Reunalaskenta arkkitehtuurit

Lauri Vene

Pro gradu -tutkielma
UNIVERSITY OF HELSINKI
Department of Computer Science

Helsinki, January 31, 2018

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Laitos — Institution — Department
Faculty of Science	Department of Computer Science

Tekijä — Författare — Author
Lauri Vene

Työn nimi — Arbetets titel — Title

Reunalaskenta arkkitehtuurit

Oppiaine — Läroämne — Subject
Computer Science

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
Pro gradu -tutkielma	January 31, 2018	10

Tiivistelmä — Referat — Abstract

Tiivistelmä.

Avainsanat — Nyckelord — Keywords
reuna, pilvi, tietojenkäsittelytiede

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — Övriga uppgifter — Additional information

Contents

1 Huomioita **1**

2 Johdanto **1**

3 Reunalaskennan perusteet **1**
3.1 Motivaatio 1
3.2 2

4 Suurimmat osatoimijat **4**
4.1 eNodeB 4
4.2 Asiakaslaite 4
4.3 Edge cloud 4
4.4 Cloud 4
4.5 Reunasolmu 4

5 Ominaisuudet **4**
5.1 Etälaskenta 5
5.2 Etäinstanssi 6
5.3 Migraatio 6
5.4 Integraatio mobiiliverkkoihin 7
5.5 IP-verkko 7
5.6 Kommunikaatio asiakaslaitteen (UE) ja Reunasolmun kanssa 7
5.7 Hallinta 7

6 Reunan rakenne **7**

7 Esitetyt ratkaisut **8**
7.1 Cloudlet 8
7.2 FMC – Follow me cloud 9
7.3 Small Cell Cloud (SCeNB) 9
7.4 MobiScud ja SMORE 9
7.5 Concert 9
7.6 ETSI reference architecture 9
7.7 Yhteenveto 9

Lähteet **9**

1 Huomioita

- reuna
- reunasolmu

Tutkielmassa puhutaan mobiililaitteesta, mutta monet toiminnallisuudet ovat sovellettavissa kaikenlaisiin mobiileihin laitteisiin, esimerkiksi kannettaviin tai älykkäisiin ajoneuvoihin.

Listan Mobile Computingin neljästä pääasiallisesta rajoitteesta.
Unpredictable variation in network quality, lowered trust and robustness of mobile elements, limitations on local resources imposed by weight and size constraints, and concern for battery power consumption [13]

[13]M. Satyanarayanan, "Fundamental Challenges in Mobile Computing," Proc. 15th ACM Symp. Principles of Dist. Comp., Philadelphia, PA, May, 1996

2 Johdanto

3 Reunalaskennan perusteet

3.1 Motivaatio

Reunalaskennan ideana on täydentää ja avustaa resurssiköyhiä asiakaslaitteita. Mobiililaitteisiin voitaisiin teoriassa lisätä enemmän resursseja, mutta ne tulisivat kannettavuuden ja käyttöajan kustannuksella. Avustamiseksi voidaan ajatella esimerkiksi tilanne, jossa akkuvirtaa säästääkseen, mobiililaite välittää resurssi intensiivisen laskutoimituksen reunasolmulle laskettavaksi. Reunasolmun voi ajatella kevennetyksi versioksi palvelimesta joten sen rajoitteet ovat hyvin erilaiset kuin asiakaslaitteen. Täydentävä toiminta reunalaskennan avulla tarkoittaa asiakaslaitteen resurssin puutetta suoriutua jostakin tehtävästä. Esimerkiksi muistin riittämättömyys kuvankäsittelyyn. Tällöin voitaisiin esimerkiksi ottaa etäkäyttöyhteys reunasolmuun jolla käsittely tehdään. Asiakaslaitteelle jäisi tässä tilanteessa tehtäväksi ainoastaan esittää reunasolmun tilaa käyttäjälle.

Satyanarayana [Satyanarayanan, 2001] esitti artikkelissaan pervasive computing esimerkkejä jokapaikan tietotekniikasta. Ympäristöön sijoitettujen laitteiden yhteistoiminnan avulla, asiakkaalle voidaan tarjota parempaa ja täsmällisempää palvelua. Yhtenä palvelun laadun ehtona on kyky ennakoida asiakkaan toimintaa. Nykyään mobiililaitteilla on mahdollista hyödyntää langattomia yhteyksiä. Suurin osa palvelinresursseista ja palveluiden tuottamiseen käytettävästä tiedosta on keskittyneenä pilveen. Käytännössä minkä tahansa palvelun käyttäminen mobiililaitteella edellyttää yhteyttä näihin pilvipalveluihin. Pilvipalveluiden ylläpitäminen

Toinen painopiste on siinä että tieto seuraa käyttäjää. Esimerkiksi pöytäkoneelta mobiililaitteeseen.(Satyanarayanan, 2001). Cyber foraging, on termi jota käytetään kuvaamaan paradigmaa jossa laite etsii ympäristöstä hyödynnettävää tietoa ja avustajia/korvikkeita. Avustajan (surrogate) rooli on täydentää lähtökohtaisesti resurssirajallista laitetta, esimerkiksi suorittamalla laskentaa, jotta asiakaslaite voisi esimerkiksi säästää akkua. Tähän toimintaan liittyy kuitenkin useita haasteita. Esimerkiksi kuinka asiakaslaite löytää avustajan? Mitäs jos avustaja on ruuhkautunut? Kuinka avustaja alustetaan ja kauanko siinä kestää? Nämä ovat keskeisiä kysymyksiä myös reunalaskennassa. Vastuun jakaminen asiakaslaitteen sekä reunanklusterin välillä on riippuvainen siitä, kuinka paljon avustusta asiakaslaite tarvitsee. Toiseen ääripäähän vietynä asiakaslaite on niin sanottu kevyt asiakaspääte (thin client), jolla ei olisi resursseja juurikaan mihinkään. Tällainen asiakaslaite joutuisi jakamaan kaiken laskennan eteenpäin avustajalle. Tämän kaltainen asiakaslaite olisi riippuvainen reunan mahdollisuuksista suorittaa palveluiden vaatimia toiminnallisuuksia. Seuraavana askeleena kohti itsenäisempää suoriutumista olisi asiakaslaite, joka pystyy osittain tarjoamaan käyttäjälle palveuita. Tämä laite tarvisi reunaklusterilta avustusta ainoastaan joissain tapauksissa. Viimeisenä toimijana olisi kokonaan itsenäinen asiakaslaite, joka tarvitsisi reunalta ainoastaan palveluita täydentäviä ominaisuuksia. Tämä laite saattaisia turvautua reunalaskentaan esimerkiksi jos akku on vähissä, tai laitteella itsellään ei ole kaikea tarvittavaa tietoa laskennan suorittamiseksi.

3.2

- Mistä reunalaskenta koostuu? (Suurimmat toimijat, keskeisimmät toiminnot)
- Mikä on MEC?
- Reunalaskenta vai reunapalvelu?
- MCC vai MEC?
- Pilvi vai palvelinkeskus
- Mitä reunalaskenta on?
- Miksei siirretä laskentaa pilveen?
- Mitkä ovat mobiilin ongelmat nykyisellään?
- Mitkä ovat reunalaskennan haasteet?

Pilvipalvleulla tarkoitetaan palvelua, joka sijaitsee internetissä. Palvelut tarjotaan käyttäjälle verkkoyhteyden välityksellä. Pilvipalvelut sisältävät

usein suuria määriä laskenta- ja tallennusresursseja. Palvelut ovat myös usein runkoverkossa kiinni, jolloin niiden voidaan ajatella sijaitsevan internetin "keskustassa". Yleensä palveluita ylläpidetään keskitettyinä korkeintaan muutamaan eri konesaliin. Pilvipalvelun palvelun kohde on usein asiakaslaite (UE, user equipment), joka sijaitsee internet-topologian näkökulmasta lehtisolmussa. Etäisyys

Reunalaskenta on yksi hajautetun laskennan muoto jossa

Mobiililaitteiden yleisiä ominaisuuksia ovat resurssien vähyys ja akkuvirran rajallisuus. Mobiililaitteiden käyttö on myös usein riippuvaista langattomista verkkoyhteyksistä. Palveluiden toiminta mobiililaitteilla on siis riippuvainen näiden kolmen ominaisuuden asettamista rajoista. Laskentaresurssien lisääminen johtaisin lyhyempään käyttöaikaan akkuvirralla. Suurempi akku mahdollistaa pidemmän käyttöajan, mutta se tekisi laitteesta suuremman. Akun kokoa ja laskentaresurssien määrää pyritäänkin tasapainottamaan. Voidaan pyrkiä minimoimaan laitteen virrankulutus esimerkiksi laittamalla laitteeseen heikkotehoisempi suoritin. Tämä näkyy siinä, millaisia palveluita mobiililaitteella voidaan tarjota. Esimerkiksi kuvankäsittelyä tai muuta raskaampaa laskentaa vaativaa toiminnallisuutta ei voida tällaisella laitteella tehdä. Seuraava vaihtoehto olisi lähettää laskentaa tehokkaammille laitteille pilveen. Laskennan siirtäminen vie aikaa ja tällöin ei voida tarjota kovin reaaliaikaisia palveluita. Siirrettyyn laskentaan kuluva aika koostuu pääasiassa verkon viiveestä, siirrosta ja itse laskennasta. Kokonaisuudessa siirtämiseen ja laskentaan kuluvan ajan määrään vaikuttaa pilven sijainti, verkon ruuhkaisuus, verkon kapasiteetti, sekä käytössä olevan laskentakapasiteetin määrä. Reunalaskenta on konsepti, jonka avulla laskenta voidaan tuoda lähemmäksi käyttäjää.

Reunalaskennassa (MEC,Mobile Edge Computing vai MCC Mobile Cloud Computing?) on tarkoitus tuoda palvelinresursseja lähemmäksi käyttäjää ? reunalle. Tässä kontekstissa reunalla tarkoitetaan käyttäjän ja pilven väliin jäävää tilaa. TCP/IP-mallissa sovellustasolla olevia toimintoja ei siis esiinny tällä välillä. Reunalaskenta siis mahdollistaa palveluiden tuottamisen lähempänä käyttäjää. Lähempänä on hieman harhaan johtava termi, koska mikä tahansa pilveä lähempänä oleva palvelu on lähempänä, eikä siis välttämättä konkreettisesti lähellä.

Reunalaskennalle ei vielä ole olemassa kokonaisvaltaista arkkitehtuuria. Ongelmakentän voi jakaa karkeasti kahteen osaan. Fyysiseen arkkitehtuuriin ja sovellustason arkkitehtuuriin. Nämä ovat toisistaan riippuvaisia. Arkkitehtuuriratkaisut ovat riippuvaisia tarjottavista palveluista. Toiset arkkitehtuuriratkaisut tukevat toisia palveluita paremmin kuin toiset, kompromisseilta on siis vaikea välttyä.

4 Suurimmat osatoimijat

4.1 eNodeB

eNodeB on televerkossa tukiasemakontekstin kokonaisuus. Sen tehtäviin kuuluu teleradiotoiminnan järjestäminen. Mukaan lukien puhelinkeskuksien kanssa kommunikointi.

4.2 Asiakaslaite

UE (User Equipment) on yleisnimitys asiakaslaitteelle, joka hyödyntää pilven (cloud) ja reunan (edge) palveluita tietoliikenneyhteyksien avulla. Usein käyttäjälaitteen esimerkkinä toimii jokin mobiililaite kuten puhelin, mutta myös esimerkiksi auto.

4.3 Edge cloud

Edge cloud on yleisnimitys reunapalveluiden tarjoamiseen tarkoitettuille toimijoille. Riippuen arkkitehtuurista reunapilvi koostuu yksittäisistä toimijoista tai reunaa lähellä olevista klustereista. Edgen on mahdollista tarjota palveluita pienemmillä viiveillä ja suuremmilla tiedonsiirtokapasiteeteillä verrattuna perinteisiin pilvipalveluihin.

4.4 Cloud

4.5 Reunasolmu

Tässä tutkielmassa reunasolmulla viitataan yksittäiseen reunalla sijaitsevaan palvelinklusteriin, joka tuottaa asiakkaille reunapalveluita. Edge Cloud koostuu reunasolmujen joukosta.

Asiakaskohtaiselle reunainstanssille ei ole mitään vakiintunutta nimeä. Cloudlet on yksi ehdotettu toteutustekniikka tällaiselle asiakaskohtaiselle reunalla sijaitsevalle virtuaali-instanssille [Satyanarayanan et al., 2009].

5 Ominaisuudet

Reunalaskennan keskeisin tarkoitus on laskennan siirtäminen reunalla toimivalle reunalaskentaklusterille. Reunaa voidaan lähestyä pilven ja käyttäjälaitteiden puolelta. Käyttäjälaitteiden, kuten älypuhelimien laskentateho on suhteellisen heikkoa, lisäksi ne ovat akkuvirrasta riippuvaisia. Mobiililaitteen käyttöajan pidentämiseksi voidaan pyrkiä tekemään mahdollisimman vähän akkuakuluttavaa laskentaa paikallisesti, siirtämällä sitä reunalaskentaklusterille (Etsi se lähde jossa verrataan tietokoneita ja mobiililaitteita - eroa oli yhden kertaluokan verran). Esimerkiksi verkkoliikenne mobiililaitteen ja kohdepalvelimen välillä on merkittävä viive toisi reunaklusteri palvelun

huomattavasti lähemmäksi ja pienentäisi viivettä palvelussa. Viiveen pienenemisen seurauksena monet reaaliaikaisuuttaa tai nopeaa reagointia vaativat palvelut ovat mahdollisia. Lisäksi verkon viiveestä tai ruuhkasta johtuen, mobiililaitteella on usein huomattavasti nopeampi yhteys fyysisesti lähelle itseään verra

Lisäksi pilven tai konesalien suunnasta asiaa lähestyttäessä runkoverkko tukkeutuu. Siirtämällä osan palveluvastuusta reunalle, runkoverkon rasitteen tulisi ainakin periaatteessa pienentyä.

5.1 Etälaskenta

Etälaskennan toteuttamiseksi tarvitaan vastaukset seuraaviin kysymyksiin. Mitä siirretään ja minne siirretään?

MOCAssa oli selitetty kuinka muodostetaan yhteys in-network cloudiin.

Etälaskenta voidaan karkeasti jakaa kahteen tyyppiin: Binääriseen ja osittaiseen [Mao et al., 2017]. *Onkohan ok lainata surveytä?*. Binäärisessä ohjelmasta suoritetaan selkeitä kokonaisuuksia joko reunasolmulla tai asiakaslaitteella. Osittaisessa etälaskennassa suoritusta siirretään dynaamisesti reunasolmulle.

Offloading on varmaan samankaltainen termi. Reunalla suoritettava etälaskenta saattaa siirtää ohjelman suorituksen kokonaan tai osittain asiakaslaitteelta reunasolmulle. Laskennan tulos lähetetään takaisin asiakaslaitteelle ja sitä käytetään osana muuta laskentaa. Laskennan hallinta suoritetaan asiakaslaitteella. Etälaskentaa motivoi raskaiden operaatioiden siirtäminen asiakaslaitteelta reunalle. Erityisesti mobiililaitteilla akkuvirran säästäminen on keskinen tekijä. Etälaskennan kannattavuus puhtaasti akkuvirran näkökulmasta muuttuu kannattavaksi, kun suoritettavan ohjelman lähettäminen ja tuloksen vastaanottaminen kuluttavat vähemmän akkua kuin ohjelman suorittaminen paikallisesti asiakaslaitteella. Todellisuudessa pelkästään akkuvirran säästäminen ei riitä, sillä muuten laskentaa voitaisiin siirtää pilveen. QoS kuitenkin heikkenee, mikäli laskennan suorittamiseen kuluva aika pitenee huomattavasti. Reunalle on teoriassa nopeampi yhteys ja nopeampi vastaus. Voitaisiin siis laskea että suoritusajassa mitaten ohjelman siirtäminen reunalle on kannattavaa kun suoritettavan ohjelman lähettäminen palvelimelle, sen suorittaminen ja tuloksen vastaanottaminen kestävät vähemmän aikaa kuin ohjelman suorittaminen paikallisesti. Ongelmana on että suorituksien aikavaatimus ei ole eksakti vaan ainoastaan arivoitavissa. Lisäksi suoritusaikainen aika-arvion tekeminen vie myös aikaa. Ajan ja akkuvirran säästämiseksi tehtävät toimet ovat siis keskeisimmät haasteet etälaskentaa toteutettaessa. Näiden käsittelyä ei tämän enempää tässä tutkielmassa käsitellä niiden monimutkaisuuden vuoksi.

5.2 Etäinstanssi

Etäinstanssissa asiakaslaitteella on yhteys reunalle. Asiakaslaitteelle tulee ainoastaan näkymä palvelun tilasta omalle laitteelleen. Vastaavasti kuin ottaisi SSH tai VNC yhteyden toiselle laitteelle.

-Tarkista oliko jossain järkevää lähdettä tähän

5.3 Migraatio

Reunalaskennan migraatiolla tarkoitetaan asiakaslaitteeseen liittyvän tilan tai laskennan siirtämistä reunasolmulta toiselle. Handoff/-over on mobiiliverkoissa yleisesti ilmenevä tilanne jossa, mobiililaitteen yhteys siirtyy tukiasemalta toiselle. Reunalaskennassa handover tehdään reunasolmulta toiselle. Reunalaskennan toteutustavasta riippuen, saatetaan tarvita niin sanottu live migraatio. Live migraatiossa suorituksen alla oleva sovellus tai virtuaalikone siirretään suoritusalustalta toiselle. Tavallisesti live migraatiota käytetään palvelinkeskusympäristössä virtuuaalikoneiden siirtämiseen suorituksen aikaiseen siirtämiseen. Live migraation tavoitteena on minimoida virtuaalikoneen käyttöön kohdistuva käyttökatkon kesto. Live migraatio toimii siten, että siirrettävää virtuaalikonetta aletaan kopioimaan kohdelaitteelle. Koska kyseessä on suoritusaikainen kopiointi, tilan kopiointi sisältää tallennustilan ja muistin kopioinnin. Mikäli palvelinkeskuksessa on jaettu levypalvelin, riittää ainoastaan muistin kopiointi. Tämä huomiona lähinnä siksi, että voidaan olettaa että reunasolmuilla ei ole jaettua levypalvelinta, jolloin joudutaan kopioimaan myös tallennustila. Kun virtuaalikoneen tila on saatu kopioitua uudelle alustalle, joudutaan kopioimaan kopioinnin aikana virtuaalikoneen tilaan tapahtuneet muutokset. Kopiointia jatketaan iteroiden, kunnes päästään tilaan, jossa muutoksien määrä kopio-iteraatiota kohden ei enää pienene. Tällöin alkuperäinen virtuaalikone pysäytetään ja viimeisten muutoksien kopioinnin aikana virtuaalikone ei ole käytettävissä. Tämän jälkeen migratoitu virtuaalikoneinstanssi on käytettävissä uudella alustalla. [Ha et al., 2015]

Reunalaskennan ja perinteisen palvelinympäristön vaatimukset live migraatiota kohtaan ovat hieman erilaiset. Päällimmäisenä erona on migraatioon käytettävän kaistan suuruus. Palvelinsaleissa yhteysnopeudet ovat suuria ja etäisyydet verrattain lyhyitä. Lisäksi palvelinsaliympäristössä migraatioita on mahdollista tehdä koordinoidusti ilman aikarajoitteita. Reunasolmujen välillä olevien yhteyksien nopeudet saattavat vaihdella suuresti. Pitkä kopiointiaika johtaa pidempään käyttökatkokseen ja täten palvelun laadun heikkenemiseen. Kopiointiajan minimoimiseksi on pyrittävä pitämään siirrettävän datan määrä mahdollisimman pienenä. Reunasolmujen migraatiotarpeeseen vaikuttaa suuresti käyttäjien liikkuminen verkossa. Voidaan kuvitella tilanne jossa aamulla kaupungin keskustaan saapuvat työmatkailijat aiheuttavat "migraatiotulvan", joka ruuhkauttaa reunasolmujen käytössä

olevat kaistat.

Virtuaalikone-aihioihin perustuvassa järjestelmässä ainoastaan virtuaalikoneeseen tehdyt muutokset siirretään [**?**]. Näin säästetään migraation aikana siirrettävän datan määrä. Mikäli on tiedossa, minne käyttäjä on siirtymässä, voitaisiin migraatio tehdä suoraan kohteena olevalle reunasolmulle. Palvelun laadun takaamiseksi migraation ennakointi on tärkeää. Mitä aikaisemmin aie siirtyä toiselle reunasolmulle tiedetään sen paremmin siihen keretään valmistautumaan. Migraation kesto on riippuvainen siirrettävän datan määrästä, sekä Mikäli käyttäjä haluaa keskeyttää reunapalvelun käytön saatetaan tila tai laskenta siirtää käyttäjän laitteelle tai jättää reunalle odottamaan.

[Ha et al., 2015] Tukivat että migraatio virtuaalikone-aihioilla + muutoksien siirroilla aiheuttaa noin 1s downtimen hitaahkolla verkolla. Tavoitteena oli minimoida siirrettävän datan määrä ja tutkia handoff+ migraation aiheuttaman käyttökatkoksen pituutta. Testi tehtiin cloudleteillä.

MobiScudissa ja SMOREssa esitetään että käytetään livemigraatiota, mutta ei sen tarkemmin pureuduta ongelmaan. Todetaan vain että käytetään live migraatiota

5.4 Integraatio mobiiliverkkoihin

SMOREssa ratkaisu on että SMORE nuuskii/monitoroi LTE/EPC kommunikointia ja injektoituu väliin.

5.5 IP-verkko

Service discovery IP-verkossa.

5.6 Kommunikaatio asiakaslaitteen (UE) ja Reunasolmun kanssa

5.7 Hallinta

Sisältää palveluiden etsimisen (Service Discovery) ja palveluihin ohjauksen esim Software Defined Networkingin avulla (SDN).

6 Reunan rakenne

Rakenteeseen vaikuttaa ehkä eniten seuraavat kaksi valintaa

- Kerroksittain
- Lähelle vähän vai kauemmaksi enemmän

Molempiin ratkaisuihin liittyy omat haasteensa. Etenkin voidaan olettaa että lähelle hajalleen aiheuttaisi huomattavasti enemmän ylläpitotyötä.

7 Esitetyt ratkaisut

Reunalaskennasta löytyy valtavasti artikkeleita, joissa esitetään reunalaskentaan liittyvien haasteiden ratkaisuja. Ratkaisut ovat yksittäisi tapauksia eivätkä sinällään arkkitehtuuriratkaisuja.

7.1 Cloudlet

Minkä ongelman cloudlet ratkaisee? Miten cloudlet toimii? Mitkä ovat cloudletin ongelmat?

Cloudlet on reunasolmulla suoritettava virtuaalikone-teknologiaan perustuva ratkaisu. [Satyanarayanan et al., 2009] Virtuaalikoneisiin pohjautuva cloudlet on yksi tunnetuimmista reunapalveluiden tuottamisen konsepteista. Cloudletin ideana on hyödyntää muokattuja virtuaalikone-instansseja, joita voidaan suorittaa reunasolmussa. Cloudletillä tarkoitetaan järjestelmää, joka suorittaa reunasolmulla virtuaalikoneiden hallintaan liittyviä toimia. Yhdellä reunasolmulla voidaan siis suorittaa usean eri käyttäjän virtuaalikoneita.

Reunan luonteesta johtuen cloudletin oletetaan ylläpitävän ainoastaan sellaisia palveluita jotka käyttävät reunaa hetkellisesti tai väliaikaisesti satya09. Cloudlettiä ei ole siis tarkoitettu pidempiaikaiseen tallennukseen, vaan enemminkin suoritusaikaisten tarpeiden täyttämiseen.

Cloudletin keskeisimmäksi rakennuspalikaksi on valittu virtuaalikone. Verrattuna prosessi tai sovellus tason virtualisointiin, virtuaalikoneiden käyttö ei aseta rajoitteita palvelun toteuttamiseen käytettävään ohjelmointikieleen [Satyanarayanan et al., 2009]. Lisäksi virtuaalikoneiden käyttö cloudlettien perustana mahdollistaa helpon tavan jakaa reunasolmun resursseja ja pitää asikaskohtaiset instanssit toisistaan erillään.

Tavallisen virtuaalikone-kuvan suuruus tallennettuna on noin 8 gigatavua. Koska reunasolmujen pääasiallinen idea on tarjota suoritusympäristö palveluille, ei näin suuren tiedoston tallentaminen reunasolmulle jokaista käyttäjää kohden ole tarkoituksenmukaista. Ja kuten jo aiemmin mainittu, reunasolmujen resurssit oletetaan rajallisiksi verrattuna palvelinsaliympäristöön. Täten pidempiaikainen tiedostojen tallennus on parempi tehdä keskitettyihin ratkaisuihin. Sen lisäksi että tallentaminen reunasolmuille ei ole mahdollista, kokonaisen virtuaalikoneen siirtäminen verkon yli olisi huomattavan hidasta ja se tukkisi verkon. Siirrettävyys korostuu etenkin tilanteessa jossa siirrettäviä virtuaalikoneita olisi useita.

Cloudletin keskeisimpiä oivalluksia onkin käyttää virtuaalikoneissa yhteistä pohjaa. Yhteisen pohjan käyttäminen mahdollistaa sen että jokaisesta virtuaalikoneesta voidaan tallentaa vain pohjaan tehdyt muutokset. Muutokset sisältävää tiedostoa kutsutaan pinnoitteeksi. Pinnoitteen ja pohjan yhdistämiseen käytetään dynaamista virtuaalikone synteesiä. Dynaamisuudella tarkoitetaan sitä, että käyttäjän alkaessa käyttämään cloudlet infrastruktuuri rakentaa käyttäjän muutokset sisältävästä pinnoitteesta ja virtuaalikone-

pohjasta, suoritettavan virtuaalikoneen. Dynaaminen virtuaalikoneiden syntetisointi edellyttää että cloudletiltä löytyy oikeanlainen virtuaalikonepohja. Kun virtuaalikone halutaan sulkea cloudlet tallentaa virtuaalikoneeseen tehdyt muutokset pinnoitteeksi ja pakkaa tiedoston. Satayanaran ensimmäisessä cloudletteja käsittelevässä julkaisussa [Satyanarayanan et al., 2009] muutoksien tallentaminen pinnoitteeksi tuottaa noin 100-200 megatavun kokoisen tiedoston. Tämän kokoinen tiedosto on nykyisillä tiedosiirtonopeuksilla siirrettävissä inhimillisessä ajassa, jopa langattomien yhteyksien yli.

Cloudlettejä voi olla useita erilaisia, siispä cloudlet toiminta edellyttää että jokaisella reunasolmulla tulee olla oikean cloudlet-pohja saatavilla ja käyttäjien cloudlettien on pohjauduttava johonkin rajalliseen määrään erilaisia cloudlet pohjia.

*Hallinta

RAW – Migraatio

Cloudlettien välinen migraatio ja handoff. Cloudlettien väliselle migraatiolle voi olla useita eri syitä. Migraatio tilanteessa jossa virtuaalikone siirretään paikasta toiseen odottamaan käyttöönottoa on keskeisin. Pelkän pinnoitteen siirtäminen riittää, mikäli tarvittava virtuaalikonepohja löytyy kohteena olevasta cloudletistä. On kuitenkin huomattava että pinnoiteen tallettamiseen voi olla kannattavaa käyttää myös asikkaan laitetta, mikäli se vain on mahdollista. Perusteluina asiakaslaitteen käyttölle tallennusmediana voisi olla tapaus, jossa asiakas matkustaa ulkomaille ja pinnoite tulisi siirtää hyvin kaukana sijaitsevalta palvelimelta kohteena olevalle cloudletille. Tällöin saattaisi olla nopeampaa siirtää pinnoite suoraa asiakaslaitteelta.

*RAW – Handoff

7.2 FMC – Follow me cloud

7.3 Small Cell Cloud (SCeNB)

7.4 MobiScud ja SMORE

7.5 Concert

7.6 ETSI reference architecture

7.7 Yhteenveto

Lähteet

[Ha et al., 2015] Ha, K., Abe, Y., Chen, Z., Hu, W., Amos, B., Pillai, P., and Satyanarayanan, M. (2015). Adaptive vm handoff across cloudlets. *Technical report, Technical Report CMU-C S-15–113, CMU School of Computer Science.*

[Mao et al., 2017] Mao, Y., You, C., Zhang, J., Huang, K., and Letaief, K. B. (2017). A survey on mobile edge computing: The communication perspective. *IEEE Communications Surveys Tutorials*, PP(99):1–1.

[Satyanarayanan, 2001] Satyanarayanan, M. (2001). Pervasive computing: vision and challenges. *IEEE Personal Communications*, 8(4):10–17.

[Satyanarayanan et al., 2009] Satyanarayanan, M., Bahl, P., Caceres, R., and Davies, N. (2009). The case for vm-based cloudlets in mobile computing. *IEEE Pervasive Computing*, 8(4):14–23.