Juuso Kapulainen

Skaalautuvuuden ongelmat ohjelmisto-ohjatussa tietoverkkoarkkitehtuurissa

Tietotekniikan kandidaatintutkielma

20. joulukuuta 2019

Jyväskylän yliopisto

Informaatioteknologian tiedekunta

Tekijä: Juuso Kapulainen

Yhteystiedot: juuso.kapulainen@gmail.com

Työn nimi: Skaalautuvuuden ongelmat ohjelmisto-ohjatussa tietoverkkoarkkitehtuurissa

Title in English: Scalability challenges in software-defined networking

Työ: Kandidaatintutkielma

Sivumäärä: 23+0

Tiivistelmä: Ohjelmisto-ohjattu tietoverkkoarkkitehtuuri on hiljalleen yleistyvä tietoverkkoarkkitehtuuri, joka pyrkii ratkaisemaan perinteisen arkkitehtuurin skaalautumattomuuden. Tässä tutkielmassa pyritään etsimään ohjelmisto-ohjatussa arkkitehtuurissa piileviä skaalautumisen haasteita. Skaalautumisen haasteet löytyivät arkkitehtuurin eriyttämiltä ohjaus- ja datatasoilta. Datatason haasteet keskittyivät lähinnä laitteiden prosessointikykyyn, kun taas ohjaustason ongelmat keskittyivät sekä prosessointikykyyn että tekniseen implementaatioon. Tutkimus toteaa ohjaustason olevan suurin mahdollinen skaalautumisen mahdollinen pullonkaula. Löytyneiden ongelmakohtien juurisyiden selvittäminen ja niiden lisätutkimus, sekä lopulta niiden selvittäminen ovat erittäin tärkeitä asioita ennen ohjelmisto-ohjatun verkkoarkkitehtuurin yleistymistä.

Avainsanat: Ohjelmisto-ohjattu tietoverkkoarkkitehtuuri, tietoverkot, skaalautuvuus

Abstract: Software-defined networking is slowly emerging network architecture that aims to solve the unscalability of traditional network architecture. This paper aims to find scalability issues in software-defined network architecture. The challenges of scalability were found at the control and data planes that are differentiated by the architecture. Data plane challenges mainly focused on the processing capabilities of the devices, while control plane problems focused on both the processing capabilities and the technical implementation of the network. The study concludes that the control plane issues are the largest bottleneck in scalability. Identifying the root causes of the problems found and further researching them, and finally, solving them, are very important issues before the software-defined network architecture becomes more common.

Keywords: Software-defined networking, networks, scalability

Kuviot

Sisältö

1 Johdanto

Tässä kandidaatintutkielmassa tutkitaan ohjelmisto-ohjatun tietoverkkoarkkitehtuurin (engl. *software-defined networking, SDN*) kykyä skaalautua alati kasvaviin verkkoihin ja niiden vaatimuksiin. Vaikka ohjelmisto-ohjattu tietoverkkoarkkitehtuuri on suunniteltu palvelemaan tietoverkkojen kasvavia vaatimuksia, on se silti herättänyt keskustelua sen kyvystä skaalautua näiden vaatimusten edellyttämällä tavalla. Ohjelmisto-ohjatun tietoverkkoarkkitehtuurin skaalautuvuuteen vaikuttavat muun muassa ohjainten ja reitityslaitteiden prosessointikyky, muistin kapasiteetti, ohjainten sijainti verkossa, ohjainten ja reitityslaitteiden välinen latenssi sekä liikenteen määrä linkeissä (Bhandarkar, Behera ja Khan 2015). Etenkin ohjaustason skaalautumiskyky ohjelmisto-ohjatuissa verkoissa on ongelma johon tulisi kiinnittää erityistä huomiota (Karakus ja Durresi 2017). Toisistaan eriytetyt ohjaus- ja datataso ovat keskeinen määritelmä ohjelmisto-ohjatussa tietoverkkoarkkitehtuurissa.

Perinteisessä nykypäivän tietoverkkoarkkitehtuurissa ohjaus- ja datataso ovat toteutettu yhdistettynä reititystä toteuttavissa verkkolaitteissa. Tietoverkon hallintaa toteutetaan tekemällä muutoksia suoraan reitittävän verkkolaitteen konfiguraatioon. Jos tietoverkkoon halutaan tehdä minkäänlaisia muutoksia, pitää muutokset toteuttaa verkon jokaisen relevantin verkkolaitteen konfiguraatioon. Tämä on osoittautunut rajoitteeksi verkko-operaattoreille, jotka haluavat skaalata verkkonsa vastaamaan muuttuviin liikennevaatimuksiin, mobiililaitteiden käytön lisääntymiseen ja massadatan (engl. *Big data*) vaikutuksiin (Sezer ym. 2013). Ohjelmisto-ohjatussa verkossa ohjaustaso on siirretty pois reitittävistä verkkolaitteista ja se on keskitetty yhteen tai mahdollisesti useampaan ohjelmoitavissa olevaan ohjelmistopohjaiseen ohjaimeen. Eriyttämisen myötä verkon älykkyys sijaitsee keskitetysti ohjaimissa ja verkkolaitteista tulee yksinkertaisia, paketteja edelleenlähettäviä laitteita, joita ohjain pystyy ohjaamaan eri tasojen välisten rajapintojen kautta (Astuto ym. 2014). Tämä mahdollistaa aiempaa kevyemmän konfiguraatioprosessin tietoverkoissa.

Perinteinen tietoverkkoarkkitehtuuri ei enää pysty vastaamaan nykypäivän tarpeisiin. Kreutz ym. 2015 huomauttavatkin, että verkon konfiguroinnin monimutkaisuuden lisäksi verkkoympäristöjen on kestettävä vikojen dynamiikka ja sopeuduttava verkon kuormituksen muutoksiin. Automaattisia uudelleenkonfigurointi- ja vastausmekanismeja ei käytännössä ole ole-

massa nykyisissä IP-verkoissa. Vaadittujen politiikkojen noudattaminen tällaisessa dynaamisessa ympäristössä on siksi erittäin haastavaa. Tämä on synnyttänyt tarpeen uudelle, kasvukykyisemmälle tietoverkkoarkkitehtuurille. Ohjelmisto-ohjattu arkkitehtuuri on kehitetty vastaamaan näihin tarpeisiin, ja siitä on povattu tulevaisuuden korvaajaa perinteiselle verkkoarkkitehtuurille. Vaikka ohjelmisto-ohjattu arkkitehtuuri olisikin perinteisen arkkitehtuurin korvaaja tulevaisuudessa, ei se ole automaattisesti vastaus kaikkiin skaalautuvuuden ongelmiin. Jain ym. 2013 kokemusten mukaan protokollapakettien yhdistämisessä ohjaustasolta datatasolle ja laitteisto-ohjelmoinnin yleiskustannusten pullonkaulat ovat tärkeitä tulevaisuuden työn alueita.

Ohjelmisto-ohjatun verkkoarkkitehtuurin tutkiminen on tällä hetkellä erittäin merkittävässä asemassa, sillä kiinnostus arkkitehtuuria ja sen soveltamista kohtaan on koko ajan kasvussa. Uusi arkkitehtuuri vastaa siihen tarpeeseen, jonka perinteisen arkkitehtuurin puutteet ovat luoneet. Yhä useammat suuret yhtiöt ovat luoneet tai ovat luomassa omaa ohjelmisto-ohjattua tietoverkkoaan ja kehittävät järjestelmiään sen ympärille. Tästäkin syystä olisi erittäin tärkeää tuntea ohjelmisto-ohjattujen verkkojen skaalautuvuuden ongelmat mahdollisimman tarkasti, jotta vältetään samanlainen tilanne kuin perinteisellä arkkitehtuurilla on tällä hetkellä. Usealta verkko-operaattorilta saattaa myös vielä puuttua resursseja ja riittävää tukea ohjelmisto-ohjatun verkkoarkkitehtuurin omaksumiseksi omissa infrastruktuureissaan (Cox ym. 2017). Tästäkin syystä tutkimuksen rooli teknologian kasvuvaiheessa on äärettömän arvokasta.

Tässä tutkielmassa pyritään löytämään niitä skaalautumisen haasteita ja ongelmakohtia, jotka piilevät ohjelmisto-ohjatussa tietoverkkoarkkitehtuurissa. Tutkielma ei ota kantaa haasteiden mahdollisiin ratkaisukeinoihin, vaan pyrkii ainoastaan paikantamaan ja erottelemaan tärkeimmät ja huomattavimmat skaalautuvuuden haasteet. Tutkielman tutkimuskysymykset ovat, millaisia skaalautuvuuden haasteita kohdataan ohjelmisto-ohjatussa tietoverkkoarkkitehtuurissa sekä missä nämä haasteet arkkitehtuurissa sijaitsevat. Tutkielman luvuissa 2 ja 3 määritellään ohjelmisto-ohjattujen tietoverkkojen sekä skaalautuvuuden käsitteet tämän tutkielman kontekstissa sekä perehdytään tarkemmin ohjelmisto-ohjatun arkkitehtuurin rakenteeseen. Määrittelyjen jälkeen luvussa 4 esitellään arkkitehtuurista löytyneet ongelmakohdat ja lopuksi luvussa 5 pohditaan löydettyjen haasteiden merkitystä.

2 Ohjelmisto-ohjatut tietoverkot

Liikenne nykypäivän ja tulevaisuuden tietoverkoissa on huomattavan erilaista kuin se liikenne mitä perinteinen tietoverkkoarkkitehtuuri on suunniteltu palvelemaan. Mobiililaitteiden lisääntymisestä johtuva päätelaitteiden räjähdysmäinen kasvu sekä pilvipalveluiden lisääntyvä käyttö luovat täysin uudenlaisia vaatimuksia tietoliikenteelle. Alasadi ja Al-Raweshidy 2018 esittävät päätelaitteiden määrän ja niiden generoiman liikenteen lisääntymisen, esimerkiksi pyydettäville videoille sekä esineiden internet -tekniikkaan liittyvien päätelaitteiden kasvavan määrän, olevan todellinen ongelma nykyisen verkkoarkkitehtuurin skaalautuvuudelle. Koska yritykset haluavat nyt ketteryyden käyttää sovelluksia, infrastruktuuria ja muita IT-resursseja tarpeidensa mukaisesti, on julkisten ja yksityisten pilvipalveluiden kasvu ollut ennennäkemätöntä ("Software-Defined Networking: The New Norm for Networks" 2012). Perinteinen tietoverkkoarkkitehtuuri ei pysty kehittymään samalla vauhdilla kuin nämä kasvavien liikennemäärien luomat tarpeet. Kyvyttömyys käsitellä kasvavia liikennemääriä riittävän tehokkaasti ei ole kuitenkaan perinteisen arkkitehtuurin ainoa skaalautuvuuden ongelma.

Perinteisen tietoverkkoarkkitehtuurin toinen suuri haaste on sen huomattava hallinnollinen ja ylläpidollinen monimutkaisuus sekä kustannustehottomuus. Perinteinen arkkitehtuuri ei tarjoa minkäänlaista tapaa ohjelmoida tietoverkkoja keskitetysti, mikä johtaa staattisiin sekä joustamattomiin verkkoihin, joista on hankala ylläpitää kattavaa globaalia kuvaa. Perinteiset verkot käsittävät usein suuria määriä toimittajakohtaisia, manuaalisesti konfiguroitavia laitteita, jotka ovat levitetty verkkoihin (Benzekki, El Fergougui ja Elalaoui 2017). Useiden verkkolaitetoimittajien laitteita sisältävät verkot ovat hankalampia ylläpitää, sillä hallinnan järkevä toteuttaminen on työlästä verkoissa joissa verkkolaitteiden konfigurointi sekä laitteiston sisäiset toimintatavat saattavat vaihdella huomattavastikin eri toimittajien välillä. Keskitetyn ohjelmoitavan hallinnan ja globaalin verkkonäkymän puute johtaa siihen, että verkon ylläpitäjien on muutettava korkean tason käytännöt manuaalisesti matalan tason konfigurointikomennoiksi mukautuen muuttuviin verkko-olosuhteisiin (Astuto ym. 2014). Verkkoarkkitehtuurilta vaaditaan nykypäivänä huomattavasti älykkäämpää ja keskitetympää hallintajärjestelmää, joka mahdollistaisi dynaamisemman hallinnan ja vastaisi verkon suori-

tuskykyvaatimuksiin.

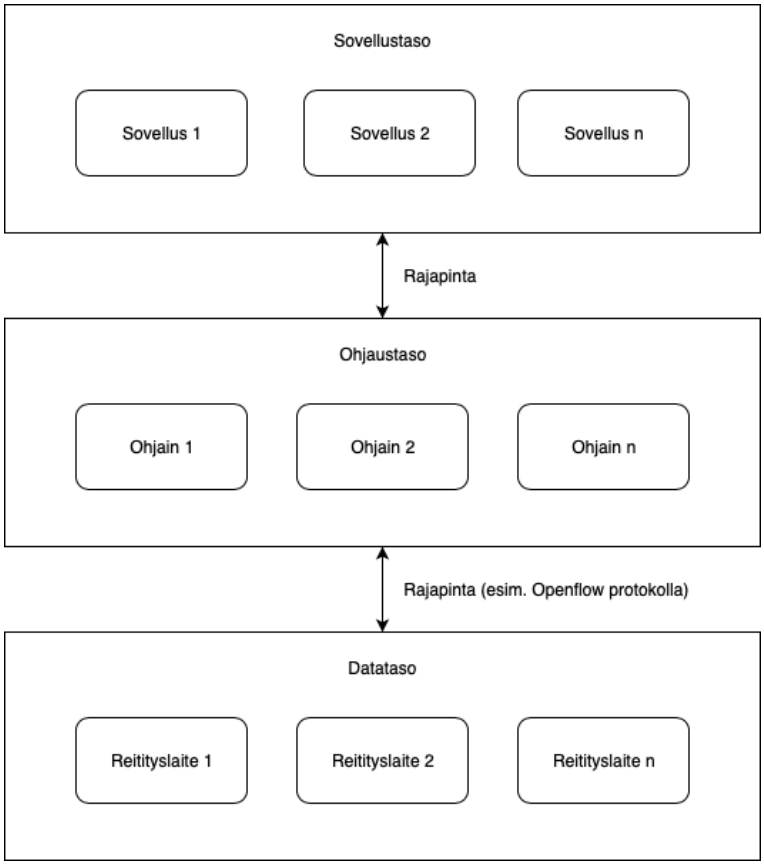
Teknisten ja ylläpidollisten haasteiden lisäksi perinteiset tietoverkot eivät ole kovinkaan kustannustehokkaita. Perinteisen tietoverkon verkkolaitteet, joissa toteutetaan niin hallinta kuin reitityskin, ovat huomattavasti arvokkaampia kuin verkkolaitteet, jotka toteuttavat vain reititystä. Hallinnan poisto reitityslaitteista pienentää sekä valmistuskuluja että ylläpidollisia kuluja huomattavasti. Metzler ja Metzler 2015 kertovat kuinka Google sovelsi ohjelmisto-ohjattua tietoverkkoarkkitehtuuria laajaverkkoonsa (engl. *Wide Area Network, WAN*) ja pystyy nyt lisäämään verkon käyttöä 95 %:iin. Perinteisen laajaverkon käyttöaste on tyypillisesti 60–65 %. Jos käyttöastetta lisätään 95 %:iin, laajaverkon kuukausikustannukset puolittuivat.

Yksinkertaisimmillaan ohjelmisto-ohjattu tietoverkkoarkkitehtuuri on verkkoarkkitehtuuri, joka eriyttää verkon hallinnan reititystä tekevistä verkkolaitteista ja keskittää hallinnan ohjelmoitavissa olevaan ohjaimeen. Tämän myötä koko tietoverkon hallinta pystytään toteuttamaan yhdellä tai useammalla ohjelmoitavalla ohjaimella, joka antaa säännöt reitittäville laitteille. Keskitetyn ohjaimen ohjelmoitavuus tuo mahdollisuuden tehdä muutoksia koko verkon reitityksiin tai lisätä ominaisuuksia verkkoon konfiguroimalla vain yhtä laitetta kaikkien verkkolaitteiden sijaan. Ohjaimella oleva, koko verkon kattava, näkymä luo mahdollisuuden tietoverkon dynaamiselle ohjaamiselle. Ohjaustason eriyttäminen sallii reitittävän infrastruktuurin abstraktoimisen sovelluksille ja verkkopalveluille, jotka täten voivat käsitellä verkkoa loogisena tai virtuaalisena kokonaisuutena (“Software-Defined Networking: The New Norm for Networks” 2012).

2.1 Arkkitehtuurin tasot

Ohjelmisto-ohjatun tietoverkkoarkkitehtuurin rakenne voidaan jakaa kolmeen eri tasoon: sovellustaso (engl. *application plane*), ohjaustaso (engl. *control plane*) ja datataso (engl. *data plane*) (“SDN Architecture Overview” 2013). Tasot ja niiden väliset rajapinnat ovat esitetty hierarkisessa järjestyksessä kuviossa 1.

Datataso koostuu tietoverkon laitteista, jotka toteuttavat tietoverkon liikenteen reitityksen ja edelleenlähetyksen laitteelta toiselle ohjaimelta saatujen sääntöjen mukaisesti. Näitä laitteita ovat muun muassa kytkimet, reitittimet ja tukiasemat. Ohjaimet hallitsevat näitä laitteita da-



Kuvio 1. Tasot ohjelmisto-ohjatussa arkkitehtuurissa.

tatason ja ohjaustason välissä olevan rajapinnan kautta. Kun datatason laitteelle saapuu uutta liikennettä, lähettää laite tätä liikennettä koskevan reitityspyynnön ohjaimelle ja toteuttaa reitityksen sen mukaisesti (Xie ym. 2015).

Ohjelmisto-ohjatun tietoverkkoarkkitehtuurin ohjaustaso käsittää joukon ohjelmistopohjaisia ohjaimia (*controller*), jotka tarjoavat ohjaustoiminnot verkon edelleenlähetyskäyttäytymisen valvomiseksi ja määrittämiseksi datatason ja ohjaustason välissä olevan rajapinnan kautta (Karakus ja Durresi 2017). Ohjaimilla on rajapinnat, joiden kautta ne pystyvät kommunikoimaan niin datatason laitteiden sekä sovellustason loppukäyttäjäsovellusten kanssa. Jos arkkitehtuuri on toteutettu tavalla, jossa on useampi kuin yksi ohjain, vaaditaan myös ohjaimien välinen rajapinta, jotta ohjaimet pystyvät toteuttamaan niiden keskinäistä kommunikaatiota. Jokainen datatason verkkolaite on yhteydessä vähintään yhteen ohjaustason ohjaimista ja saa tältä määritetyn säännöstön mukaiset käskyt tietoverkon liikenteen reitittä-

miselle. Ohjaimet ovat myös yhteydessä sovellustason sovelluksiin, joiden kautta ohjaimia voidaan ohjelmoida ja käsitellä verkkoa ohjaimien kautta.

Sovellustaso koostuu yhdestä tai useammasta loppukäyttäjäsovelluksesta, jotka ovat vuorovaikutuksessa ohjaimien kanssa hyödyntääkseen niiden luomaa abstraktia näkymää verkosta sisäisessä päätöksentekoprosessissaan (Karakus ja Durresi 2017). Sovellustason sovelluksia käytetään luomaan erilaisia säännöstöjä ja toimintatapoja, joiden perusteella ohjaimet toteuttavat verkon hallinnan. Perinteisessä verkossa näiden sovellusten tilalla olisi esimerkiksi palomuurien, kytkinten ja kuormantasaajien omat käyttöliittymäsovellukset ja rajapinnat. Ohjelmisto-ohjatussa tietoverkkoarkkitehtuurissa nämä toteutetaan sovellustasolla luoduissa säännöstöissä, joita ohjaimet ohjaustasolla toteuttavat. Sovellustason sovellukset ovat se rajapinta, jonka kautta verkon ylläpitäjät määrittelevät verkon sisäistä toimintaa.

2.2 Arkkitehtuurin toteuttaminen

Ohjelmisto-ohjattu tietoverkkoarkkitehtuuri voidaan toteuttaa keskitettynä tai hajautettuna arkkitehtuurina. Tämä tarkoittaa hallinnan keskittämistä joko vain yhteen ohjaimeen, joka hallitsee koko verkkoa, tai useampaan ohjaimeen, jotka hallitsevat verkon eri osia säilyttäen kuitenkin globaalin kuvan tietoverkosta.

Keskitetty arkkitehtuuri on yksi ohjain, joka hallinnoi kaikkia kyseisen tietoverkon datatason laitteita. Keskitetty arkkitehtuuri on sopiva toteutustapa pienehköille tietoverkoille, mutta tietoverkon koon kasvaessa ei yksittäisen ohjaimen resurssit riitä palvelemaan kasvavia laitemääriä. Yhden ohjaimen keskitetty arkkitehtuuri luo tietoverkkoon kohdan, joka mahdollistaa ongelmatilanteessa koko verkon rikkoutumisen. Näihin ongelmiin vastaa hajautettu arkkitehtuuri, jossa tietoverkkoa hallinnoidaan usean keskenään kommunikoivan ohjaimen avulla. Ohjaimia voidaan hajauttaa koko verkkoon, jossa kukin ohjain hallitsee tiettyä verkkosegmenttiä, vähentäen yhden ohjaimen vian vaikutusta (Kreutz ym. 2015). Useampi hajautettu ohjain jakaa verkon kuorman tasaisemmin ohjaimien kesken ja pystyy näin palvelemaan suurempaa määrää datatason laitteita kuin keskitetty arkkitehtuuri.

Bhandarkar, Behera ja Khan 2015 jakavat hajautetut arkkitehtuurit kahteen kategoriaan: horisontaalisesti hajautettuun ja vertikaalisesti hajautettuun ohjaustasoon. Horisontaalisesti ha-

jautetussa arkkitehtuurissa jokainen ohjain hallitsee tiettyä osaa verkosta ja kommunikoi kaikkien muiden ohjaimien kanssa tarvittaessa. Vertikaalisesti hajautetussa arkkitehtuurissa on tämän lisäksi toteutettu hierarkia ohjaimien välille ja siihen kuuluu ohjaimia jotka hallitsevat hierarkiassa alempana olevia ohjaimia. Hierarkiassa ylimpänä oleva ohjain kommunikoi sovellustason kanssa ja jakaa täältä saadun datan relevanteille, hierarkiassa alempana oleville ohjaimille. Näin kaikkien tietoverkon ohjaimien ei tarvitse kommunikoida jatkuvasti sekä sovellustason sekä datatason kanssa.

Sekä keskitetyllä, että hajautetuilla arkkitehtuureilla on omat etunsa toisiinsa nähden. Kreutz ym. 2015 mukaan vaikka ensimmäinen vaihtoehto voi tarjota suuren suoritustehon tiheille datakeskuksille, jälkimmäiset voivat olla kestävämpiä erityyppisten loogisten ja fyysisten vikojen varalta.

3 Skaalautuminen

Yleinen määritelmä skaalautumiselle on se, kuinka hyvin jokin tietty ratkaisu johonkin ongelmaan toimii, kun tämän ongelman koko kasvaa. Skaalautuvuus käsitteen käyttö ei ole rajoittunut vain tietoteknisiin ratkaisuihin vaan sitä voidaan käyttää kuvaamaan lähes minkä tahansa ratkaisun toimintakykyä ongelman kasvaessa. Ratkaisujen skaalautuvuutta ei voida näin ollen mitata aina samoin mittarein, vaan skaalautuvuuden mittarit ja ominaisuudet valitaan ongelman mukaan. Tässä tutkielmassa kiinnostuksen kohteena on tietoverkkoarkkitehtuurien skaalautuvuus. Tässä luvussa pyritään määrittelemään mitä skaalautuvuudella tarkoitetaan tietoverkkojen kontekstissa.

Neuman 1994 kirjoittaa, että tietojärjestelmä on skaalautuva, jos se pystyy käsittelemään käyttäjien ja resurssien lisäämistä kärsimättä huomattavia menetyksiä tai lisäämättä hallinnollista monimutkaisuutta. Tämä määritelmä on relevantti myös tietoverkkojen skaalautumisessa. Tietoverkko kasvaa kun siihen lisätään uusia verkko- tai päätelaitteita ja skaalautuvan tietoverkkoarkkitehtuurin on kyettävä mukautumaan tähän ilman, että verkon ylläpitämisestä tai datan liikkumisesta verkossa tulee hankalampaa. Tietoverkon kasvamisena voidaan pitää myös tietoverkon sisäisen liikkennemäärän lisääntymistä ilman varsinaisten verkkolaitteiden lisääntymistä. Tietoverkkoarkkitehtuuri ei ole skaalautuva jos se ei verkon kasvaessa pysty käsittelemään kaikkia saapuvia pyyntöjä tarjoamalla niille samat palvelutakuun kuin ennen kasvua (Yeganeh, Tootoonchian ja Ganjali 2013).

3.1 Vertikaalinen ja horisontaalinen skaalautuminen

Skaalautuminen voidaan jakaa horisontaaliseen ja vertikaaliseen skaalautumiseen (Khare ym. 2012).

Järjestelmä tai tietoverkko skaalautuu horisontaalisesti kun siihen lisätään uusia osakokonaisuuksia, joiden toiminnot ovat identtisiä järjestelmässä jo oleviin osakokonaisuuksiin verrattuna. Tietoverkkojen kontekstissa tämä tarkoittaisi esimerkiksi uusien reitittimien tai SDN-ohjainten lisäämistä verkkoon, jotta verkon synnyttämä kuorma voidaan jakaa useamman laitteen kesken. Horisontaalisesti skaalautumiskykyinen verkko mahdollistaa vähemmän vi-

kaantumisalttiin ympäristön, sillä useampi laite ja työkuorman jakaminen takaavat, että verkosta ei menetetä yhtä suurta osaa jos yksi verkkolaite vikaantuu. Toisaalta mitä useampia laitteita tietoverkossa on, sitä enemmän se vaatii resursseja verkon hallinnalta ja ylläpidolta

Tietoverkon vertikaalisella skaalautumisella tarkoitetaan taas resurssien lisäämistä olemassa oleviin verkkolaitteisiin, jotta pystytään maksimoimaan yhden verkkolaitteen kyvykkyyttä käsitellä sille saapuvaa kuormaa. Käytännössä vertikaalinen skaalautuminen tarkoittaa esimerkiksi keskusmuistin tai prosessorien lukumäärien lisäämistä tietoverkon verkkolaitteisiin. Vertikaalinen skaalautuminen on mahdollista myös ohjelmistoille. Ohjelmisto-ohjatuista tietoverkoista puhuttaessa tällä voidaan tarkoittaa esimerkiksi ohjainohjelmistojen reitityssalgoritmien optimointia. Vertikaalisen skaalautumisen suurimpia etuja on, että järjestelmä voi skaalautua käytännössä rajattomasti vertikaalisesti ilman, että se lisää ylläpidollisia tarpeita.

4 Skaalautuvuuden haasteet

Jo lähdemateriaalia etsiessä kävi ilmi, että etenkin ohjaustasoon kannattaa kiinnittää erityistä huomiota kun kartoitetaan ohjelmisto-ohjatun verkkoarkkitehtuurin skaalautumisen ongelmia. Oletankin, että suuri osa ongelmista löytyy tutkimuksen aikana juuri tältä osa-alueelta. Vaikka pyrin keskittymään paljolti ohjaustasoon ja keskitettyyn hallintaan, ei tutkimuksessa saa unohtaa muita arkkitehtuurin tasoja

Useissa lähdeartikkeleissa on todettu, että suurimmat skaalautumisen ongelmat ohjelmisto-ohjatussa tietoverkkoarkkitehtuurissa liittyvät ohjaustasoon ja keskitetyn hallinnan kykyyn hallita suuria kokonaisuuksia. Etenkin ohjaustason skaalautumisen ongelmat pitää ottaa tutkimuksessa erittäin hyvin huomioon. Yleinen konsensus löydetyissä lähdeartikkeleissa tuntuu olevan, että vaikka kaikki skaalautumisen ongelmat eivät ole uniikkeja vain ohjelmisto-ohjatulle tietoverkkoarkkitehtuurille, on se erittäin tärkeä tutkimusaihe tämän arkkitehtuurin suosion kasvaessa.

Seuraavissa alaluvuissa esittelemme ohjelmisto-ohjatun tietoverkkoarkkitehtuurin eri tasoilta löytyneitä skaalautumisen ongelmia. Tarkoituksena ei ole löytää vastauksia näihin ongelmiin ja niiden herättämiin kysymyksiin, vaan vain osoittaa ne ongelmakohdat jotka on syytä ottaa huomioon ohjelmisto-ohjatun tietoverkkoarkkitehtuurin skaalautumisessa.

4.1 Ohjaustason ongelmat

Keskitetty toteutus ohjelmisto-ohjatulle tietoverkkoarkkitehtuurille olisi hallinnollisesta näkökulmasta helpoin mahdollinen tapa toteuttaa verkko. Keskitetty arkkitehtuuri ei synnytä ohjainten välistä liikennettä ja globaali kuva verkosta on helppo säilyttää ilman lisääntyvää verkon liikennettä. Kuten aiemmin todettiin, keskitetty arkkitehtuuri onkin hyvä ratkaisu kun verkon koko pysyy pienenä. Verkon kasvaessa yksittäistä ohjainta voidaan yrittää skaalata vertikaalisesti, mutta vastaan tulee väistämättä raja, jossa ohjain ylikuormittuu käsitellessään yhä useampaa pyyntöä ja pyrkiessään samanaikaisesti saavuttamaan kaikille saman suoritustakuun (Bannour, Souihi ja Mellouk 2017). Näin yksittäisestä ohjaimesta tulee koko arkkitehtuurin skaalautumisen estävä pullonkaula.

Bannour, Souihi ja Mellouk 2017 mukaan ohjaustason skaalautuvuutta arvioidaan yleensä sillä, montako virtauspyyntöä käsitellään sekunnissa, sekä sillä, miten suuri viive on virtauspyyntöön vastaamisessa.

NOX, joka on yksi aikaisimmista ohjaimista ohjelmisto-ohjatussa arkkitehtuurissa, pystyy käsittelemään vähintään 30 000:tta uutta virtauspyyntöä sekunnissa pitäen samalla alle 10 ms:n palveluajan virtauspyynnölle (Tavakoli ym. 2009). Tämä käsittelynopeus on nykypäivään mennessä toki kasvanut, mutta se ei siltikään riitä palvelemaan suurien verkkojen tarpeita, jotka saattavat olla sadoistatuhansista miljooniin virtauspyyntöä sekunnissa. Kun ohjain hallitsee verkkoliikennettä suuressa verkossa, saattaa kasvava verkkoliikenne ylittää ohjaimen suorituskyvylliset ominaisuudet ja aiheuttaa siten skaalautuvuus- ja joustavuusongelmia (Benzekki, El Fergougui ja Elalaoui 2017).

Hajautettuna toteutetun ohjelmisto-ohjatun tietoverkkoarkkitehtuurin skaalautuvuus on riippuvainen ohjaimien määrästä ja sijoittelusta (Hu ym. 2018). Jos tietoverkon ohjaimet ovat aseteltu verkkoon epäloogisesti, aiheuttaa se tietoverkon liikennekuorman jakaantumisen eri ohjaimille epätasaisesti. Tämä saattaa aiheuttaa lisääntynyttä viivettä kuormittuneemmille ohjaimille, ja näin heikentää ohjaustason kykyä palvella kaikkia saapuvia pyyntöjä samalla palvelutakuulla. Myös jos ohjain on sijoitettu fyysisesti kauas sen ohjaamista reitityslaitteista, lisää se viivettä reitityslaitteen ja ohjaimen välisessä kommunikaatiossa. Liu ja Li 2015 huomauttavan tämän olevan erityisen relevanttia laajaverkoissa, koska ne saattavat kattaa suuren alueen maantieteellisesti, jolloin keskitetyn ohjaustason tulisi ottaa huomioon etenemisviiveet, jotka voivat merkittävästi heikentää vasteaikaa. Reitityslaitteen ja ohjaimen välisen kommunikaation viive lisää mahdollisesti myös reitityslaitteen läpi kulkevan liikenteen viivettä sekä viivettä reitityslaitteen reaaliaikaisessa konfiguroinnissa ohjaimen toimesta. Tämä puolestaan voi johtaa ruuhkautumiseen sekä ohjaustasolla että datatasolla ja pidempään vian kestoaikaan, mikä heikentää ohjaimen skaalautuvuutta merkittävästi (Karakus ja Durresi 2017).

Useimmat hajautetun arkkitehtuurin toteutukset tarjoavat heikon johdonmukaisuuden semantiikan, mikä tarkoittaa, että tiettyyn tietoverkon laitteeseen tehty tietojen päivitys tullaan vasta sen jälkeen tekemään kaikkiin muihin verkon ohjaimiin (Kreutz ym. 2015). Tämä tarkoittaa, että on olemassa ajanjakso, jonka kuluessa tietoverkon eri ohjaimet tai reitityslait-

teet saattavat lukea kahta eri reitityssäännöstöä liikenteelle, jolla pitäisi olla vain yksi päivitetty reitityssäännöstö. Tämä saattaa aiheuttaa tietoverkon liikenteen katoamista ja liikenteen tuottaneen sovelluksen virheellisen toiminnan.

Jatkuvien päivitysten tekeminen verkon ohjaimiin on releventtia myös silloin, kun ohjaimissa pyritään säilyttämään tarkka globaali kuva tietoverkosta. Tämä synnyttää huomattavan määrän niin ohjainten keskinäistä kuin ohjainten ja reitityslaitteiden välistä liikennettä. Verkon kasvaessa näiden monitorointiviestien vähimmäismäärä verkossa kasvaa eksponentiaalisesti. Monitorointiviestien vähimmäismäärää kuvaa $(O(N^2)$, missä N on verkon halkaisija (Liu ja Li 2015).

4.2 Datatason ongelmat

Ohjelmisto-ohjatun tietoverkkoarkkitehtuurin datataso on pohjimmiltaan “tyhmä“ kerros, joka ainoastaan suorittaa, ohjaustason tuottamat, liikenteen edelleenlähetykseen liittyvät komennot. Tästä johtuen datatason skaalautumisen ongelmat liittyvät pääasiassa laitteiston suorituskykyyn käsitellä liikennettä riittävästi. Datatason skaalautuvuutta pyritäänkin parantamaan muuttamalla verkon tai edelleenlähetyslaitteen laitteistoa ja ohjelmistoa, mikä tarkoittaa, että se riippuu muistin koosta ja prosessorin nopeudesta (Bhandarkar, Behera ja Khan 2015).

Kuten todettu, datataso edelleenlähettää sille saapuvan liikenteen ohjaimelta saatujen ohjeiden mukaisesti. Ohjaimelta saadut ohjeet tallennetaan virtaustauluun reitittävän laitteen sisäiseen muistiin. Jos muistista ei löydy edelleenlähetysohjeita saapuvalle liikenteelle, pyydetään vastuussa olevalta ohjaimelta reitytysohjeet liikenteelle. Mitä suurempi määrä liikennettä kulkee reitittävän laitteen läpi, sitä enemmän sen tarvitsee tallentaa edelleenlähetysohjeita sisäiseen muistiinsa. Yksi käytännön haaste on toimittaa kytkimille riittävän suuret ja tehokkaat virtaustaulut sääntöjen tallentamiseksi (Kreutz ym. 2015). Virtaustaulujen tehokkuutta rajoittavat laitteiston muistin koko sekä prosessointikyky. Reitityslaitteet, jotka kykenevät tallentamaan riittävän suuria virtaustaulukkoja ja etsimään taulukoista ohjeet riittävällä nopeudelle palvellakseen suuria liikennemääriä, ovat hankalia ja kalliita toteuttaa.

Jatkuva edelleenlähetysohjeiden kysyminen ohjaimelta puolestaan aiheuttaa liikenteen kär-

simisen kyselyviiveestä (Liu ja Li 2015). Reitityslaitteelle saapuva liikenne joutuu odottamaan kunnes laite on pyytänyt ja saanut edelleenlähetysohjeet ohjaimelta, mikä saattaa aiheuttaa huomattavaa viivettä alkuperäiseen yhteyteen. Tämä käytäntö aiheuttaa huomattavaa latenssia yhteyksiin, jos kyselyt joudutaan tekemään joka kerta kun liikennettä saapuu reitityslaitteelle. Koska ulkopuolinen yksikkö konfiguroi reaaliaikaisesti kytkimien virtaustaulukoita, syntyy myös reitityslaitteiden ja ohjaimien välisille yhteyksille huomattavaa latenssia etenkin suurissa verkoissa, joissa ohjain joutuu käsittelemään potentiaalisesti jopa miljoonia virtauspyyntöjä sekunnissa (Kreutz ym. 2015).

Liu ja Li 2015 nostavat vielä yhdeksi tärkeäksi skaalautumisen ongelmaksi datatasolla sen, että verkon päivityksessä verkon massiivisia pakettihäviöitä ja stokastisia tiloja tapahtuu ilman asianmukaisia mekanismeja pakettikohtaisen johdonmukaisuuden varmistamiseksi käytäntöpäivitysten aikana. Virheellisen reitityksen lisäksi päivitysviestien määrä ja niiden aiheuttama viive voivat kasvaa lineaarisesti verkon skaalautuessa. Pakettihäviöt syntyvät kun uudet säännöstöt asentuvat reitityslaitteisiin eri ajanhetkinä, mikä sallii epäyhtenäisten reitityssääntöjen olemassaolon hetkellisesti. Kuten aiemmin todettiin tämä ongelma koskee sekä data- ja ohjaustasoa.

5 Yhteenveto

Tämän tutkielman tavoitteena oli löytää ja osoittaa ohjelmisto-ohjatussa tietoverkkoarkkitehtuurissa piilevät skaalautumisen haasteet ja ongelmakohdat. Tutkielmassa onnistuttiin löytämään merkittävimmät haasteet sekä paikantamaan niiden sijainnit ja syyt arkkitehtuurin rakenteessa.

Tutkielmassa huomattiin, että skaalautuvuuden ongelmat keskittyvät ohjelmisto-ohjatun arkkitehtuurin eriytettyihin ohjaus- ja datatasoihin. Suurimmat ongelmat keskittyivät kuitenkin pääasiassa ohjaustasolle ja sen ohjainohjelmistoihin. Ohjaustasolta löytyneet haasteet olivat ohjainten epälooginen määrä ja sijoittelu, ohjainten prosessointikykyyn liittyvät ongelmat, yhtenäisen verkkokuvan säilyttäminen sekä ohjainten keskinäisen kommunikaation välisen liikenteen aiheuttama ruuhkautuminen. Ohjaustason ongelmien voidaan katsoa liittyvän paljolti siihen millä tavalla ohjelmisto-ohjattu tietoverkko on implementoitu. Tukielmassa kuitenkin löydettiin myös teknisiä rajoitteita esimerkiksi ohjainohjelmistoteknologioissa jotka eivät ole millään lailla riippuvaisia verkon implementaatiosta.

Datatason skaalautuvuuden haasteet liittyivät, tason rakenteesta johtuen, pääasiassa datatason verkkolaitteiden liikenteen prosessointikykyyn. Kuitenkin myös ohjaustason ja datatason välisen liiallisen hallintaliikenteen aiheuttama viive voidaan laskea olevan ainakin osittain datatason aiheuttama haaste ohjelmisto-ohjatun tietoverkkoarkkitehtuurin skaalautuvuudelle. Datatason prosessointikykyä koskevia skaalautumisen haasteita voidaan pitää relevantteina myös perinteisessä arkkitehtuurissa.

Skaalautumisen haasteiden yhtäläisyydet tietoverkkoarkkitehtuureissa on yleisesti huomioitu seikka, jonka mukaan skaalautumisen haasteet eivät ole fundamentaalisesti sen erilaisempia ohjelmisto-ohjatuissa verkoissa kuin perinteisessä arkkitehtuurissakaan. Tässä tutkielmassa löydetyt haasteet kuitenkin ovat rakenteeltaan selkeästi erilaisia kuin perinteisen arkkitehtuurin haasteet ja vaativat uusia ja erilaisempia ratkaisuja kuin aiemmin.

Tutkielma osoittaa, että tarve ohjelmisto-ohjatun tietoverkkoarkkitehtuurin ja sen skaalautuvuuden tutkimukselle ja kehittämiselle on todellinen. Arkkitehtuuri sisältää vielä useita ratkaisemattomia skaalautumisen haasteita jotka haittaavat arkkitehtuurin laajaa käyttöönottoa.

Tämän tutkimuksen pohjalta olisi erittäin tärkeää tutkia haasteita ja niiden juurisyitä syvemmin ja kehittää ratkaisuja tässäkin tutkielmassa löytyneisiin verkkoarkkitehtuurin skaalautuvuuden ongelmakohtiin. Tutkimusta tulisi myös toteuttaa mahdollisimman paljon aidoissa verkkoympäristöissä eikä vain teoriapohjaisesti. Aidoissa verkkoympäristöissä voidaan esimerkiksi laitteiden performanssia ja vikasietoisuutta mitata huomattavasti tarkemmin kuin ainoastaan tietoverkkojen teoriapohjaa tutkimalla.

Lähteet

Alasadi, E., ja H. Al-Raweshidy. 2018. “OLC: Open-Level Control Plane Architecture for Providing Better Scalability in an SDN Network”. *IEEE Access* 6:34567–34581. ISSN: 2169-3536. doi:10.1109/ACCESS.2018.2848638.

Astuto, Bruno, Marc Mendonça, Xuan Nguyen, Katia Obraczka ja Thierry Turletti. 2014. “A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks”. *IEEE Communications Surveys and Tutorials* 16 (3): 1617–1634.

Bannour, Fetia, Sami Souihi ja Abdelhamid Mellouk. 2017. “Distributed SDN Control: Survey, Taxonomy and Challenges”. *IEEE Communications Surveys & Tutorials* (joulukuu): 1–1. doi:10.1109/COMST.2017.2782482.

Benzekki, Kamal, Abdeslam El Fergougui ja Abdelbaki Elalaoui. 2017. “Software-defined networking (SDN): A survey”. *Security and Communication Networks* (helmikuu). doi:10.1002/sec.1737.

Bhandarkar, Smriti, Gyanamudra Behera ja Kotla Khan. 2015. “Scalability Issues in Software Defined Network (SDN): A Survey”. *Advances in Computer Science and Information Technology* 2 (1): 81–85.

Cox, J. H., J. Chung, S. Donovan, J. Ivey, R. J. Clark, G. Riley ja H. L. Owen. 2017. “Advancing Software-Defined Networks: A Survey”. *IEEE Access* 5:25487–25526.

Hu, T., Z. Guo, P. Yi, T. Baker ja J. Lan. 2018. “Multi-controller Based Software-Defined Networking: A Survey”. *IEEE Access* 6:15980–15996. doi:10.1109/ACCESS.2018.2814738.

Jain, Sushant, Alok Kumar, Subhasree Mandal, Joon Ong, Leon Poutievski, Arjun Singh, Subbaiah Venkata, Jim Wanderer, Junlan Zhou, Min Zhu ym. 2013. “B4: Experience with a globally-deployed software defined WAN”. Teoksessa *ACM SIGCOMM Computer Communication Review,* 43:3–14. 4. ACM.

Karakus, Murat, ja Arjan Durresi. 2017. “A survey: Control plane scalability issues and approaches in Software-Defined Networking (SDN)”. *Computer Networks* 112:279–293. ISSN: 1389–1286. doi:https://doi.org/10.1016/j.comnet.2016.11.017. http://www.sciencedirect.com/science/article/pii/S138912861630411X.

Khare, Amulya, Yipeng Huang, Hung Doan ja Mohit Kanwal. 2012. “Scalability”. *A Fresh Graduate’s Guide to Software Development Tools and Technologies* (huhtikuu).

Kreutz, D., F. M. V. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky ja S. Uhlig. 2015. “Software-Defined Networking: A Comprehensive Survey”. *Proceedings of the IEEE* 103 (1): 14–76.

Liu, Shuhao, ja Baochun Li. 2015. “On scaling software-Defined Networking in wide-area networks”. *Tsinghua Science and Technology* 20, numero 3 (kesäkuu): 221–232.

Metzler, Jim, ja Ashton Metzler. 2015. “The 2015 Guide to SDN and NFV”.

Neuman, B. Clifford. 1994. “Scale in Distributed Systems”. *Readings in Distributed Computing Systems:* 463–489.

“Software-Defined Networking: The New Norm for Networks”. 2012. *ONF White Papers.* https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf.

“SDN Architecture Overview”. 2013. *ONF White Papers* (joulukuu). http://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/SDN-architecture-overview-1.0.pdf.

Sezer, S., S. Scott-Hayward, P. K. Chouhan, B. Fraser, D. Lake, J. Finnegan, N. Viljoen, M. Miller ja N. Rao. 2013. “Are we ready for SDN? Implementation challenges for software-defined networks”. *IEEE Communications Magazine* 51 (7): 36–43.

Tavakoli, Arsalan, Martin Casado, Teemu Koponen ja Scott Shenker. 2009. “Applying NOX to the Datacenter”. https://www2.cs.duke.edu/courses/cps296.4/fall13/838-CloudPapers/hotnets2009-final103.pdf.

Xie, Junjie, Deke Guo, Zhiyao Hu, Ting Qu ja Pin Lv. 2015. “Control plane of software defined networks: A survey”. *Computer Communications* 67:1–10. ISSN: 0140-3664. doi:https://doi.org/10.1016/j.comcom.2015.06.004. http://www.sciencedirect.com/science/article/pii/S0140366415002200.

Yeganeh, S. H., A. Tootoonchian ja Y. Ganjali. 2013. “On scalability of software-defined networking”. *IEEE Communications Magazine* 51, numero 2 (helmikuu): 136–141. doi:10.1109/MCOM.2013.6461198.