoelementtien kanssa voisivat olla selkeitä opinnäytetyöaiheita, joiden pohjalta SDN-tekniikan tutkimista voi jatkaa.

Lähteet

Benitez, J., Bugenhagen, M., Chiosi, M., Clarke, D., Cui, C., Damker, H., Delisle, D., Demaria, E., Deng, H., Fargano, M., Feger, J., Fukui, M., Guardini, I., Khan, W., Kolias, C., Loudier, Q., López, D., Manzalini, A., Matsuzaki, T., Michel, U., Minerva, R., Ogaki, K., Reid, A., Ruhl, F., Salguero, F., Sen, P., Shimano, K. & Willis, P. 2012. Network Functions Virtualisation. Raportti aiheesta Network functions virtualisation. 24.10.2012. Viitattu 10.12.2014. http://portal.etsi.org/NFV/NFV_White_Paper.pdf

Carrier Ethernet and SDN. 2014. Dokumentti Metro ethernet forumin verkkosivustolla. Viitattu 24.3.2015. http://www.metroethernetforum.org/Assets/White_Papers/Carrier_Ethernet_and_SDN_Part_1_-_An_Industry_Perspective_08-14-14.pdf

Downloads archive. N.d. Lista OpenDaylight-ohjelman versioista ja latauksista OpenDaylight-projektin verkkosivustolla. http://www.opendaylight.org/software/release-archives

Handigol, N., Heller, B., Jeyakumar, V. & Lantz, B. N.d. Introduction to Mininet. Dokumentti Mininetin Github verkkosivulla. Viitattu 10.3.2015 https://github.com/mininet/mininet/wiki/Introduction-to-Mininet

Hogg, S. 2014. SDN Security Attack Vectors and SDN Hardening. Artikkeli Networkworld verkkosivustolla 28.10.2014. Viitattu 4.3.2015. http://www.networkworld.com/article/2840273/sdn/sdn-security-attack-vectors-and-sdn-hardening.html

Hubbard, S. 2012. Software-Defined Metro Networks: Virtualizing the Network & Services Edge. Dokumentti Cyanin verkkosivustolla. Viitattu 24.3.2015. http://www.cyaninc.com/assets/docs/whitepapers/Heavy_Reading_SDN_Metro_Networks.pdf

JYVSECTEC. N.d. JYVSECTEC-hannkkeen verkkosivusto. Viitattu 10.12.2014. http://jyvsectec.fi/en

Jyväskylän ammattikorkeakoulu. N.d. Jyväskylän ammattikorkeakoulun verkkosivusto. Viitattu 10.12.2014. http://www.jamk.fi/fi/Etusivu/

Member Listing. N.d. Open networking foundation verkkosivusto. Viitattu 4.1.2015. https://www.opennetworking.org/our-members

Metzler, J. 2012. What is software defined networkin (SDN)? 29.8.2012. Viitattu 4.1.2015. http://www.networkworld.com/article/2159545/software/what-is-software-defined-networking--sdn--.html

ONF Overview. N.d. Open networking foundation verkkosivusto. Viitattu 4.1.2015. https://www.opennetworking.org/about/onf-overview

Open vSwitch. N.d. Dokumentaatio Open vSwitch Github verkkosivulla. Viitattu 1.4.2015. https://github.com/openvswitch/ovs/blob/master/README.md

OpenContrail – Quick Start Guide. N.d. Dokumentti OpenContrail-järjestelmän käytöstä OpenContrailin verkkosivustolla. Viitattu 24.3.2015. http://www.opencontrail.org/opencontrail-quick-start-guide/

OpenDaylight Controller:Architectural Framework. 2014. Wiki-dokumentti OpenDaylight-projektin omalla wikisivustolla. Viitattu 24.3.2015. https://wiki.opendaylight.org/view/OpenDaylight_Controller:Architectural_Framework

OpenDaylight Virtual Tenant Network (VTN):Overview. Artikkeli OpenDaylightin wikisivustolla. N.d. Viitattu 8.1.2015 https://wiki.opendaylight.org/view/OpenDaylight_Virtual_Tenant_Network_%28VTN%29:Overview

OpenFlow Switch Specification. 2014. Dokumentti OpenFlow-protokollasta. Viitattu 14.1.2015. https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-switch-v1.3.4.pdf

OpenFlow. N.d. Artikkeli Open networking foundationin verkkosivustolla. Viitattu 14.1.2015. https://www.opennetworking.org/sdn-resources/openflow

OpenFlow-Enabled Hybrid Cloud Services Connect Enterprise and Service Provider Data Centers. 2012. Dokumentti Open networking foundationin verkkosivustolla. Viitattu 24.3.2015. https://www.opennetworking.org/images/stories/downloads/sdn-resources/solution-briefs/sb-hybrid-cloud-services.pdf

OpenStack Compute. N.d. Dokumentti OpenStack-järjestelmän kotisivulla. Viitattu 3.4.2015. http://www.openstack.org/software/openstack-compute/

OpenStack Dashboard. N.d. N.d. Dokumentti OpenStack-järjestelmän kotisivulla. Viitattu 3.4.2015. http://www.openstack.org/software/openstack-dashboard/

OpenStack Networking N.d. Dokumentti OpenStack-järjestelmän kotisivulla. Viitattu 3.4.2015. http://www.openstack.org/software/openstack-networking/

OpenStack Storage. N.d. Dokumentti OpenStack-järjestelmän kotisivulla. Viitattu 3.4.2015. http://www.openstack.org/software/openstack-storage/

OpenStack: The Open Source Cloud Operating System. N.d. Dokumentti OpenStack-järjestelmän kotisivulla. Viitattu 3.4.2015. http://www.openstack.org/software/

Pepelnjak, I. 2013. ProgrammableFlow Technical Deep Dive. . Webinaari ipSpace verkkosivustolla 13.2.2013. Viitattu 12.2.2015. http://demo.ipspace.net/get/2.1%20-%20ProgrammableFlow%20Basics.mp4

Release/Helium/Virtualization/User Guide. 2014. Wiki-dokumentti OpenDaylight-virtualization –versiosta OpenDaylight-projektin omalla wikisivustolla. Viitattu 24.3.2015. https://wiki.opendaylight.org/view/Release/Hydrogen/Virtualization/User_Guide

Release/Hydrogen/Base. 2014. Wiki-dokumentti OpenDaylight-base –versiosta OpenDaylight-projektin omalla wikisivustolla. Viitattu 24.3.2015. https://wiki.opendaylight.org/view/Release/Hydrogen/Base

Release/Hydrogen/Service Provider/User Guide. 2014. Wiki-dokumentti Open-Daylight-Service Provider –versiosta OpenDaylight-projektin omalla wikisivustolla. Viitattu 24.3.2015. https://wiki.opendaylight.org/view/Release/Hydrogen/Service_Provider/User_Guide

Salisbury, B. 2012a. The Control Plane, Data Plane and Forwarding Plane in Networks. Artikkeli NetworkStatic verkkosivustolla. Julkaistu 27.9.2012. Viitattu 12.12.2014. http://networkstatic.net/the-control-plane-data-plane-and-forwarding-plane-in-networks/

Salisbury, B. 2012b. Juniper and Cisco Comparisosns of RIB, LIB, FIB and LFIB Tables. Artikkeli NetworkStatic verkkosivustolla. Julkaistu 15.4.2012. Viitattu 7.3.2015. http://networkstatic.net/juniper-and-cisco-comparisons-of-rib-lib-fib-and-lfib-tables/

SDN architecture. 2014. Dokumentti Open networking foundation verkkosivustolla Viitattu 18.1.2015. https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/TR_SDN_ARCH_1.0_06062014.pdf

Singla, A. & Rijsman, B. N.d. Architecture Documentation. Dokumentti OpenContrail-järjestelmän arkkitehtuurista OpenContrail verkkosivustolla. Viitattu 24.3.2015. http://www.opencontrail.org/opencontrail-architecture-documentation/

Software-defined Networking: The New Norm for Networks. 13.4.2012. Viitattu 4.1.2015. https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf

Technical overview. N.d. Artikkeli OpendayLight projektin verkkosivustolla. Viitattu 22.1.2015. http://www.opendaylight.org/project/technical-overview

What are SDN Controllers? N.d. Artikkeli SDN Centralin verkkosivulla. Viitattu 8.1.2015. https://www.sdxcentral.com/resources/sdn/sdn-controllers/

Liitteet

Liite 1. OpenDaylight-kontrollerin asennus

OpenDaylight-virtualization versio voidaan asentaa asennuspaketeista (.rpm) tai ladata valmiiksi asennettu kontrolleri, joka tarvitsee purkaa. Tässä tapauksessa valittiin asentaminen asennuspaketeista.

Asennus tapahtuu CentOS 7 minimal käyttöjärjestelmälle. Ensimmäisenä asennetaan Java ja wget seuraavilla komennoilla,

```
[root@odl ~]# yum install java-1.7.0-jdk
[root@odl ~]# yum install wget
```

Tämän jälkeen asetetaan Javan kotikansio seuraavasti.

```
[root@odl ~]# export JAVA_HOME=/usr/lib/jvm/java-1.7.0-
openjdk-1.7.0.75-2.5.4.2.el7_0.x86_64
```

Seuraavaksi haetaan ja asennetaan tiedot asennuspaketeista. Tämä tapahtuu seuraavilla komennoilla.

```
[root@odl ~]# wget
https://nexus.opendaylight.org/content/repositories/opendayli
ght-yum-fedora-19-x86_64/rpm/opendaylight-release/0.1.0-
2.fc19.noarch/opendaylight-release-0.1.0-2.fc19.noarch.rpm
[root@odl ~]# rpm –Uvh Opendaylight-release-0.1.0-
2.fc19.noarch.rpm
```

Nyt on mahdollista asentaa haluttu versio OpenDaylight-kontrollerista. Tässä tapauksessa halutaan asentaa virtualization versio. Tämä tapahtuu seuraavasti.

```
[root@odl ~]# yum install opendaylight-virtualization
```

Asennuksen jälkeen sallitaan OpenDaylight-kontrolleri palvelu ja käynnistetään se seuraavilla komennoilla.

```
[root@odl ~]# systemctl enable opendaylight-
controller.service
[root@odl ~]# systemctl start opendaylight-controller.service
```

Nyt OpenDaylight-kontrolleri on käynnissä ja sen toimintaa voidaan todentaa menemällä webselaimella kontrollerin ositteeseen käyttäen porttia 8080. Esimerkiksi *192.168.0.4:8080*.

Liite 2 OpenDaylight-ympäristön asentaminen

Graafinen käyttöliittymä

Ensimmäisen Centos-laitteelle asennetaan graafinen käyttöliittymä todentamista varten. Tämän laitteen avulla voidaan todentaa OpenDaylight-kontrollerin graafista käyttöliittymää ja käyttää Mininet-verkkoemulaattoriin sisällytettyä Wireshark-ohjelmaa, jolla voidaan kaapata ja tarkastella verkossa tapahtuvaa tietoliikennettä. Graafisen käyttöliittymän asentaminen tapahtuu seuraavalla komennolla.

```
[root@Centos ~]# yum –y groupinstall ”Desktop” ”Desktop Plat-
form” ”X Window System” ”Fonts” ”Graphical Administration
Tools” ”Internet Browser”
```

Tämän jälkeen Centos-laite on uudelleenkäynnistettävä, jonka jälkeen graafisen käyttöliittymän voi käynnistää. Uudelleenkäynnistys ja graafisen käyttöliittymän käynnistys tapahtuu seuraavilla komennoilla.

```
[root@Centos ~]# reboot
[root@Centos ~]# startx
```

Mininet

Seuraava askel on valmistella Mininet. Mininet on saatavissa valmiina virtuaalikoneena, jonka voi käynnistää haluamallaan virtualisointiohjelmalla. Tässä tapauksessa Mininet-VM käynnistettiin vCloud-ympäristössä valmiista levykuvasta. Mininet levykuva on ladattavissa osoitteesta https://github.com/mininet/mininet/wiki/Mininet-VM-Images. Tässä todennuksessa käytettiin Mininetistä versiota 2.2.0.

Mininetillä voidaan emuloida tietoverkkoja. Tässä tapauksessa sitä käytetään emuloimaan päätelaitteita ja kytkimiä, joita voidaan ohjata OpenDaylight-kontrollerilla. Mininet-VM –laitteelle kirjautuminen tapahtuu käyttäjätunnuksella *mininet* ja salasanalla *mininet*. Mininetin toimivuus voidaan testata Mininet-VM –laitteella seuraavalla komennolla.

```
mininet@mininet-vm:~$ sudo mn –test pingall
```

Edellinen komento luo Mininetillä yhden kytkimen, johon liitetään kaksi päätelaitetta. Tämän jälkeen päätelaitteet suorittavat *ping*-komennon toisiinsa. Tuloksena tulisi nähdä ”completed in” –ilmoitus.¨

Mikäli emulointi halutaan lopettaa, voidaan se tehdä seuraavalla komennolla.

```
mininet>exit
```

Emulaation sammuessa voi Mininet-VM –laitteelle jäädä ylimääräisiä emuloituja laitteita päälle. Tämä on mahdollista erityisesti emulaation kaatuessa. Tämän takia on hyvä aina emulaation päätteeksi puhdistaa emulaation mahdolliset jäänteet seuraavalla komennolla.

```
mininet@mininet-vm:~$ sudo mn -c
```

OpenDaylight-kontrolleri

Seuraavaksi asennetaan erillinen SDN-kontrolleri laittelle ODL. Ensimmäisenä tarkistetaan, että järjestelmä on päivitetty seuraavalla komennolla.

```
opendaylight@ODL:~$ sudo apt-get update
```

Tämän jälkeen asennetaan OpenDaylight-kontrollerin asennukseen ja toimintaan tarvittavat riippuvuudet. Näitä ovat Maven, Java ja Git. Riippuvuudet asennetaan seuraavalla komennolla.

```
opendaylight@ODL:~$ sudo apt-get install maven git openjdk-7-jre openjdk-7-jdk
```

Asennuksen jälkeen on ladattava OpenDaylightin asennukseen tarvittavat tiedostot ja asennettava itse OpenDaylight. Nämä tapahtuvat seuraavilla komennoilla.

```
opendaylight@ODL:~$ git clone http://git.opendaylight.org/gerrit/p/controller.git
opendaylight@ODL:~$ cd controller/opendaylight/distribution/opendaylight
opendaylight@ODL:~$ mvn clean install
```

Tämän jälkeen on muutettava Javan kotikansio OpenDaylight-kontrollerin toimintaa varten. Tämä tapahtuu muokkaamalla tekstieditorilla kotikansiossa sijaitsevaa .bashrc-tiedostoa. Seuraavalla komennolla voidaan avata kyseinen tiedosto.

```
opendaylight@ODL:~$ sudo nano /home/odl/.bashrc
```

Tiedoston loppuun lisätään seuraava rivi.

```
JAVA_HOME=:/usr/lib/jvm/java-1.7.0-openjdk-amd64
```

Tämän jälkeen on mahdollista käynnistää OpenDaylight-kontrolleri. Käynnistys tapahtuu suorittamalla erillinen skripti. Seuraavilla komennoilla siirrytään kansioon, jossa skripti sijaitsee ja käynnistetään skripti.

```
opendaylight@ODL:~$ cd
/controller/opendaylight/distibution/target/distribution.opendaylight-
osgipackage/opendaylight
opendaylight@ODL:~$ sudo ./run.sh
```

Tämän jälkeen Centos-koneelta käynnistetään webselain ja avataan selaimella OpenDaylight-kontrollerin hallintasivu kirjoittamalla selaimen osoitekenttään ODL-laitteen osoite. OpenDaylight-kontrolleri kuuntelee porttia 8080 eli osoitekenttään kirjoitetaan *192.168.0.4:8080*. Avautuvalla sivulla on mahdollista kirjautua kontrollerin hallintasivulle. Oletus käyttäjätunnus ja salasana ovat *admin*.

OpenDaylight-kontrolleri voidaan sammuttaa seuraavasti.

```
osgi>exit
Really want to sto Equinox? (y/n; defaylt=y) y
```

Koordinaattori

VTN-verkkoja varten on asennettava erillinen koordinaattori. Koordinaattori asennetaan Coordinator-laitteelle. Koordinaattorin avulla voidaan hallita useammalle kontrollerille ulottuvaa VTN-verkkoa. Koordinaattorin asennus alkaa asentamalla tarvittavat riippuvuudet seuraavalla komennolla.

```
[root@coordinator ~]# yum install make glibc-devel gcc gcc-c++ boost-devel openssl-
devel ant perlExtUtils-MakeMaker unix ODBC-devel perl-Digest-SHA uuid libxslt libcurl
libcurl-devel git
```

Tämän jälkeen asennetaan Java ja asetetaan Javan kotikansio. Tämä tapahtuu seuraavilla komennoilla, joista viimeinen on avautuvaan tekstitiedostoon lisättävä rivi Javan kotikansion määrittämiseksi.

```
[root@coordinator ~]# cd
[root@coordinator ~]# vi .bashrc
JAVA_HOME=/usr/lib/jvm/java-1.7.0-openjdk.x86_64
```

Seuraavaksi on haettava Postgresql tiedostot. Rpm-tiedostot ovat ladattavissa osoitteesta http://yum.postgresql.org/9.1/redhat/rhel-6.4-x86_64. Tiedostot, jotka tarvitaan, ovat postgresql91-libs, postgresql91, postgresql91server, postgresql91-contrib, postgresql91-odbc. Tiedostojen lataaminen ja asentaminen tapahtuu jokaisen rpm-tiedoston kohdalla seuraavalla tavalla.

```
[root@coordinator ~]# rpm -Uvh http://yum.postgresql.org/9.1/redhat/rhel-6.4-
x86_64/postgresql91-contrib-9.1.13-1PGDG.rhel6.x86_64.rpm
```

Tämän jälkeen tarvitaan Maven, joka asennetaan seuraavasti.

```
[root@coordinator ~]# wget http://repos.fedorapeople.org/repos/dchen/apache-maven/epel-apache-maven.repo -O /etc/yum.preos.d/epel-apache-mave.repo
[root@coordinator ~]# yum install apache-maven
```

Seuraavaksi asennetaan gtest-devel ja json-c kirjastot. Tämä tapahtuu seuraavilla komennoilla.

```
[root@coordinator ~]# wget http://dl.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm
[root@coordinator ~]# rpm –Uvh epel-release-6-8.noarch.rpm
[root@coordinator ~]# yum install gtest-devel json-c json-c-devel
```

Näiden jälkeen ladataan tarvittavat tiedostot VTN koordinaattoria varten ja suoritetaan koordinaattorin asennus seuraavilla komennoilla.

```
[root@coordinator ~]# git clone https://git.opendaylight.org/gerrit/p/vtn.git
[root@coordinator ~]# cd vtn/coordinator
[root@coordinator ~]# mvn –f dist/pom.xml package
[root@coordinator ~]# make install
```

VTN-koordinaattori tarvitsee toimiakseen tietokannan, joka luodaan seuraavasti.

```
[root@coordinator ~]#/usr/local/vtn/sbin/db_setup
```

Tämän jälkeen koordinaattori on valmis käynnistettäväksi. Käynnistys ja sammutus tapahtuvat seuraavilla komennoilla.

```
[root@coordinator ~]# /usr/local/vtn/bin/vtn_start
[root@coordinator ~]# /usr/local/vtn/bin/vtn_stop
```

Käynnissä ollessaan VTN-koordinaattorin toiminta voidaan todentaa seuraavalla komennolla.

```
[root@coordinator ~]# curl –X GET –H ’content-type: application/json’ –H ’username: admin’ –H ’password: adminpass’ –H ’ipaddr:127.0.0.1’ http://127.0.0.1:8083/vtn-webapi/api_version.json
```

Saatava vastaus tulisi olla seuraavaa mallia.

```
[root@coordinator ~]# {"api_version":{"version":"V1.1"}}
```

OpenDaylight-manager

Aikaisemmin asennettu OpenDaylight-kontrolleri ei sisällä VTN-verkkoja varten tarvittavaa manageria, jota koordinaattori ohjaisi. ODL- ja ODL2-laitteille asennetaan managerin sisältävä OpenDaylight-kontrolleri seuraavasti.

```
opendaylight@ODL:~$ git clone https://git.opendaylight.org/gerrit/p/vtn.git
```

Tiedostot latautuvat *vtn*-nimisen kansion alle. Seuraavaksi voidaan suorittaa varsinainen asentaminen.

```
opendaylight@ODL:~$ cd vtn
opendaylight@ODL:~$ sudo mvn –f manager/dist/pom.xml install
```

Asennettu kontrolleri voidaan käynnistää seuraavalla tavalla.

```
opendaylight@ODL:~$ cd /vtn/manager/dist/target/distribution.vtn-manager-0.2.0-
SNAPSHOT-osgipackage/opendaylight
opendaylight@ODL:~$ sudo ./run.sh
```

Liite 3 Vuomerkinnät

Flows

Node	In Port	DL Src	DL Dst	DL Type	DL Vlan	Vlan PCP	NW Src	NW Dst	ToS Bits	NW Proto	TP Src	TP Dst	Actions	Byte Count	Packet Count	Duration Seconds	Idle Timeout	Priority
OF|00:00:00:00:00:00:00:02	*	*	*	IPv4	*	*	10.0.0.2	*	*	*	*	*	SET_DL_DST = fe:69:e9:d8:5f:f0 OUTPUT = OF|1	490	5	885	0	1
OF|00:00:00:00:00:00:00:02	*	*	*	IPv4	*	*	10.0.0.4	*	*	*	*	*	SET_DL_DST = 2e:aa:71:c0:5c:6b OUTPUT = OF|2	490	5	885	0	1
OF|00:00:00:00:00:00:00:02	*	*	*	IPv4	*	*	10.0.0.3	*	*	*	*	*	OUTPUT = OF|3	392	4	885	0	1
OF|00:00:00:00:00:00:00:02	*	*	*	IPv4	*	*	10.0.0.1	*	*	*	*	*	OUTPUT = OF|3	392	4	885	0	1

1-4 of 4 items

◀ Page 1 of 1 ▶

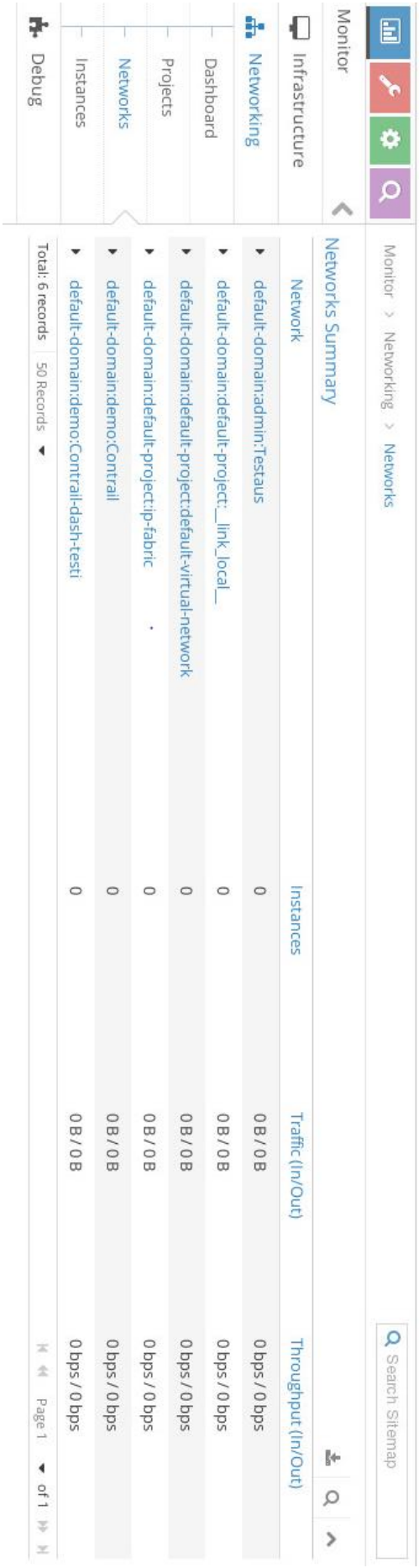
Liite 4 Python-skripti

```
#!/usr/bin/python

from mininet.cli import CLI
from mininet.log import info, setLogLevel
from mininet.net import Mininet
from mininet.node import Host, OVSKernelSwitch, RemoteController
from mininet.topo import Topo

TreeDepth = 2
FanOut = 2
ControllerAddress = ["192.168.0.4", "192.168.0.5"]

class MultiTreeTopo(Topo):
   """Topologian maaritys"""

   def __init__(self):
      Topo.__init__(self)

      self.hostSize = 1
      self.switchSize = 1
      self.treeSwitches = []

      prev = None
      for cidx in range(len(ControllerAddress)):
         switches = []
         self.treeSwitches.append(switches)
         root = self.addTree(switches, TreeDepth, FanOut)
         if prev:
            self.addLink(prev, root)
         prev = root

   def addTree(self, switches, depth, fanout):
      """Add a tree node."""
      if depth > 0:
         node = self.addSwitch('s%u' % self.switchSize)
         self.switchSize += 1
         switches.append(node)
         for i in range(fanout):
            child = self.addTree(switches, depth - 1, fanout)
            self.addLink(node, child)
      else:
         node = self.addHost('h%u' % self.hostSize)
         self.hostSize += 1

      return node

   def start(self, net):
      """Kaynnistetaan kontrollerit ja kytkimet"""

      cidx = 0
      for c in net.controllers:
         info("*** Starting controller: %s\n" % c)
         info("    + Starting switches ... ")
         switches = self.treeSwitches[cidx]
         for sname in switches:
            s = net.getNodeByName(sname)
            info(" %s" % s)
```

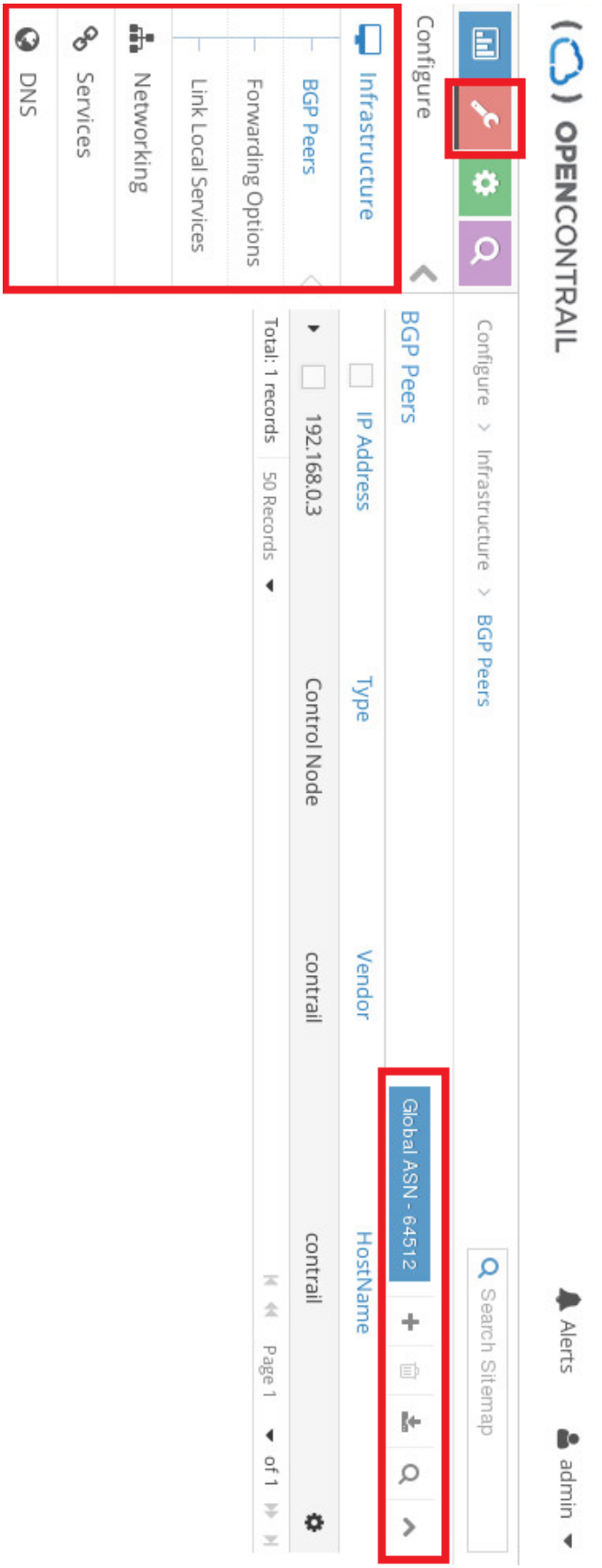
```
            s.start([c])
        cidx += 1
        info("\n")

    self.treeSwitches = None

class MultiTreeNet(Mininet):
  """Emuloitava verkko, jossa kaytetaan ulkoisia kontrollereita"""

  def __init__(self, **args):
    args['topo'] = MultiTreeTopo()
    args['switch'] = OVSKernelSwitch
    args['controller'] = RemoteController
    args['build'] = False
    Mininet.__init__(self, **args)

    idx = 1
    for addr in ControllerAddress:
      name = 'c%d' % idx
      info('*** Kontrolleri: %s (%s)\n' % (name, addr))
      self.addController(name, ip=addr, port=6633)
      idx = idx + 1

  def start(self):
    "Kaynnistetaan kontrollerit ja kytkimet"
    if not self.built:
      self.build()

    self.topo.start(self)

if __name__ == '__main__':
  setLogLevel('info')  # for CLI output
  net = MultiTreeNet()
  net.build()

  print "*** Kaynnistetaan verkko"
  net.start()

  print "***CLI"
  CLI(net)

  print "*** Pysaytetaan verkko"
  net.stop()
```

Liite 5 OpenContrail-hallintaliittymä

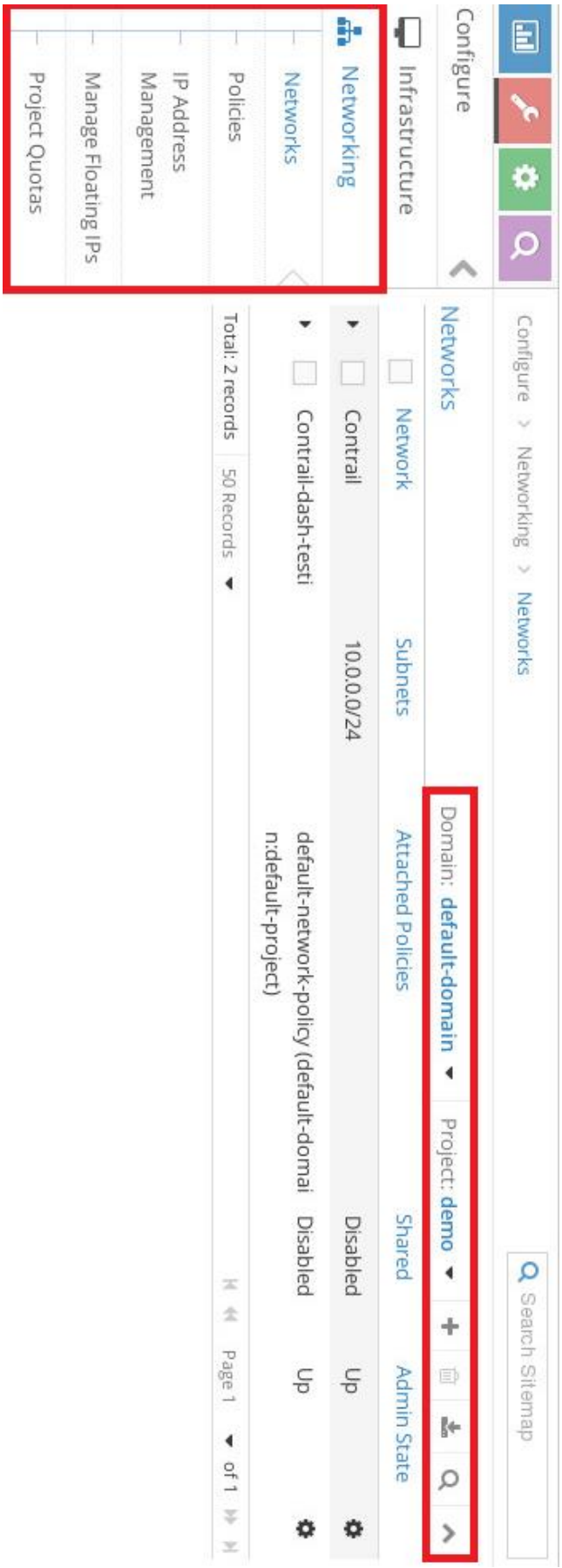
Liite 6 Networking-välilehti



Liite 7 Configure-välilehti



Liite 8 Networks-välilehti



Liite 9 Flavors-välilehti