TEKNILLINEN KORKEAKOULU

Sähkö- ja tietoliikennetekniikan osasto

Matti Laipio

Hybriditietoverkon simuloinnin luotettavuudesta

Diplomityö, joka on jätetty opinnäytteenä tarkastettavaksi diplomi-insinöörin tutkintoa varten Espoossa 4.6.2007

Työn valvoja	Professori Raimo Kantola
Työn ohjaaja	TkL Mika Nordman

Tekijä: Matti Laipio

Työn nimi: Hybriditietoverkon simuloinnin luotettavuudesta

Päivämäärä: 4.6.2007 Sivumäärä: 51

Osasto: Sähkö- ja Tietoliikennetekniikanosasto

Professuuri: Tietoverkkotekniikka

Työn valvoja: Professori Raimo Kantola

Työn ohjaaja: TkL Mika Nordman

Tämän työn tarkoitus oli selvittää simuloidun ja reaalisen tietoverkon muodostaman hybriditietoverkon simuloinnin luotettavuutta. Tämän lisäksi haluttiin tutkia simuloinnin taustaa ja tietoverkkojen simulointiin liittyviä aiheita ja sovelluksia. Hybridiverkon luotettavuutta tutkittiin tarkemmin mittausten avulla.

Diskreetti tapahtumapohjainen simulaatio on käytössä monissa tietoverkkosimulaatioon ja tietoverkon mallintamiseen erikoistuneissa sovelluksissa. Siihen liittyy muutamia ongelmia reaaliaikasimulaatioiden suhteen, kuten samalle ajanhetkelle ajoitetut tapahtumat. Nämä ongelmat tulee ottaa huomioon mallinnettaessa, jotta päästäisiin mahdollisimman realistisiin simulointituloksiin.

Mittauksia varten rakennettiin Teknillisellä korkeakoululla tietoverkko, jonka mittaustuloksia käytettiin vertailupohjana hybridiverkon ja täysin simuloidun verkon tuloksille. Verkossa käytettiin eri tekniikoita riittävän monimutkaisuuden saavuttamiseksi, jotta verkko ja sen liikenne ei olisi liian yksinkertainen simuloitavaksi.

Mittaustuloksista voidaan päätellä, että jo kohtuullisen pienen tietoverkon reaaliaikainen simuloiminen raskaalla liikenteellä vaatii paljon laskentatehoa simulointityöasemalta. Täysin simuloidun verkon tulokset viittaavat siihen, että mallit ovat kohtuullisen realistisia. Monissa tapauksissa simulointityöaseman laskentatehoa kasvattamalla päästään realistisempiin tuloksiin reaaliaikaisissa simulaatioissa.

Avainsanat: Tietoverkko, Simulaatio, Emulaatio, Luotettavuus, OPNET

Author: Matti Laipio

Name of the Thesis: Reliability in hybrid network simulation

Date: 4.6.2007 Number of pages: 51

Department: Department of Electrical and Communications Engineering

Professorship: Networking Technology

Supervisor: Professor Raimo Kantola

Instructor: Lic.Sc. (Tech.) Mika Nordman

The purpose of this thesis was to research the reliability in simulation of a hybrid of simulated and real network. Simulation, network simulation and network simulation software were studied to get information of the factors, which affect to reliability. More knowledge of reliability was gathered by measurements.

Discrete event simulation is used in many software tools, which are made for network simulation and modeling. There are some issues related to real-time simulation, like simultaneous events. These issues should be considered when modeling to get more realistic results.

The network for measurements was set up in Helsinki University of Technology's laboratory. The results were compared with hybrid network and fully simulated network. Different technologies were used to get more complex network and to make sure that the network and its traffic wouldn't be too simple to simulate.

The results indicate that real-time simulation of even a relatively small network requires much processing power from the simulation workstation. The results from fully simulated network suggest that models are quite realistic. In many cases it is possible to get more realistic results in real-time simulations by increasing processing power.

Keywords: Network, Simulation, Emulation, Reliability, OPNET

ALKULAUSE

Tämä työ on tehty Puolustusvoimien Teknillisellä Tutkimuslaitoksella Elektroniikka- ja Informaatiotekniikkaosastolla. Hybridisimulaation mahdollisuuksista ja luotettavuudesta haluttiin tietoa uuden simulointitutkimusympäristön käyttöönoton vuoksi. Suurin osa työhön kuuluvista mittauksista tehtiin Teknillisen Korkeakoulun Tietoverkkolaboratoriossa. Erittäin mielenkiintoisen aiheen lisäksi työssä oli mukavasti käytännönläheisyyttä, kun rakennettiin oikea verkko laboratorioon vertailumittauksia varten.

Ensinnäkin, haluan kiittää professori Raimo Kantolaa työn valvomisesta ja TkL Marko Luomaa työn alkuun saattamisesta sekä ohjeista mittausten järjestämiseksi.

Ehdottomasti suurin kiitos itseni jälkeen kuuluu työn ohjaajalle TkL Mika Nordmanille, joka jaksoi kannustaa, ohjeistaa ja luottaa työn vaikeimpinakin hetkinä. Kiitän dosentti Matias Aunolaa ja DI Sami Peltotaloa lukuisista hyvistä kommenteista ja parannusehdotuksista. Lisäksi haluan kiittää koko PVTT:n henkilökuntaa avuliaisuudesta ja hyvästä työilmapiiristä. Kiitos myös kaikille niille, jotka edesauttoivat opintojani ja jaksamistani niissä, erityisesti opiskelutovereilleni.

Lauralle kiitos vierelläni kulkemisesta.

Haluan omistaa tämän työn vanhemmilleni kiitoksena kaikesta tukemisesta ja mahdollisuuden antamisesta.

Hyvinkäällä 4.6.2007

Matti Laipio

SISÄLLYSLUETTELO

LYHENNELUETTELO

ARP	Address Resolution Protocol Osoitteenselvittämisprotokolla
ATM	Asynchronous Transfer Mode Tahdistamaton toimintamuoto
DES	Discrete Event Simulation Diskreetti tapahtumapohjainen simulaatio
FIFO	First-In-First-Out Tapa palvella saapumisjärjestyksen mukaan
FTP	File Transfer Protocol Tiedostonsiirtoprotokolla
GW	Gateway Rajapinta
HLA	High Level Architecture Standardi simulaatioiden väliseen kommunikointiin
HTML	Hypertext Markup Language Hypertekstikieli
HTTP	Hypertext Transfer Protocol Hypertekstinsiirtoprotokolla
ICMP	Internet Control Message Protocol Internetin hallintaviestiprotokolla
IP	Internet Protocol Internetin protokolla
ISO	International Organization for Standardization Kansainvälinen standardointielin
LAN	Local Area Network Lähiverkko
LSDB	Link State Database Linkkien tilatietokanta
MAC	Medium Access Control Siirtoyhteyskerroksen ohjaus
MANET	Mobile Ad hoc Network Langaton tiettyä tarkoitusta varten muodostuva tietoverkko

MPOA	Multiprotocol Over Asynchronous Transfer Mode (ATM) Menetelmä eri protokollien käyttöön ATM:n kanssa
NIC	Network Interface Controller Verkkokortti
NS	Network Simulator University of California Berkeleyssä kehitetty tietoverkkosimulaattori
OC	Optical Carrier Optinen kuitu
OSI	Open Systems Interconnection Menetelmä järjestelmien standardointiin
OSPF	Open Shortest Path First Reititysprotokolla
OTcl	Object Oriented extension of Tool Command Language Oliopohjainen tulkattava ohjelmointikieli
PVC	Permanent Virtual Circuit Pysyvä virtuaalinen kytkentä
RIP	Routing Information Protocol Reititysprotokolla
RFC	Request for Comments Dokumentointimalli Internetin tekniikoille
RTT	Round-Trip Time Menopaluuaika
SITL	System-in-the-Loop Reaalisen järjestelmän liittäminen simulaatioon
Tcl	Tool Command Language Tulkattava ohjelmointikieli
TCP	Transmission Control Protocol Kuljetuskerroksen protokolla
VoIP	Voice over Internet Protocol Puheensiirto Internetissä

1. Johdanto

Tietoverkkojen simulointi on jo pitkään ollut hyvä väline uusien tietoverkkolaitteiden ja niiden käyttämien protokollien suunnittelussa. Uusien tekniikoiden testaaminen simuloimalla oikeaa tietoverkkoa on usein tehokkain vaihtoehto. Joskus simulointi on myös ainoa vaihtoehto tiedon keräämiseen, koska oikeaa tietoverkkoa ei vielä ole.

Oikean tietoverkkolaitteen liittäminen osaksi suurempaa tietoverkkosimulaatiota tulee kyseeseen, kun halutaan tutkia laitteen toimintaa suhteessa suurempaan verkkoon, jossa on mallinnettavissa olevia tai jo mallinnettuja laitteita. Testit oikealla tietoverkolla, simulointi ja oikean tietoverkon osittainen simulointi eivät ole toisensa poissulkevia ratkaisuja, vaan niitä kaikkia voidaan käyttää yhdessä saamaan enemmän ja luotettavampaa tietoa tutkittavasta tietoverkosta. Näillä kaikilla kolmella menetelmällä on omat vahvuudet ja heikkoudet, joita käsitellään tässä työssä.

Simuloitavien laitteiden mallintaminen tulisi tehdä aina kulloistakin ongelmaa silmälläpitäen, jolloin mallinnetaan vain niitä asioita, jotka vaikuttavat simuloitaviin tuloksiin. Monimutkaisten tietoverkkolaitteiden mallintaminen on joka tapauksessa erittäin haasteellinen tehtävä, joka vie paljon aikaa ja resursseja. Tilanne voi olla myös sellainen, että tutkittavan tietoverkkolaitteen mallintaminen ei ole toivottavaa, koska sen toiminta on vähintään luottamukselliseksi määriteltävää tietoa.

Hybridisimulaatiolla tarkoitetaan yleensä simulaatiota, jossa käytetään liikenteen mallintamiseksi sekä pakettiliikennettä että vähemmän laskentatehoa vaativia liikenteen virtamalleja, joita käsitellään myöhemmin. Tästä poiketen hybridisimulaatiolla ja hybridiverkolla tarkoitetaan tässä työssä simuloidun ja reaalisen tietoverkon yhdessä muodostamaa järjestelmää.

Tässä diplomityössä esitellään yksi ratkaisu oikeiden tietoverkkolaitteiden liittämiseen simulaatioon ja tutkitaan tällaisen hybridiverkon sekä täysin simuloidun verkon eroja suhteessa vastaavaan oikeaan verkkoon. Lisäksi työssä selvitetään simuloinnin taustaa, sekä hybridiverkkojen simulointia.

Toisessa luvussa käsitellään simuloinnin ja erityisesti tietoverkon simuloinnin taustaa. Luvussa 3 esitellään tietoverkkosimulaattoreita, joilla voidaan toteuttaa reaalisen ja simuloidun tietoverkon hybridi. Luvussa 4 esitellään mittausjärjestelyt erityyppisten tietoverkkojen vertailuun ja luvussa 5 kerätyt tulokset. Luku 6 käsittelee johtopäätöksiä mittaustuloksista.

2. Tietoverkon simulointi

Tässä luvussa käsitellään tietoverkon simulointia ja simulointiin liittyviä periaatteita sekä nykyisin laajasti käytössä olevia ratkaisuja luotettavaan simulointiin.

2.1 Johdanto simulointiin

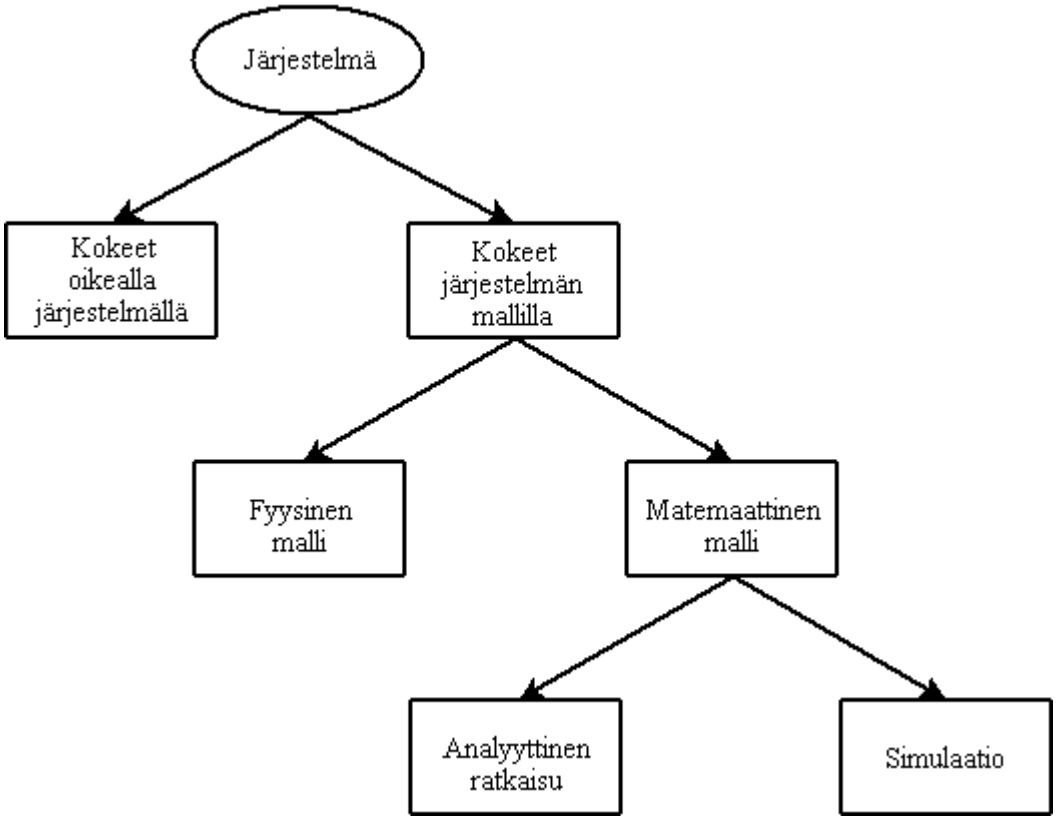
Jos tarkasteltava kohde on tarpeeksi yksinkertainen, voidaan sen toiminta määritellä analyyttisesti matemaattisia menetelmiä käyttäen. Tällöin saadaan tarkkaa tietoa kohteen toiminnasta. Useimmat oikean maailman järjestelmistä ovat kuitenkin liian monimutkaisia realistisen mallin analyyttiselle tarkastelulle. Tietokonesimulaatio on tällöin sopiva tapa estimoida järjestelmän tuottamia tuloksia. Järjestelmät, joita mallinnetaan, voidaan ajatella joko diskreetti- tai jatkuva-aikaisiksi järjestelmiksi. Harvat järjestelmät ovat pelkästään jompaakumpaa, mutta simulointia ajatellen on mahdollista luokitella mallinnettava järjestelmä diskreetiksi tai jatkuvaksi. Järkevästi luotu malli on aina oikean vastineensa yksinkertaistus. Mallin ei siis tule olla kopio oikeasta maailmasta, vaan mallintaa vain ne komponentit, jotka vaikuttavat estimoitaviin tuloksiin. Kuvassa 2.1 esitetään simulaation muodostamista järjestelmästä. [Law00, Ban01]

Mallintaminen ja simulointi tulevat kyseeseen erityisesti silloin kun [Gar90]:

- On mahdotonta käyttää itse järjestelmää – esimerkiksi, kun sitä ei vielä ole
- Itse järjestelmän tutkiminen on liian kallista tai riskit liian suuret – esimerkiksi, kun on kyseessä fysiologiset järjestelmät
- On epäkäytännöllistä tutkia itse järjestelmää – esimerkiksi siksi, koska se veisi liian paljon aikaa

Simulaatiomallit voidaan jakaa staattisiin tai dynaamisiin, deterministisiin tai stokastisiin ja diskreetteihin tai jatkuviin malleihin. Staattiset mallit esittävät järjestelmän tietyllä ajanhetkellä, kun taas dynaaminen malli kuvaa järjestelmää ajan kuluessa. Mallit, joissa ei

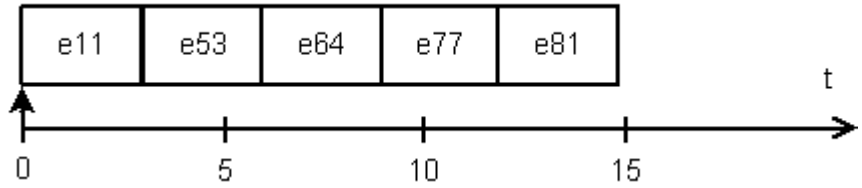
ole ollenkaan satunnaismuuttujia, ovat deterministisiä. Toisaalta yksittäistä tietoverkon solmua mallinnettaessa voidaan käyttää stokastista mallia, jolloin pakettien saapumisajat luodaan satunnaislukugeneraattorilla.



Kuva 2.1 Järjestelmän mallinnus ja simulointi [Law00]

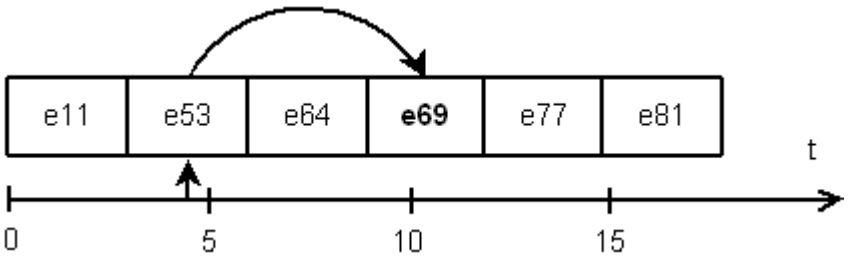
2.2 Diskreetti tapahtumapohjainen simulaatio

Diskreetti tapahtumapohjainen simulaatio (Discrete Event Simulation – DES) simuloi järjestelmän mallia, jonka tila muuttuu vain tiettyinä ajan hetkinä. Järjestelmä voi muuttaa tilaansa vain äärellisen monta kertaa. Kaikki tilanteet, joissa järjestelmän tila saattaa muuttua, voidaan kuvata erilaisilla tapahtumilla. Luonnollisesti on pidettävä kirjaa simuloidusta ajasta, joka yleensä DES-ajoissa eroaa merkittävästi reaaliajasta. Yleisin käytetty ajankirjausmenetelmä on seuraavan tapahtuman mukaan lisättävä aika (*next-event time advance*). Tällöin tapahtumien tapahtuma-aika on tiedossa etukäteen ja simulaatioaika päivitetään aina uuden tapahtuman tullessa suoritukseen. Tapahtumien välinen aika voidaan sivuuttaa ja hypätä aina seuraavaan tapahtumaan, jolloin tietokonesimulaatiossa säästyy laskenta-aikaa, eikä suorittimella ole tyhjäkäyntiä. [Law00, Las06, Opn06]



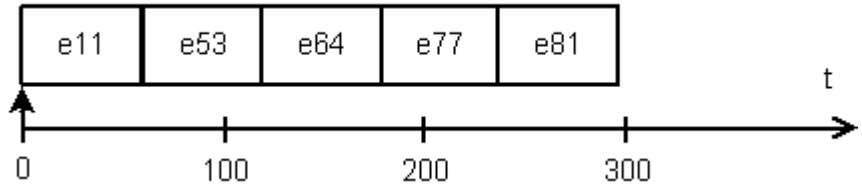
Kuva 2.1 Tapahtumalista simulaation alussa

Käynnistettäessä DES-simulaatio, luodaan tapahtumalista, johon kootaan kaikki etukäteen tiedossa olevat tulevat tapahtumat. Kuvassa 2.2 on havainnollistettu tapahtumalistaa simulaation alussa. Kuvan tapahtumat ovat nimetty siten, että tapahtumien tapahtuma-aika on ”e”-etuliitteen jälkeinen luku. Esimerkiksi tapahtuman ”e53” tapahtuma-aika on 53 sekuntia. Vaaka-akselilla on esitetty simulaation suoritusaika sekunneissa. Kuvan 2.2 simulaation suorittamiseen kuluisi n. 15 sekuntia ja simuloitu ajanjakso olisi 81 sekuntia, mikäli tapahtumalistaan ei tulisi muutoksia suorituksen aikana. Aika-akselille kohtaan ”0”, on merkitty simulaation suoritusvaihetta kuvaava nuoli. Kuvat 2.3-2.6 ovat esitetty samalla tavalla. Tietoverkkosimulaattoreissa ei välttämättä ole juuri kuvien 2.2-2.6 kaltaista tietorakennetta, mutta tapahtumalistan voidaan mieltää toimivan, kuten kuvissa on esitetty. Tapahtumalista voi olla useammassa eri tietorakenteessa, jotka on suunniteltu siten, että niiden luku- ja kirjoitustoimenpiteet myös listan keskelle ovat erittäin nopeita. Tapahtumalistaoperaatioiden tehokkuus on erittäin merkittävä asia koko simulaattorin tehokkuuden kannalta. [Opn06, Law00]



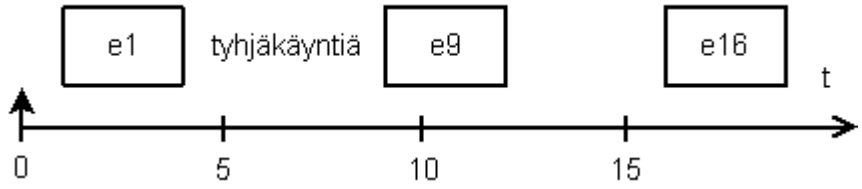
Kuva 2.2 Suorituksen aiheuttama muutos tapahtumalistaan

Simulaatiokelloa, joka pitää tiedon simuloidusta ajasta, päivitetään uusien tapahtumien tullessa suoritukseen. Tapahtumalista saattaa muuttua simulaation aikana, kun suoritettavat tapahtumat lisäävät uusia tapahtumia tapahtumalistaan. Kuvan 2.3 tapahtumassa ”e53” syntyy uusi tapahtuma ”e69”, joka sijoitetaan tapahtumalistaan aikajärjestyksessä oikeaan kohtaan.



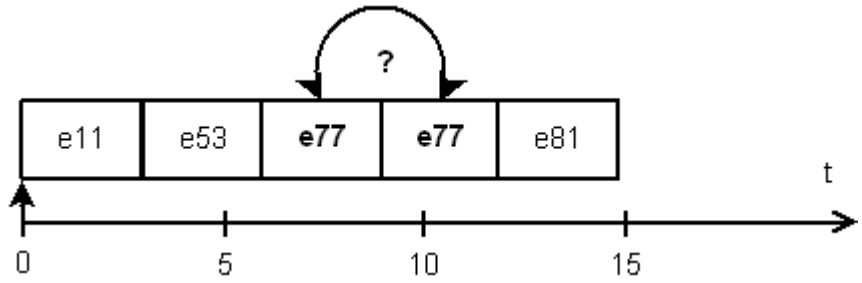
Kuva 2.3 Raskaan simulaation tapahtumalista

Kuvassa 2.4 on esitetty tapahtumalista simulaatiossa, jossa tapahtumien suorittamiseen menee niin paljon aikaa, että simulaatiokello etenee hitaammin kuin suoritusaika. Tilanne johtuu suoritukseltaan pitkäkestoisista tapahtumista tai erittäin lyhyestä ajasta tapahtumien välillä.



Kuva 2.4 Reaaliaikaisen simulaation tapahtumalista

Simulaatiokello ja suoritusaika on mahdollista synkronoida siten, että simulaatio etenee reaaliajassa, ts. simulaation tapahtumat suoritetaan oikeine suoritusaikoineen, eikä välittömästi perätysten kuten normaalisti. Kuvassa 2.5 on esitetty reaaliaikasimulaatio, joka on luonnollisesti mahdollinen vain, jos kuvan 2.4 kaltainen raskas simulaatio ei ole suoritettavana. Reaaliaikainen simulaatio ei ole järkevä tavanomaisissa simulaatioissa, koska siinä on tyhjäkäyntiä tapahtumien välissä ja siten suoritusaikaa kuluu tarpeettoman paljon.



Kuva 2.5 Ongelmatilanne tapahtumalistassa

Kuvan 2.6 esittämä tilanne on tyypillinen tapahtumalistan suoritusjärjestykseen liittyvä ongelma. On varsin tavanomaista varsinkin tietoverkkosimulaatioissa, että tapahtumia

kirjataan samalle ajanhetkelle. Tämä johtuu mm. siitä, että tietoverkon järjestelmien eri osat toimivat usein samalla kellolla, jolloin monet niiden generoimat tapahtumat sattuvat varsin todennäköisesti samalle ajanhetkelle. Toinen syy on ideaalisten laitteiden mallintaminen, jolloin voidaan esimerkiksi määritellä tietyn prosessin viiveeksi 0 sekuntia. Tällöin prosessi generoi uuden tapahtuman täsmälleen samalle ajanhetkelle. Yhtäaikaisia tapahtumia voi syntyä muutenkin monimutkaisessa ja laajassa simulaatiossa, joten simulaattoreihin on lisätty mekanismeja ongelman ratkaisemiseksi. Seuraavaksi esitellään luvussa 3 käsiteltävien tietoverkkosimulaattoreiden OPNET Modelerin ja Network Simulator 2:n (NS2) käyttämiä mekanismeja samanaikaisten tapahtumien järjestyksen ratkaisemiseen. [Opn06, Law00]

Luonnollinen järjestys

Luonnollista järjestystä käytetään oletuksena OPNET Modeler-simulointiohjelmistossa, mikäli muuta samanaikaisten tapahtumien järjestämistapaa ei ole määritelty tai muut tavat eivät pysty ratkaisemaan järjestystä. Samanaikaiset tapahtumat suoritetaan First-In-First-Out (FIFO) -periaatteella eli siinä järjestyksessä, jossa ne on lisätty tapahtumalistalle. Tämä on ainoa samanaikaisten tapahtumien järjestämistapa NS2-simulointiohjelmistossa. [Opn06, Fal07]

Järjestäminen keskeytysten avulla

Modeler tarjoaa kaksi muuta tapaa järjestää samanaikaiset tapahtumat. Kun tapahtumat ovat samassa mallinnettavassa verkon oliossa, voidaan niiden suorittamisjärjestykseen vaikuttaa olion muodostamien keskeytysten avulla. Keskeytyksillä on oma tyyppi, koodi ja prioriteettiarvo, jotka kaikki vaikuttavat keskeytyksen muodostaman tapahtuman suoritusjärjestykseen. [Opn06]

Verkon olioiden priorisointi

Eri verkon olioissa tapahtuvat samanaikaiset tapahtumat voidaan järjestää priorisoimalla oliot siten, että korkeamman prioriteetin omaavan olion tapahtumat suoritetaan ensin. [Opn06]

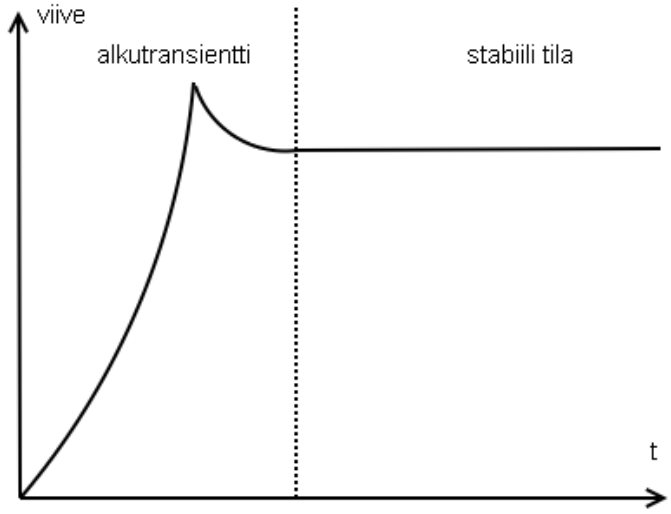
Vaikka reaaliaikaisessa simulaatiossa tapahtumat saataisiin järjestettyä siten, että simulaation toiminta vastaisi reaalimaailman vastaavaa järjestelmää, tulee vastaan silti ongelma, joka liittyy kuviin 2.5 ja 2.6. Tapahtumia tulee joka tapauksessa suoritettavaksi samalle ajankohdalle tai edellisen suoritus on vielä kesken, kun seuraavan tapahtuman suoritus tulisi alkaa. Tapahtumat saavat simulaatiossa suoritusajankohdakseen simulaation virtuaalikellon ajan, eli kuvassa 2.6 samaan aikaan suoritettavat tapahtumat saavat saman ajan. Reaaliaikasimulaatiossa NS2 ja Modeler toimivat tällä tavoin, joten päällekkäisten tapahtumien suoritusaikoihin tulee virhettä, koska tapahtumien suoritukseen kuluvaa aikaa ei oteta huomioon. Mahrenholz ja Ivanov [Mah04] ovat kehittäneet NS2:een parannuksia, jotka korjaavat tämän ongelman aiheuttamaa virhettä. Heidän ratkaisussaan päällekkäisten tapahtumien suoritusajat haetaan järjestelmän kellosta, joka vastaa reaaliaikaa. Ongelmalla on merkitystä vain silloin, kun tarkastellaan itse simulaattorin generoimia tuloksia. Tämä ongelma ei vaikuta tuloksiin, kun tutkitaan simulaation ulkopuolella olevia järjestelmiä, kuten mittalaitteita luvussa 4.1. Täysin simuloidun verkon tapauksessa tapahtumien suoritukseen kuluneella ajalla ei ole merkitystä, koska virtuaalikello on pysähdyksissä suorituksen ajan ja etenee ainoastaan tapahtumasta toiseen siirryttäessä.

2.3 Transienttien ja stabiilien tilanteiden simulointi

Järjestelmää tutkittaessa ollaan yleensä kiinnostuneita joko stabiileista tai transienteista tilanteista. Joskus voidaan haluta simuloida molempia tilanteita. Joka tapauksessa, simuloinnin tulosten kannalta on olennaista pyrkiä määrittämään molemmat kyseisistä vaiheista simuloinnin aikana ja mahdollisesti poistaa tuloksista epämieluisa vaihe.

Transientti tilanne syntyy yleensä simuloinnin alkuvaiheessa, kun järjestelmä on vielä tyhjä. Kun ollaan kiinnostuneita esimerkiksi keskimääräisestä viiveestä, alkutilanteen järjestelmän kuormittamattomuus ja alkuarvot vaikuttavat merkittävästi lopputuloksiin. Alkutransientin kestoa voidaan pienentää valitsemalla sopivammat alkuarvot, mutta sitä ei voida kokonaan poistaa tällä tavalla. Yksinkertaisin ja yleispätevin tapa selvittää alkutransientin kesto on määritellä se graafisesti kuvaajasta, joka esittää tarkasteltavan suureen muuttumista ajan suhteen. [Las06, Law00]

Kuvassa 2.7 on kuvattu mahdollista tilannetta simuloidussa tietoverkossa, jossa alkutilanteessa ei ole ollenkaan liikennettä. Liikenteen generointi tyhjään verkkoon aiheuttaa alkutransientin tarkasteltavaan suureeseen, joka tässä tapauksessa on liikenteeseen aiheutunut viive.



Kuva 2.6 Alkutransientin määrittäminen

Alkutransientista mahdollisesti aiheutuva virhe on helppo poistaa simulaation aikaleimatuista tuloksista leikkaamalla transientti vaihe pois tai aloittamalla tulosten kerääminen kun transientti ei enää vaikuta.

3. Mallinnus ja simulointi simulointiohjelmistojen avulla

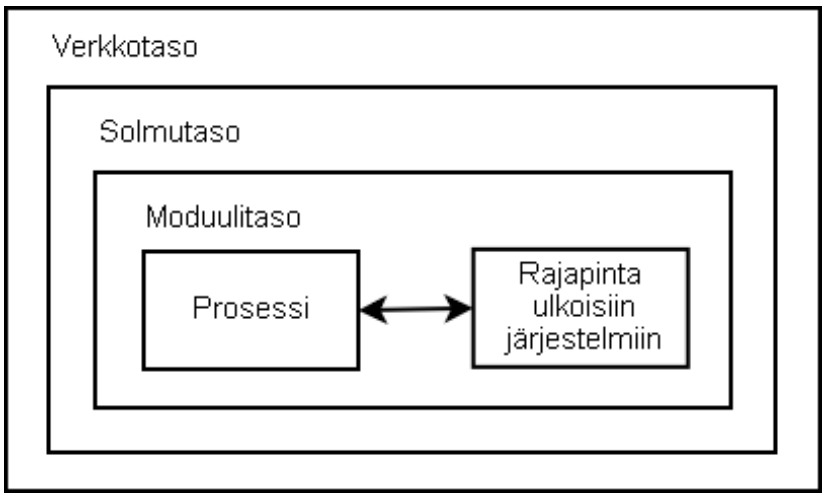
Tässä luvussa esitellään työkaluja tietoverkon simulointiin ja niiden käyttöä mallintamiseen sekä simulointiin.

3.1 Simulointiohjelmistot

Seuraavaksi käsiteltävät simulaattorit esitellään lyhyesti.

3.1.1 OPNET Modeler

OPNET Modeler [Opn06] on kaupallinen ja suljettu tietoverkkosimulaattoriohjelmisto. Lähdekoodia itse sovellukseen ei tästä syystä ole saatavilla, joten ohjelmiston tarkempi toiminnallisuuden määrittäminen on mahdotonta. Modelerin ensisijainen käyttökohde on sen nimensä mukaan verkon eri laitteiden mallintaminen, johon se tarjoaa työkalut graafisessa käyttöliittymässä. Käyttäjä voi mallintaa verkon laitteet ja protokollat itse kirjoittamalla C++-koodia tai käyttämällä OPNET:n valmista mallikirjastoa. Mallikirjaston malleista suurimpaan osaan on mukana lähdekoodit, joten myös valmiiden mallien tarkastelu ja muokkaaminen on mahdollista. Modelerista on saatavana myös karsittu versio nimellä ITGuru, joka perustuu valmiiseen mallikirjastoon, jonka malleja ei voi muokata tai tarkastella, eikä omia malleja tehdä.



Kuva 3.1 Modelerin hierarkiatasot [Opn06]

Modelerissa mallintaminen tapahtuu kolmessa eri tasossa:

- *Moduulitaso*
- *Solmutaso*
- *Verkkotaso*

Verkon mallintaminen alkaa hierarkiatason alimmalta tasolta, jossa on moduulin sisältämät prosessit. Kuvassa 3.1 on esitetty hierarkiatasot. Prosessit voivat myös keskustella ulkoisten järjestelmien kanssa, joihin voidaan lukea esimerkiksi toinen simulaatio tai oikea tietoverkkolaite. Yksi verkon solmu koostuu yhdestä tai useammasta moduulista. Korkeimmassa abstraktiotasossa, verkossa, on vain solmuja ja niiden välisiä linkkejä. Verkot voivat vielä olla sisäkkäisiä, eli aliverkoille löytyy oma mallinsa. Suurten verkkojen mallinnus selkeytyy, kun aliverkkoja saa yhden solmun taakse.

Modelerilla mallintamista ja simulointia käsitellään tarkemmin luvussa 3.2.

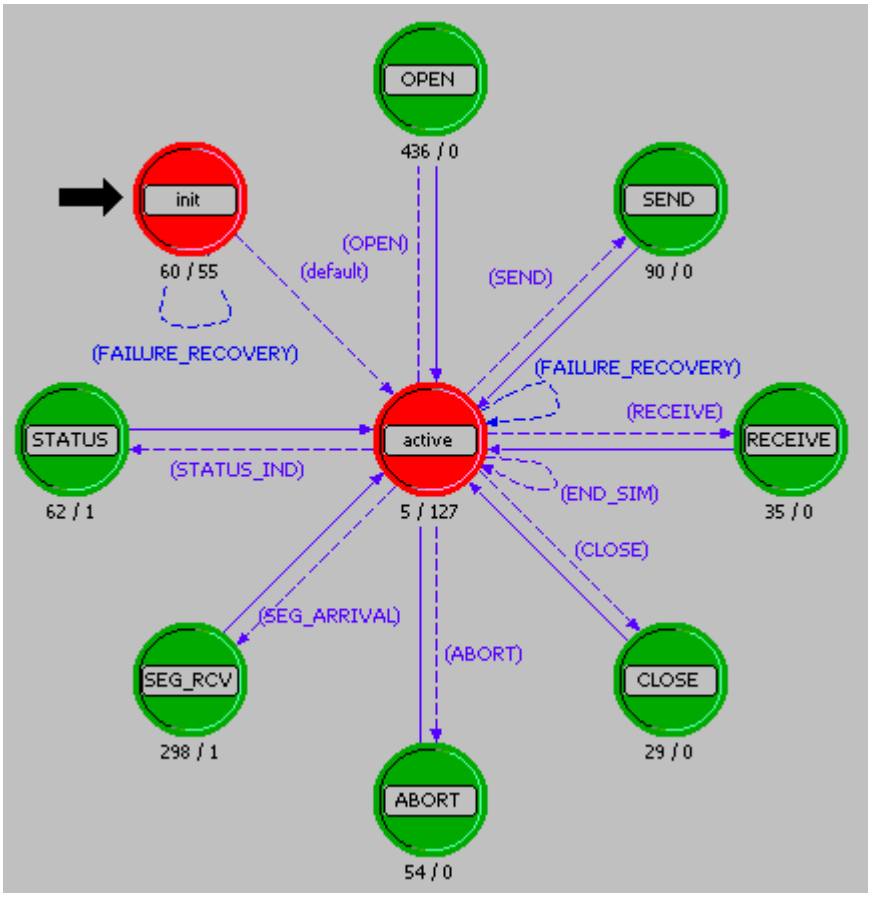
3.1.2 Network Simulator 2

Network Simulator 2 (NS2) on vapaa ohjelmistotyökalu tietoverkkojen simulointiin. NS2 perustuu kahteen ohjelmointikieleen, C++:aan ja Tool Command Languagesta (Tcl) jalostettuun oliopohjaiseen kieleen Object Oriented Tcl:iin (OTcl). NS2 on lisenssioitu General Public License (GPL) 2:lla, joka takaa ohjelmiston vapauden (*free software*), mutta on eri asia kuin ohjelmiston ilmaisuus. C++-kielellä on toteutettu tehokkuutta vaativat oliot, kuten protokollat ja muut tietoverkon komponentit. OTcl toimii lähinnä käyttöliittymänä ja sen avulla voidaan määritellä simuloitava tietoverkko. Kuten Modelerista, myös NS2:sta löytyy mittava kirjasto valmiille tietoverkon komponenteille ja protokollille. Lisäksi vapaan ohjelmiston toiminta voidaan tarvittaessa määrittää tarkasti tai muokata sitä tarpeen mukaiseksi lähdekoodin avulla.

3.2 Tietoverkon komponenttien mallinnus

Moduuleita tarvitaan Modelerissa esimerkiksi eri protokollien mallintamiseen ja niiden mallinnus tapahtuu tilakaavioiden avulla. Kuvassa 3.2 on esitetty Transmission Control Protocolin (TCP) [RFC793] mallinnus Modelerin mallikirjastossa. Kuvan TCP-mallia

käytettiin luvussa 4 esiteltävissä mittauksissa. Tilat voivat olla joko pakotettuja (vihreä) tai pakottamattomia (punainen). Jokaisella tilalla on suoritettavana alustuskoodi ja lopetuskoodi. Molempien koodirivien määrä on ilmoitettu kunkin tilan alapuolella. Pakottamattomassa tilassa prosessin suoritus pysähtyy tilan alustuskoodin suorittamisen jälkeen ja jatkuu vasta kun seuraava keskeytys prosessille saapuu. Itse moduulissa voi lisäksi olla määriteltynä muuttujia tai koodia, joita tilat ja tilasiirtymät voivat kutsua ja käyttää.



Kuva 3.2 TCP-protokollan tilakaavio

Kuvan 3.2 TCP-moduulin suoritus alkaa ”init”-tilasta ja tila muuttuu ”active”-tilaan, kun ”default”-keskeytys prosessille saapuu. Muut tilat ovat pakotettuja, eli tila muuttuu alkutilan jälkeen aina takaisin ”active”-tilaan pakotetusta tilasta ilman keskeytystä. Pakotettuihin tiloihin on mallinnettu TCP:n keskeisiä ominaisuuksia, kuten yhteyden muodostaminen ja sulkeminen.

Solmut koostuvat moduuleista, joita on viittä eri tyyppiä:

- **Suoritinmoduulit**
- **Jonomoduulit**
- **Ulkoisten järjestelmien moduulit**
- **Lähetysmoduulit**
- **Vastaanottomoduulit**

Suoritinmoduulit

Suoritinmoduulit ovat Modelerin tärkeimpiä moduuleita ja niiden avulla voidaan mallintaa mm. eri protokollat ja pakettivirtoihin liittyvät toimenpiteet. Niiden tehtäviin kuuluu pakettien kehystäminen ja purkaminen viiveineen. Suoritinmoduulit on täysin määritelty niiden tilakaavioiden avulla, joten ne ovat monikäyttöisiä ja muokattavissa eri mallinnustarpeisiin sopivaksi. Kuvan 3.2 TCP-moduuli on suoritinmoduuli.

Jonomoduulit

Jonomoduulit eivät eroa suoritinmoduuleista muuten kuin, että niihin on jonojen mallintamista helpottamaan lisätty alijonoja. Alijonojen kokoa voidaan tarvittaessa rajoittaa mallinnuskohteen mukaan ja määritellä miten jonon paketteja käsitellään sen täyttyessä.

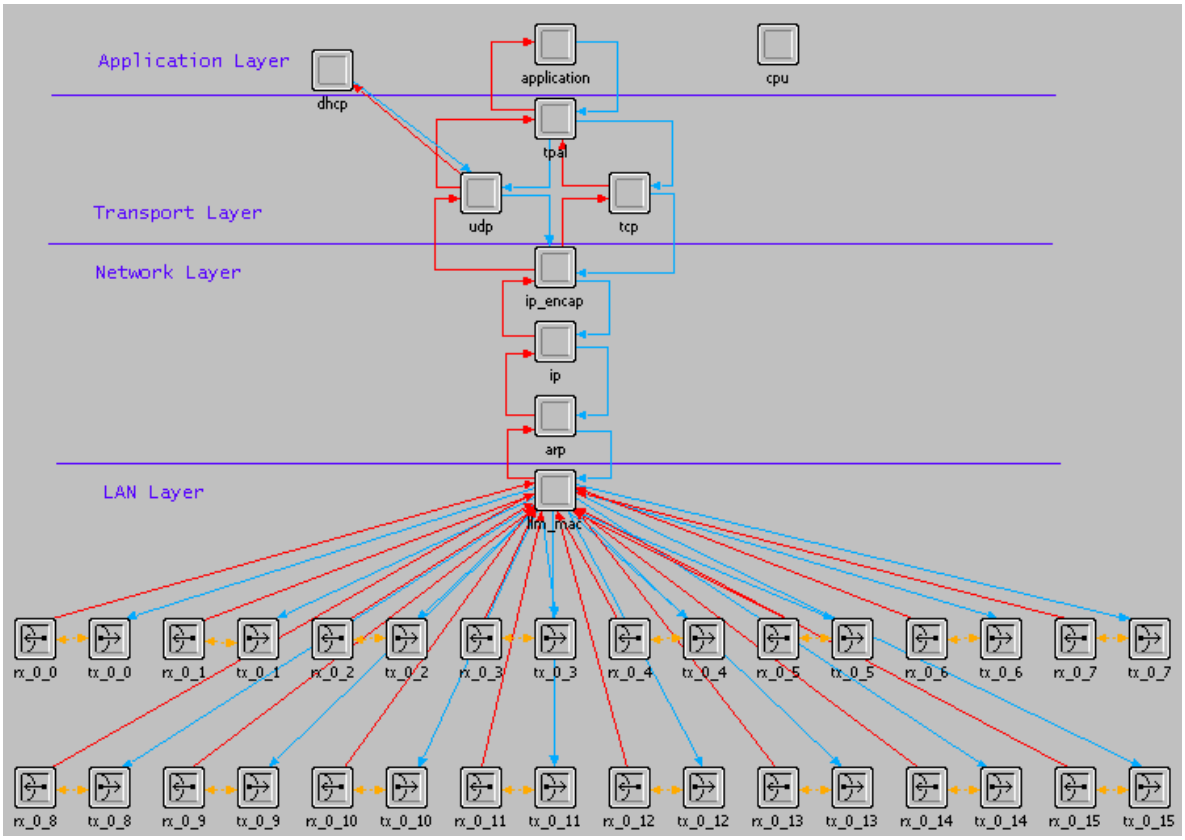
Ulkoisten järjestelmien moduulit

Ulkoisten järjestelmien moduulit ovat jonomoduuleita, joihin on lisätty kyky toimia rajapintana simulaation ulkopuolisiin järjestelmiin. Simulaatioon voidaan siten liittää toinen simulaatio tai System-In-The-Loop (SITL) -laajennus.

Lähetys- ja vastaanottomoduulit

Lähetys- ja vastaanottomoduulit toimivat solmussa rajapintana muihin solmuihin. Solmutasolla ne toimivat pakettien lähtö- ja päätepisteinä, eli lähetysmoduuliin vain saapuu paketteja muista moduuleista muihin solmuihin lähetettäviksi ja vastaanottomoduulista vain lähtee paketteja muihin moduuleihin käsiteltäviksi. Toisaalta verkkotasolla näihin

moduuleihin kiinnittyvät solmujen väliset linkit. Jokainen kaksisuuntainen yhteys käyttää yhden lähetys- ja vastaanottomoduulin, joten jokaiselle mallinnettavalle yhteydelle on luotava omat moduulinsa.



Kuva 3.3 Local Area Network (LAN) -solmun toteutus moduuleilla

Kuvassa 3.3 on esitetty lähiverkon mallinnus moduuleita apuna käyttäen. Toteutus on Modelerin mallikirjastosta ja sitä käytettiin luvussa 4 esiteltävissä mittauksissa. Moduulit on järjestetty kuvassa Open Systems Interconnection (OSI) -mallin mukaisesti siten, että moduulit ovat järjestyksessä sovelluskerros, kuljetuskerros, verkkokerros ja fyysinen kerros ylhäältä alas luettuna. Verkkokerroksen ja sitä ylempien kerrosten protokollat ja toiminnot on mallinnettu suoritinmoduulien avulla. Fyysisellä kerroksella on lähetys- ja vastaanottomoduuleita yhteensä 15 paria, joten LAN-solmuun voidaan liittää 15 kaksisuuntaista linkkiä. Moduuleitten välillä olevat yksisuuntaiset nuolet kuvaavat paketti- ja datavirtoja, joihin välikerrosten suoritinmoduulit tekevät muutoksia tai lisäävät viiveitä. Vastaanottomoduuleista lähtevät punaiset nuolet solmun ylempiin kerroksiin. ”application”-moduuli mallintaa sovelluskerroksen protokollien kuten File Transfer Protocol (FTP) - ja Hypertext Transfer Protocol (HTTP) [RFC959, RFC2616] käyttöä.

Verkkotasolla mallinnetaan varsinainen tietoverkko käyttäen itse mallinnettuja tai valmiita verkon solmuja, kuten LAN-solmu. Verkkotasolla käytettävissä on solmumallien lisäksi linkkimalleja ja erikoismalleja, joita tarvitaan esimerkiksi verkon vikaantumisen mallintamiseen.

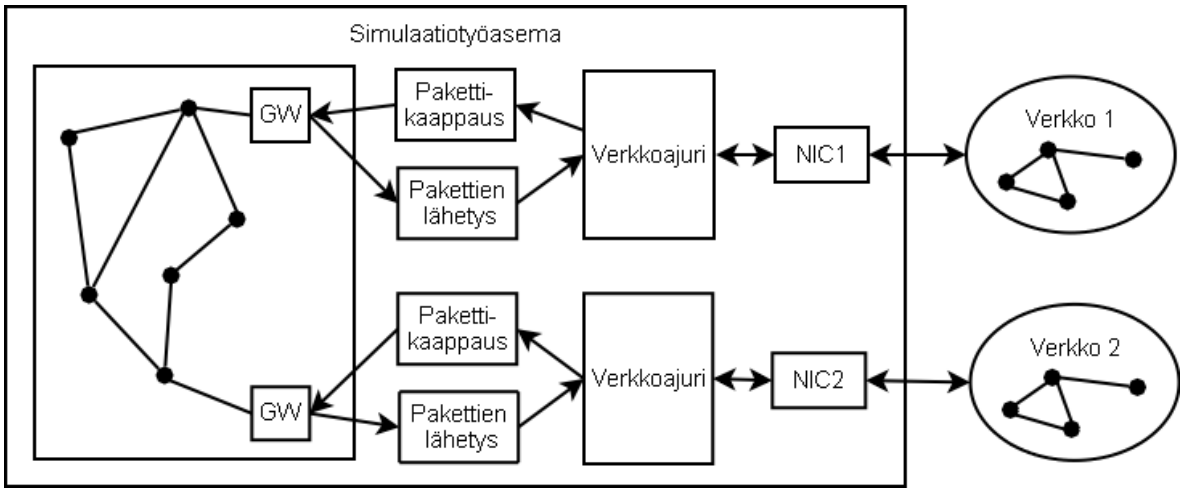
Broeck ja kumppanit [Bro02] ovat pyrkineet validoimaan Modelerin yhden reitittimen mallin eli osoittamaan, että malli on realistinen. He mittasivat reaalisesta Ciscon 2621-reitittimestä prosessointiviiveen, joka kuluu reitittämiseen ja joka on jokaiselle reititintyypille yksilöllinen. Modelerin malleissa oletuksena oleva prosessointiviive ei välttämättä ole realistinen. Mittaamansa viiveen he määrittelivät Modelerin vastaavaan reititinmalliin ja tekivät vertailumittaukset simuloidulla ja reaalisella verkolla. Verkko koostui vain yhdestä reitittimestä ja kahdesta työasemasta. Prosessointiviiveen määrittämiseksi heidän piti määritellä yksinkertaiselle verkolle viive yhteen suuntaa eli työasemalta toiselle. Jotta viive yhteen suuntaan voitiin mitata, työasemien piti olla synkronoitu eli niiden kellojen käyvän samassa ajassa. Työasemat synkronoitiin vertailuliikennevirralla, minkä takia yhdensuuntaisen viiveen määrittäminen oli hieman epätarkkaa. Kyseisellä menetelmällä he saivat kuitenkin riittävän tarkan arvon prosessointiviiveelle. Vertailumittaukset reaalisella ja simuloidulla verkolla osoittavat, että reititinmalli on realistinen, kun malliin määritellään realistinen prosessointiviive. Heidän ratkaisunsa luotettavaan simulointiin ei kuitenkaan ole skaalautuva suurempiin heterogeenisiin verkkoihin, jossa on useita erilaisia laitetyyppejä erilaisine suoritusarvoineen. Kirjavan laitekannan suoritusarvojen selvittäminen etukäteen mittaamalla saattaa joissain tapauksissa olla liian työläs tai jopa mahdoton tehtävä. Tällöin on tyydyttävä laitevalmistajan ilmoittamiin suoritusarvoihin ja otettava huomioon niiden mahdollinen epäluotettavuus simulaation luotettavuutta kokonaisuudessaan määriteltäessä.

3.3 Reaalisen tietoverkon ja simulaation liittäminen

Seuraavaksi esitellään vaihtoehtoja hybridiverkon rakentamiseen.

3.3.1 OPNET System-in-the-Loop

Modeler-työkaluun on kehitetty System-in-the-Loop (SITL) -laajennus, joka toimii rajapintana simulaation ja oikeiden laitteiden välillä. Sen avulla simulaatioon voidaan liittää yksi tai useampi oikea verkon laite kuten reititin tai työasema. Liitettävien laitteiden määrää rajoittaa vain simulaattorityöasemaan saatavien verkkoyhteyksien määrä ja simulaattorin suorituskyky.



Kuva 3.4 Verkkojen liittäminen simulaatioon

Kuvassa 3.4 on esitetty kahden verkon liittäminen simulaatioon. Kuvan verkot 1 ja 2 voivat olla samaa verkkoa tai jopa sama laite. Simulaattorityöasema kytketään reaalisiin verkkoihin Ethernetin avulla. Tässä tapauksessa simulaattorityöasemassa on simulaation kannalta käytössä kaksi erillistä verkkokorttia (Network Interface Controller – NIC). Simulaatiossa näitä kahta verkkokorttia edustaa kaksi liityntää (Gateway – GW) simulaation ja reaalimaailman välillä. Kummallekin liittymälle tulee määritellä käytettävän verkkokortin Medium Access Control (MAC) -osoitteet. Myös niihin kytkettävien laitteiden MAC-osoitteet tulee määritellä, tässä tapauksessa verkkojen 1 ja 2 ne laitteet, jotka ovat suoraan kytkettynä simulaattorityöasemaan.

Modeler poistaa Ethernet-otsakkeen saapuvista paketeista, joten GW:lle paketit saapuvat Internet Protocol (IP) -paketteina. SITL osaa käsitellä IPv4 ja IPv6 paketteja, mutta pakettien pilkkoutumista (*fragmentation*) ei tueta. Internet Control Message Protocol (ICMP) - ja ICMPv6-viestejä voidaan käyttää simulaation ja reaalisten laitteiden välillä. OSI-mallin ylempien kerrosten protokollista SITL:n kanssa voidaan käyttää Transmission

Control Protocollaa (TCP) ja User Datagram Protocollaa (UDP). Reititysprotokollista on tuettuna Open Shortest Path First (OSPF), Routing Information Protocol (RIP) -v1 ja RIPv2. Simulaatioissa, joissa vain reititetään IP-paketteja simuloidun verkon läpi reaalisesta verkosta toiseen, voidaan IP:n päällä käyttää mitä tahansa OSI-mallin sovelluskerroksen protokollia, koska silloin simulaation ei tarvitse käsitellä niitä sovellustasolla.

SITL-laajennus on suhteellisen uusi ja tutkimukseen vielä vähän käytetty verrattuna esimerkiksi NS2:n vastaavaan laajennukseen. Wellington ja Kubischta [Wel03] olivat ensimmäisiä käyttämään SITL:n esiversiota tutkimukseen. He käyttivät OPNET:ia ja SITL:ia muodostamaan hybridin simuloidusta langattomasta verkosta ja reaalisista verkon solmuista. Heidän ratkaisussaan implementoitiin lisäksi erillinen Real-time Controller -moduuli, joka huolehti simulaation pysymisestä reaaliajassa ja koko verkon liikenne kulki sen kautta. He määrittivät viiveen, jonka verran simulaatio oli jäljessä reaaliaikaa, eli minkä verran myöhässä paketit keskimäärin lähtivät simulaation takia. Mittauksien mukaan viive jäi alle 2,5 prosentin tutkitun verkon viiveen minimistä, joka aiheutui paketin kulkiessa toisesta verkon laidasta toiseen. Verkon koon muuttaminen tietysti vaikuttaa simulaation aiheuttamaan viiveeseen, kuten myös simulaatiotyöaseman tehokkuus. Wellington ja Kubischta käyttivät 1,7 GHz:n kellotaajuudella toimivaa simulointityöasemaa.

3.3.2 Tietoverkon emulointi Network Simulator 2:lla

Oikean tietoverkon ja NS2-simulointityöaseman liittämisestä käytetään yleisesti termiä ”tietoverkon emulointi”. Tämä on toiminnallisuudeltaan ja periaatteeltaan samankaltainen kuin Modeleriin kehitetty SITL. Emulointi on implementoitu NS2:een muutamilla muutoksilla alkuperäisen ohjelmiston toimintaan. Näihin kuuluvat mm. muutokset tapahtumalistan suoritukseen sekä osoitteistuksen ja reitityksen muutokset. [Fal99]

Muutokset tapahtumalistan käsittelyyn on implementoitu, jotta saadaan simulaattorille kuvan 2.5 mukainen toiminta. Tämä on toteutettu siten, että tapahtumien suoritukseen on lisätty viive, jonka avulla tapahtumat tulevat suoritukseen oikeaan aikaan. [Fal99]

3.3.3 Muita sovelluksia reaalisen ja simuloidun tietoverkon liittämiseen

OPNET:n ja NS2:n lisäksi on muitakin ratkaisuja hybridiverkkojen rakentamiseen. Useat tahot ovat kehittäneet omia hybridisimulointiin kykeneviä simulaattoreita soveltumaan juuri omiin tarpeisiinsa [Gio05, Zhe04, Mac03, Kid05]. Zheng ja Ni [Zhe03, Zhe04] ovat kehittäneet simulaattorin, jolla simuloitavan verkon topologia voidaan jakaa simuloitavaksi eri työasemilla. Siten voidaan emuloida suurempaa verkkoa, koska työtaakka jakaantuu eri työasemille. Heidän ratkaisussaan simulointityöasemat liitettiin yhteen kytkimellä, joka oli ainoa reaalinen verkon laite ja muu verkko simuloitiin hajautetusti eri työasemissa. Heidän mittausten mukaan yhden työaseman simuloidessa neljää jonoon kytkettyä verkon solmua, emulaation maksimiläpäisyksi jää noin 55 Mbit/s, vaikka verkkoon tarjoaisi liikennettä nopeammin. Simulaatiosta johtuva viive pakettien lähetyksissä jäi kuitenkin neljällä simuloitavalla solmulla noin 100 mikrosekuntiin, jolla ei ollut merkitystä kymmeniä millisekunteja olevaan verkon kokonaisviiveeseen.

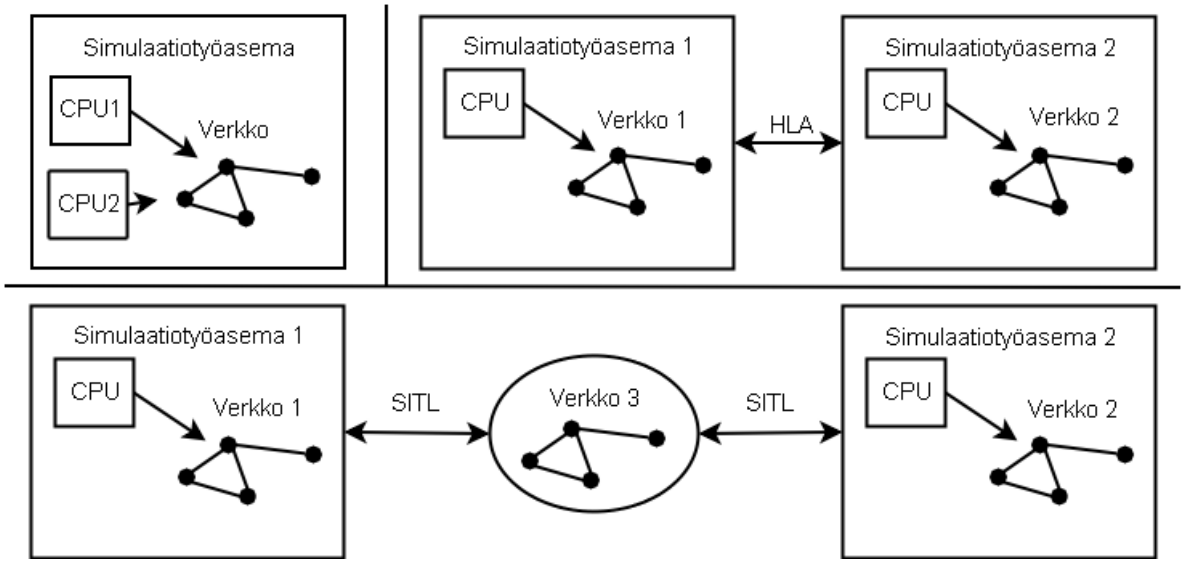
Kiddle, Simmonds ja Unger [Kid05] käyttivät tutkimuksissaan myös simuloinnin hajauttamista, mutta lisäsivät vielä tehokkuutta abstrahoimalla osan liikenteestä virraksi (*flow*). Tavallisesti diskreetissä tapahtumapohjaisessa simulaatiossa (DES – Discrete Event Simulation) jokainen paketin lähetys ja vastaanotto generoi uuden tapahtuman. Tällöin simulaatiosta saadaan tarkempi ja realistisempi, mutta mallintamalla ainakin osa liikenteestä virraksi voidaan simuloida suurempia tietoverkkoja, koska vain muutokset virtaan generoivat tapahtuman ja muuttumaton virta ei aiheuta toimenpiteitä. Muutoksia virtaan tapahtuu joka tapauksessa vähemmän kuin sama liikenne aiheuttaisi paketin lähetys- ja vastaanottotapahtumia, joten laskentatehoa säästyy.

Lankaverkkojen lisäksi on emuloitu myös langattomia tietoverkkoja [Gio05, Mac03]. Langaton tiettyä tarkoitusta varten muodostuva tietoverkko (Mobile Ad hoc Network – MANET) tuo omat haasteensa simulointiin. Liikkuvuuden emulointia on yleensä pyritty toteuttamaan moniosaisilla järjestelmillä, jossa langattomuus on eriytetty omaksi osakseen ja se on erillään muusta simulaatiosta.

3.4 Simulaation hajauttaminen

Simulaatio voidaan hajauttaa eri työasemille tai työasemien eri suoritinytimille, jotta suurten verkkojen reaaliaikainen simulaatio tulisi mahdolliseksi.

Kuvassa 3.5 on esitetty yleisiä simulaation hajauttamisvaihtoehtoja. Vasemmalla ylhäällä on simulaation hajauttaminen kahdelle suoritinytimelle eli ns. rinnakkaissimulointi. Oikealla ylhäällä on kuvattuna simulaation hajauttaminen kahdelle eri työasemalle, jolloin on käytettävä jotain vuorovaikutusrajapintaa näiden kahden simulaatioajon välillä. Yleinen käytetty rajapinta on High Level Architecture (HLA). Alhaalla on kuvattuna hajauttaminen SITL:n avulla.



Kuva 3.5 Eri vaihtoehtoja simulation hajauttamiselle

Thoppian ja kumppanit [Tho06] ovat pyrkineet selvittämään rinnakkaissimuloinnin tuomia etuja reaaliaikasimulointiin Modelerilla. He tekivät vertailumittauksia useammalle suorittimen ytimelle hajautetusta simulaatiosta ja tavallisen sarjasuoritukseen yhdellä ytimellä perustuvan simulaation kanssa. Simulointiajoja ei tehty kuitenkaan reaaliajassa vaan tavallisena DES-ajona, jolloin tapahtumien välille ei lisätty tyhjäkäyntiä. Eri ajojen suoritusnopeudesta voitiin siten päätellä, mikä soveltuisi parhaiten reaaliaikasimulaatioihin. Tulokset olivat kuitenkin yllättäviä; heidän mittausten mukaan rinnakkaissimulointiajot suoriutuivat samasta tehtävästä hitaammin kuin tavallinen sarjasuoritteinen simulaatio. Tähän voi olla syynä se, että Modelerin mallikirjaston kaikki mallit eivät ole suunniteltuja rinnakkaissimulointiin [Opn06] ja simulaatioissa käytetyt mallit oli kehitetty nimenomaan

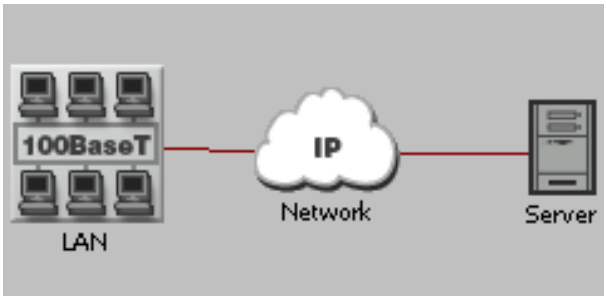
sarjasuoritteiselle simulaatiolle. Rinnakkaisajosta ei ole hyötyä, jos muut ytimet joutuvat odottamaan yhden ytimen suorituksen tuloksia tai suoritukseen tarvittavien resurssien vapautumista merkittävän osan ajasta. Siksi mallit tuleekin implementoida rinnakkaisajoa silmälläpitäen, jos halutaan hyötyä usean ytimen simulaatiotyöasemista.

4. Mittaukset

Tässä luvussa esitetään mittausjärjestelyt. Mittausten päämääränä oli vertailla simuloitua, osittain simuloitua ja oikeaa tietoverkkoa. Mittaukset suoritettiin Teknillisen Korkeakoulun tietoverkkolaboratoriossa, luvun 4.5 käsittelemää laskentatehon vaikutusta lukuun ottamatta. Luvussa 5 käsitellään mittauksien tuloksia.

4.1 Mittalaitteet

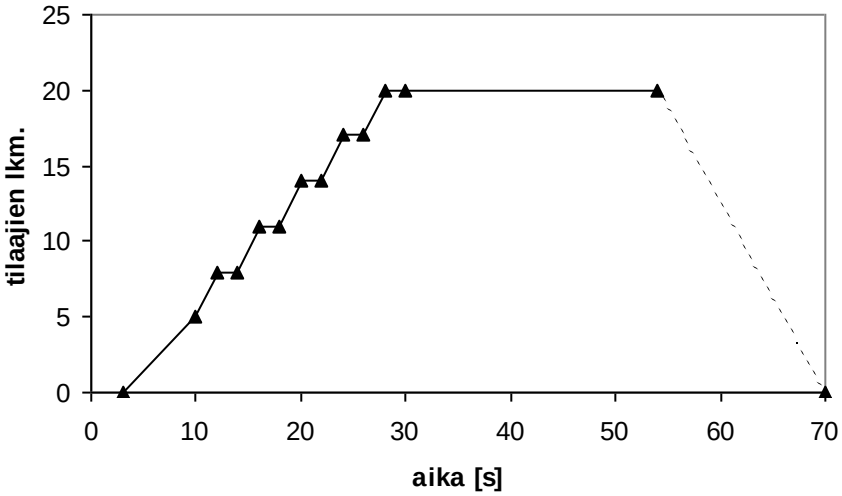
Mittalaitteena käytettiin Spirent Communications Avalanche ja Reflector [Spi06] paria. Laitteet injektoivat verkkoon liikennettä toimien verkossa tilaajina ja palvelimina.



Kuva 4.1 Mittalaitteiden havainnolliskuva:
Avalanche toimii LAN:ina ja Reflector Serverinä

Kuvassa 4.1 on esitetty mittalaitteiden roolit rakennetussa ja simuloidussa verkossa. Avalanche toimii verkossa tilaajina lähiverkossa (LAN - Local Area Network) ja generoi liikennettä IP-verkon yli palvelimelle. Reflector voi toimia useampana palvelimena, mutta näissä mittauksissa se asetettiin toimimaan yhtenä laitteena. Kuvan 4.1 IP-verkkoa edustaa rakennettu verkko reaalisen tietoverkon tapauksessa ja simuloitu verkko hybridiverkon tapauksessa. Täysin simuloidun verkon tapauksessa mittalaitteita ei käytetty, vaan liikenne generoitiin simuloimalla ja tutkittiin simulaattorin keräämiä suoritusarvoja.

Kaikki mittaukset suoritettiin samoin tavoin eri verkoille muutamin eri vaihein. Mittausajaksi asetettiin 70 sekuntia kuhunkin vaiheeseen. Mittausaika oli syytä pitää kohtuullisen pienenä, koska eri mittauksia oli useita. 70 sekuntia on kuitenkin riittävä, jotta tarkasteltavia tuloksia saadaan tarpeeksi. Suuremmalla mittausajalla olisi saatu mittausdataa jopa liikaa, ettei sitä olisi voitu analysoida järkevästi. Palvelimia oli joka mittauksessa yksi.



Kuva 4.2 Mittalaitteiden verkkoon injektoima kuorma (tilaajat) ajan funktiona

Useamman palvelimen käyttäminen näissä mittauksissa olisi ollut tarpeetonta ja monimutkaistanut mittauksia turhaan. Tilaajien lukumäärä ajan suhteen oli kaikissa mittauksissa sama: mittauksen alkuhetkenä 0 ja kasvoi portaittain 20 tilaajaan. Kuvassa 4.2 on esitetty tilaajien lukumäärä ajan funktiona. Palvelun nopeudesta riippuen 70 sekunnin aikana saatettiin palvella tuhansia tilaajia; järjestelmässä oli siis korkeintaan 20 tilaajaa samaan aikaan. Kun yksi tilaaja saatiin palveltua, mittalaitteet generoivat uuden. 20:n yhtäaikaisen tilaajan ajateltiin olevan sopiva määrä verkon kuormittamiseksi. Yhdellä tilaajallakin olisi saatu verkon koko kapasiteetti käyttöön, mutta useammalla tilaajalla saatiin realistisempi tilanne verkon rasituksesta, jolloin useampi tilaaja jakaa käytössä olevan kapasiteetin. Tilaajien määrää kasvattamalla vielä 20:stä yksittäisen tilaajan saama kapasiteetti olisi pienentynyt, mutta tilanne ei olisi muuttunut verkon kannalta. 54 sekunnin jälkeen uusia tilaajia ei enää generoitu verkkoon, vaan sillä hetkellä verkossa olleet tilaajat palveltiin loppuun.

4.2 Mittauksen eri vaiheet

Eri verkoille tehtiin mittauksia kahdella eri Open Systems Interconnection (OSI) -mallin [ISO7498] sovelluskerroksen protokollalla. File Transfer Protocol (FTP) - ja Hypertext Transfer Protocol (HTTP) -protokollilla [RFC959, RFC2616] injektoitua liikennettä mitattiin muutamilla eri asetuksilla. Koska jo ennen mittauksia oli tiedossa, että reaaliaikainen simulointi saattaa olla liian raskasta, liikenteen läpäisyä rajoitettiin

mittalaitteiden palvelinpäästä muutamissa mittauksissa. Taulukossa 4.1 on esitetty kunkin mittauksen injektoima kuorma ja liikenteen rajoitus palvelimelta. Liikenteen rajoitus toteutettiin siten, että palvelin injektoi verkkoon vain rajoituksen määrittämän maksimimäärän liikennettä lisäämällä viivettä pakettien lähetykseen tilaajille. FTP-mittauksissa kukin tilaaja latasi 100 kilotavun kokoisen tiedoston ja HTTP-mittauksissa yhden tyhjän Hypertext Markup Language (HTML) -sivun. Tyhjän sivun lataamisella haluttiin tarkastella tilannetta, joka eroaa täysin FTP-mittauksista; verkossa ei ole juurikaan hyötyliikennettä ja tilaajien palvelu erittäin nopeaa, jolloin palveltujen tilaajien kokonaismäärä kasvaa suureksi.

Mittaus	Kuorma/Tilaaja	Rajoitus
HeavyFTP	Tiedosto ~100kt	-
MediumFTP	Tiedosto ~100kt	1Mbit/s
Low FTP	Tiedosto ~100kt	500kbit/s
HeavyHTTP	Tyhjä sivu	-
MediumHTTP	Tyhjä sivu	1Mbit/s
LowHTTP	Tyhjä sivu	500kbit/s

Taulukko 4.1 Mittausvaiheet

Lisäksi tehtiin mittaukset käyttäen simuloinnissa tehokkuutta lisääviä yksinkertaistuksia ja tarkasteltiin verkon toimintaa vikatilanteessa. Näitä mittauksia ja niiden tuloksia käsitellään luvussa 5.

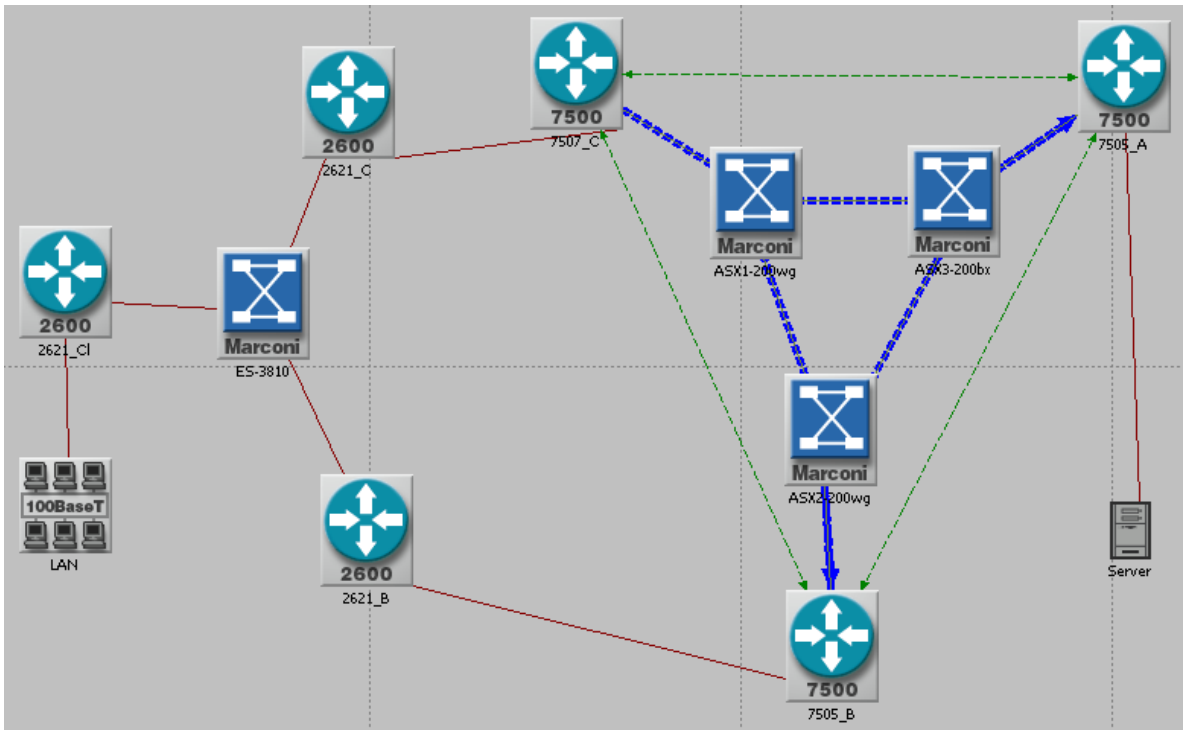
4.3 Reaalinen tietoverkko

Oikea verkko rakennettiin käyttäen Asynchronous Transfer Mode (ATM) -tekniikkaa [Gru97] keskellä ja Ethernet-verkkoa laidoilla. Mittalaiteet, jotka toimivat tilaajina ja palvelimena, kytkettiin verkon reunoille. Laboratorion tutkimusympäristöön sopivasti IP-osoiteavaruudeksi valittiin ”10.112.16.0-10.112.31.0”. Kaikkien ATM-reitittimien välille asetettiin Permanent Virtual Circuit (PVC) -reitit ja ATM-verkko muodostettiin Classical-IP-Over-ATM-menetelmällä [Cis00, RFC1577]. ATM-reitittimet on nimetty kuvissa 4.3 ja 4.4 tunnisteilla 7505_A, 7505_B ja 7507_C niiden tyyppimerkkien mukaan. ATM-laitteet kytkettiin toisiinsa Optical Carrier (OC) -3-kuitujen avulla. Ethernet-linkkien nopeus oli kaikissa laitteissa 100Mbit/s, paitsi ATM-reitittimillä 10Mbit/s, joten verkon teoreettinen maksimiläpäisy tilaajan ja palvelimen välillä oli 10Mbit/s. Verkossa käytettiin OSPF-reititysprotokollaa [RFC2328].

Laite	Lkm.	Tehtävä
Cisco 2600	3	Ethernet-reititin
Cisco 7500	3	ATM-reititin
Fore ASX-200	3	ATM-kytkin
Fore ES-3810	1	Ethernet-kytkin

Taulukko 4.2 Verkossa käytetyt laitteet

Taulukossa 4.2 on listattuna mittauksissa käytetyt laitteet ja niiden tehtävä verkossa. Ciscon 7500-sarjan reitittimet toimivat sekä ATM-, että Ethernet-reitittiminä.



Kuva 4.3 Rakennetun verkon havainnolliskuva mittalaitteineen

Kuvassa 4.3 on esitetty rakennettu oikea verkko. Verkon topologia pidettiin kaikissa mittauksissa samana, mutta mittauksesta riippuen laitteet olivat joko reaalisia tai simuloituja. Seuraavaksi esitellään ja perusteellaan verkkoon valitut tekniikat.

Classical-IP-Over-ATM

ATM valittiin tekniikaksi verkon keskelle, jotta verkkoon tulisi riittävästi kompleksisuutta eikä sen simuloiminen olisi liian suoraviivaista. ATM on OSI-mallin siirtokerroksen protokolla, joka perustuu virtuaalipiireihin. Jotta virtuaalipiirikytkentäistä ATM-tekniikkaa voidaan käyttää pakettikytkentäisten IP-verkkojen osana, tarvitaan menetelmä näiden kahden tekniikan yhteensovittamiseksi. Yksi menetelmä IP:n käyttöön ATM-verkon yli on

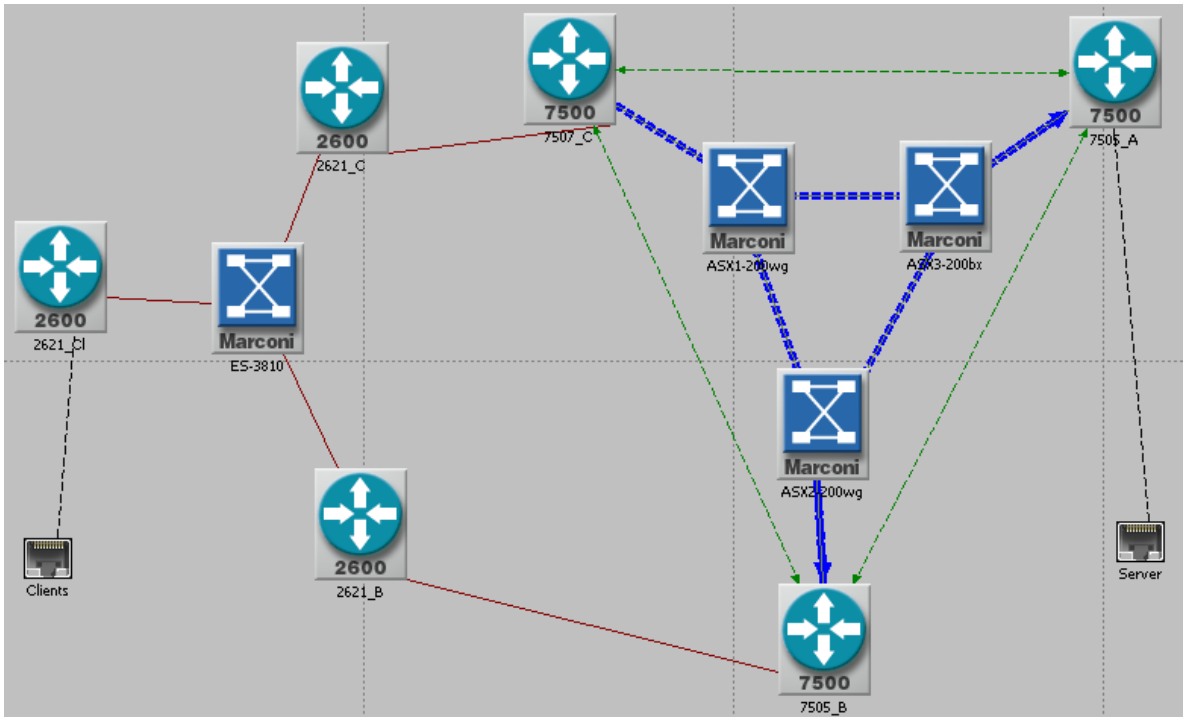
Multiprotocol Over ATM (MPOA) [Cis00]. Classical-IP-Over-ATM on toteutukseltaan yksinkertaisempi ja sen heikkoutena on eri IP-aliverkoissa sijaitsevien solmujen välisen liikenteen kulkeminen aina reitittimen kautta. Vaikka kaksi eri IP-aliverkossa sijaitsevaa solmua olisi samassa fyysisessä ATM-verkossa, kaikki niiden välinen liikenne pitää kulkea reitittimen kautta, toisin kuin MPOA:ssa. Näihin mittauksiin Classical-IP-Over-ATM soveltui kuitenkin mainiosti, koska ATM-verkko oli kohtuullisen pieni ja koko ATM-verkko asetettiin samaan IP-aliverkkoon.

Open Shortest Path First

Open Shortest Path First (OSPF) valittiin verkkoon reititysprotokollaksi. OSPF-protokollalla on joitakin etuja verrattuna esimerkiksi Routing Information Protocol (RIP) -reititykseen [RFC2453] verrattuna, kuten parempi skaalautuvuus suurempiin verkkoihin. RIP olisi toiminut mainiosti tämän työn kokoisessa tietoverkossa, mutta haluttiin tutkia nimenomaan simulointia OSPF-protokollalla, jota voisi käyttää suurempien verkkojen simulointiin. OSPF-protokollalla reitittimet lähettävät toisilleen Link State Database (LSDB) -taulun, johon on kerättynä osoitetiedot eri linkeiltä. Linkeillä on oman hintansa (*cost*), joiden perusteella tehdään reitityspäätös. Hinta määräytyy linkin kaistanleveyden (*bandwidth*) perusteella.

4.4 Hybriditietoverkko

Hybridiverkko rakennettiin samaan tapaan OPNET Modeler -simulointityökalulla, jossa oli erillinen SITL-laajennus. Mittauksissa käytettiin Modelerin mallitietokannan valmiita malleja simuloimaan kaikkia oikean verkon laitteita, mittalaitteiden edustamat tilaajat ja palvelin poisluettuna. Mallinnettava verkko asetettiin täysin samoin kuin oikea verkko IP-osoitteineen ja PVC-reitteineen. Mittalaiteet kytkettiin nyt simulointityöasemaan siten, että simuloitu verkko vastasi aikaisemmin mittalaitteiden välissä ollutta oikeaa verkkoa reitittimineen ja kytkimineen. Modeler-ohjelmistosta ja SITL-laajennuksesta käytettiin versiota 12.0 PL1 ja kehityskerneliä.



Kuva 4.4 Osittain simuloidun verkon simuloitu osuus mittalaitteiden kytkentäpisteineen: Clients kytkettiin Avalancheen ja Server kytkettiin Reflectoriin

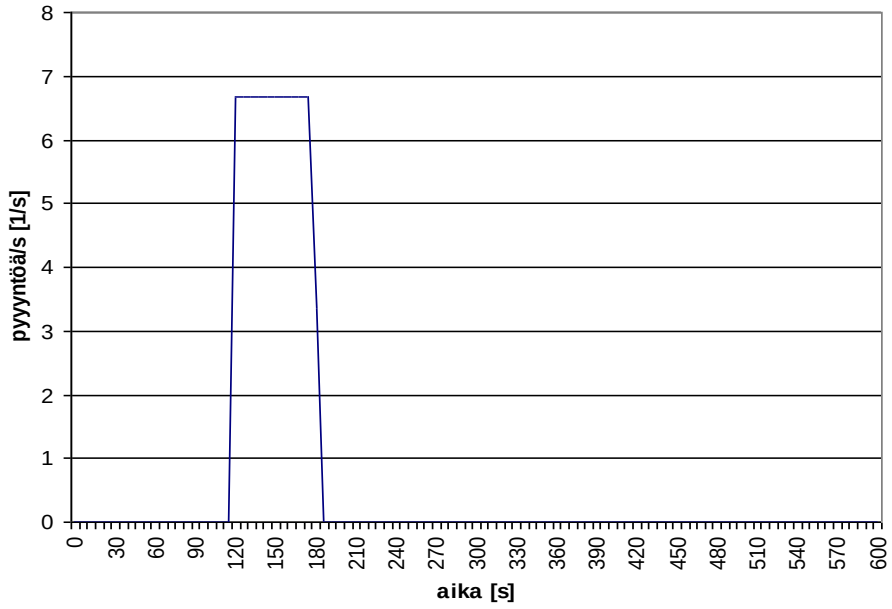
Kuvassa 4.4 on esitetty osittain simuloidun verkon topologia. Tässä vaiheessa simuloitiin koko verkko tilaajia ja palvelinta lukuun ottamatta. Simulointityöasemana käytettiin työasemaa, jossa oli Intelin Pentium 4 2,8 GHz suoritin, 3,24 Gt keskusmuistia ja käyttöjärjestelmänä Windows XP.

4.5 Täysin simuloitu tietoverkko

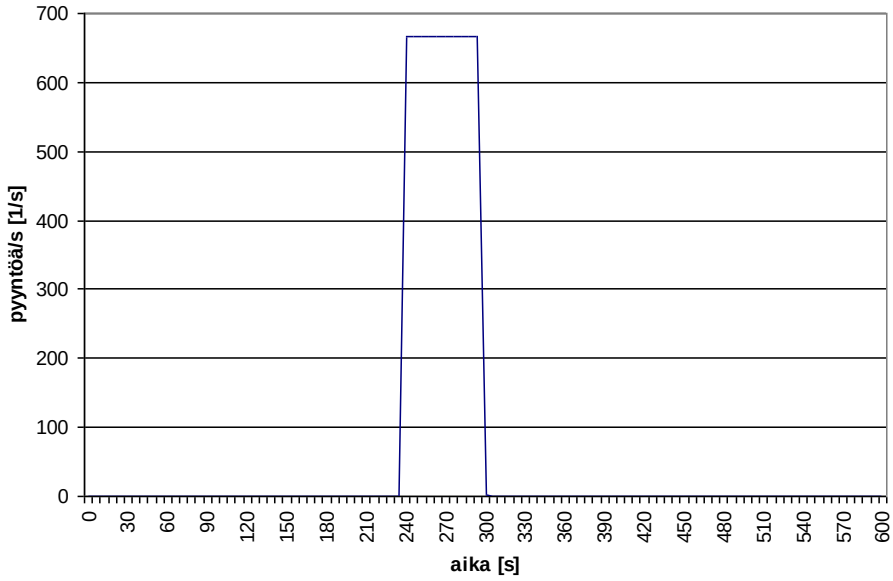
Täysin simuloitu tietoverkko toteutettiin kokonaan OPNET Modeler-simulointiohjelmistolla ja samoilla ohjelmistoversioilla kuin hybridiverkko. Mittalaitteita ei siis kytketty, vaan liikenne generoitiin simulaatiossa ja seurattiin muutamia suoritusarvoja, joita voidaan verrata reaaliseen ja hybridiverkkoon. Näitä suoritusarvoja käsitellään luvussa 5. Verkon topologia oli sama kuin kaikissa vertailumittauksissa eli kuvan 4.3 esittämä topologia. Tietoverkon ja sen liikenteen täydellinen simulointi tehtiin, jotta saataisiin tietoa miten hyvin sen tuottamat tulokset vastaisivat oikeaan verkkoon kytkettyjen mittalaitteiden tuloksia. Näin voitaisiin erotella syyt mahdollisiin eroihin hybridiverkon ja oikean verkon tuloksissa. Mikäli täysin simuloidun verkon tulokset vastaisivat paremmin oikeaa, erot hybridiverkolla johtuisivat reaaliaikasimulaation aiheuttamista ongelmista eivätkä niinkään epärealistisista malleista.

Verkossa simuloitiin liikennettä, jonka lähiverkossa olevat 20 työasemaa ja palvelin välilleen generoivat. Täysin simuloituun verkkoon liikenteen määrittäminen tapahtuu eri tavalla, kuin edellä käytettyjen mittalaitteiden tapauksessa. Liikenteen generointi määriteltiin siten, että jokainen työasema lähettää FTP-pyyntöjä 100 kilotavun tiedostosta 3 sekunnin välein minuutin ajan ja tämän jälkeen HTTP-pyyntöjä tyhjästä sivusta 30 millisekunnin välein minuutin ajan. Tämä liikenteen generointitapa eroaa merkittävästi mittalaitteiden vastaavasta, jossa pyyntöväliä ei määritelty vaan mittalaitteet pyrkivät pitämään jatkuvasti määritellyn määrän tilaajia palvelimella kuormana. Mittalaitteissa liikenne määriteltiin siis tilaajien hetkellisenä lukumääränä, ei pyyntöjen intervalliaikoina kuten Modelerissa. Intervalliajat valittiin oikean verkon mittaustulosten perusteella vastaamaan mittalaiteiden generoimaa maksimikuormaa.

Kuvassa 4.5 on esitetty liikennemallin generoimien FTP-pyyntöjen lukumäärä sekuntia kohden. Pyyntöjen lukumäärä kuvaa yhden tilaajan lähettämiä pyyntöjä, joten kuvaaja ei vastaa pyyntöjen kokonaismäärää, jonka 20 tilaajaa generoivat. Kuvassa 4.6 on esitetty tilaajien HTTP-pyynnöt, jotka generoitiin samaan tapaan kuin FTP-pyynnöt edellä. Molemmat liikenteen generoinnit suoritettiin saman simulaatioajon aikana.



Kuva 4.5 FTP-pyynnöt palvelimelle



Kuva 4.6 HTTP-pyynnöt palvelimelle

4.6 Laskentatehon vaikutus mittaustuloksiin

Reaali- ja hybridiverkkojen vertailujen lisäksi haluttiin tietoa simulaatiotyöaseman laskentatehon vaikutuksesta mittaustuloksiin. Kuvan 4.4 verkkoa simuloitiin kahdella eri suorituskyvyn omaavalla työasemalla. Liikennettä generoitiin hybridiverkkoon kahdesta työasemasta käyttäen Iperf-sovellusta [Tir03]. Vertailusimulointityöasemina käytettiin samaa aiemmin reaali- ja hybridiverkkojen vertailumittauksissa käytettyä Intelin Pentium 4-suorittimella varustettua työasemaa ja kahdella Intelin Xeon 5160-tuplaydinsuorittimella varustettua tehotyöasemaa, jossa oli 16 Gt keskusmuistia ja käyttöjärjestelmänä Windows XP 64. Vaikka Modelerin DES-ajo pystyy hyödyntämään 64-bittisen muistiosoiteavaruuden [Opn06], 64-bittisestä käyttöjärjestelmästä ja suuresta muistin määrästä ei tässä tapauksessa ollut hyötyä toiseen työasemaan nähden, koska molemmissa työasemissa muistia oli kuitenkin riittävästi kyseistä simulaatiota varten. Toisaalta Xeon-työaseman nopeampi muisti todennäköisesti vaikutti suorituskykyyn simulaatioissa. Simulaatiot ajettiin sarjasuoritteisesti molemmilla työasemilla eikä rinnakkaissimulointimahdollisuutta moniytimisellä työasemalla käytetty. Moniytiminen työasema saattoi hyötyä siitä, jos käyttöjärjestelmän toiminnot olivat yhdellä ytimellä ja simulaatioajo sai yhden ytimen kokonaan käyttöönsä.

Mittauksista kerättiin tietoa hybridiverkon läpäisymääristä ja mittauksia tehtiin kolmella eri pakettikoolla 512 tavua, 1024 tavua ja 1460 tavua. Pakettikoot valittiin siten, että kukin mahtuisi yhteen Ethernet-kehykseen. Iperf-sovellus generoi liikennettä Transmission Control Protocol (TCP) -protokollalla hyödyntäen verkon koko kapasiteetin. Sovellus lähetti verkon toiselta puolelta mainittuja pakettikokoja ja toiselta puolelta muutaman tavun mittaisia kuittausviestejä. Verkossa ei ennen liikenteen generointia ollut muuta liikennettä, kuin reititysprotokollien muodostama liikenne.

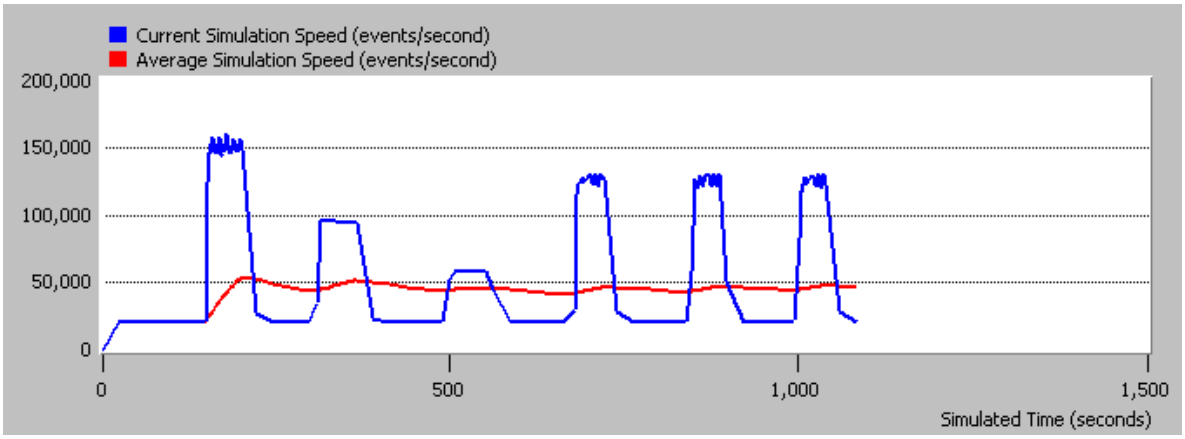
Kahden eri simulointityöaseman lisäksi verrattiin kahden eri simulaatiokernelin vaikutusta tuloksiin. Modeler-ohjelmistossa on kehityskernelin lisäksi optimoitu kerneli. Näillä kahdella tehtiin samat mittaukset, joten simulointiajoja kertyi yhteensä neljä, kun pakettikokoja vaihdettiin simulaatioajon aikana. Optimoidusta kernelistä on karsittu työkalut koodin korjaamiseen (*debugging*) [Opn06] ja sen avulla simulointi siten nopeampaa. Liikenteet generoitiin verkkoon siten, että reititysprotokollat ehtivät muodostaa reititystaulut simulaation käynnistyessä ja verkko tyhjentyä eri pakettikokomittausten välissä. Yhtä pakettikokoa lähetettiin 60 sekunnin ajan.

5. Tulokset

Tässä luvussa käydään läpi mittauksista saatuja tuloksia. Luvussa 6 analysoidaan tuloksia tarkemmin.

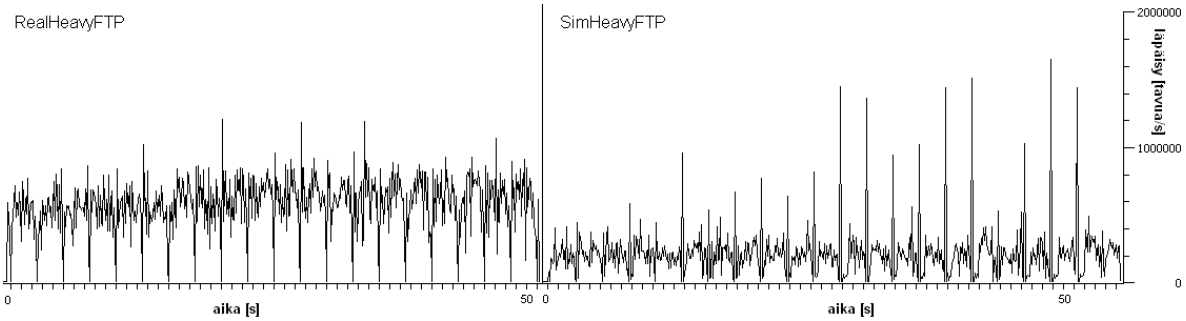
5.1 Reaalisen tietoverkon ja osittain simuloidun tietoverkon vertailu

Kaikki kuusi mittausta suoritettiin yhden hybridisimulaatioajon aikana peräkkäin noin minuutin välein järjestyksessä HeavyFTP, MediumFTP, LowFTP, HeavyHTTP, MediumHTTP ja LowHTTP. HeavyFTP-mittaus aloitettiin noin kahden minuutin simulaatioajon jälkeen, jotta OSPF ehti muodostaa reititystaulut kuhunkin reitittimeen. Simulaatioajon aikana voi seurata tapahtumien suoritusnopeutta reaaliaikaisesti. Hybridisimulaatioajon aikana voi myös tarkkailla, että simulaatio pysyy reaaliajassa, eikä jää jälkeen, jolloin syntyy virhettä tuloksiin. Tätä voidaan seurata tarkkailemalla simulaatiokelloa ja reaaliaikakelloa, joka kertoo simulaatioon kuluneen ajan. Reaaliaikasimulaatiossa näiden kellojen pitää olla samassa ajassa. Jos simulaatiokello ”jätättää”, simulaatiotuloksiin tulee virhettä. Myös hybridin toinen osapuoli eli reaalinen verkko, kuten mittalaitteet näissä mittauksissa, huomaa simulaation myöhästelyn. Ajon aikana oli havaittavissa, että simulaatiokello jäi ajoittain jälkeen reaaliajasta, mutta aika kuroutui kiinni mittausten välillä. Kuvassa 5.1 on esitettynä mittausten aiheuttamat tapahtumat simulaatioon. Ennen tulosten varsinaista tarkastelua voidaan todeta, että HeavyFTP-mittauksessa ja kaikissa HTTP-mittauksissa on virhettä, koska niissä simulaatiokello ei pysynyt reaaliajassa. Toisaalta MediumFTP- ja LowFTP-mittaukset pysyivät hyvin reaaliajassa. Tarkastelemme tässä vaiheessa kuitenkin vain mittalaitteiden antamia suoritusarvoja, jotka on taulukoitu taulukkoon 5.1, ja mittalaitteissa tehtyjä pakettikaappauksia.



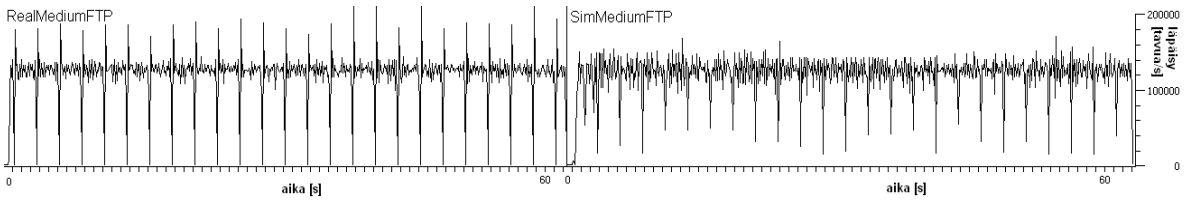
Kuva 5.1 SITL-simulaation aiheuttamat tapahtumat järjestyksessä: (6 kpl sinisiä pylväitä) HeavyFTP, MediumFTP, LowFTP, HeavyHTTP, MediumHTTP ja LowHTTP

Kuvissa 5.2-5.7 on esitettynä kaapattu pakettiliikenne molempiin suuntiin tilaajilla. Vasemmalla puolella kuvassa on reaalisen verkon ja oikealla hybridiverkon pakettikaappaus. Kuvissa pakettiliikenne on esitetty muodossa tavua/sekunti pystyakselilla ja aika sekunneissa vaaka-akselilla. Pakettikaappauksista saatiin liikenteen määrä ajan funktiona ja siten verrattua simuloidun ja oikean verkon läpäisyä sekä liikenteen profiilia.



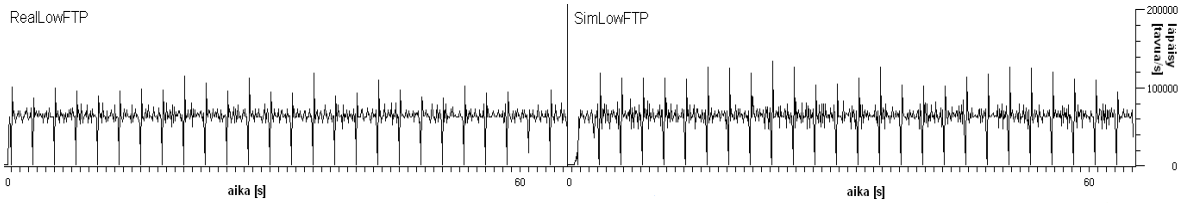
Kuva 5.2 Mittalaitteiden pakettikaappaukset: HeavyFTP – Reaalinen verkko vasemmalla (RealHeavyFTP) ja hybridi oikealla (SimHeavyFTP)

HeavyFTP-mittauksessa kuvassa 5.2 oikean verkon läpäisy on n. 1 Mt/s = 8 Mbit/s, joka on odotettu tulos teoreettiseen maksimiläpäisyyn 10 Mbit/s verrattuna. Hybridiverkon läpäisy jää selvästi oikean verkon vastaavasta. Pakettikaappauksesta laskettu läpäisy on n. 250 kt/s = 2 Mbit/s.



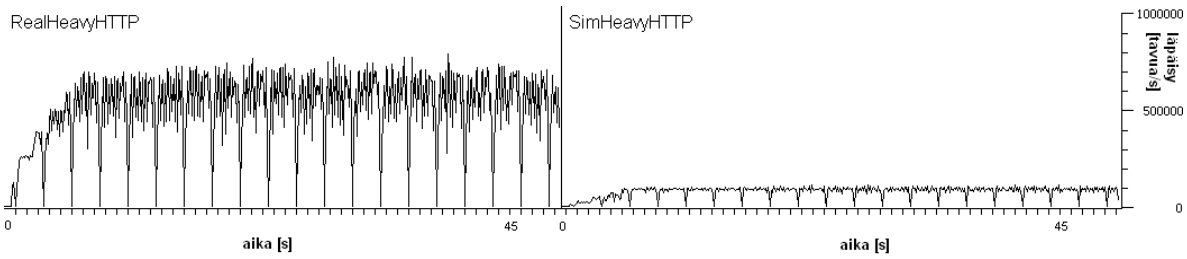
Kuva 5.3 Mittalaitteiden pakettikaappaukset: MediumFTP

Kuvassa 5.3 MediumFTP-mittausten pakettikaappaukset olivat molemmilla verkoilla miltei samanlaiset. Läpäisy oli molemmissa n. 125 kt/s = 1 Mbit/s. Läpäisy oli siis rajoitettu palvelimella 1 Mbit/s:iin, koska jo ennakkoon osattiin odottaa hybridiverkon maksimiläpäisyn jäävän oikeasta verkosta.



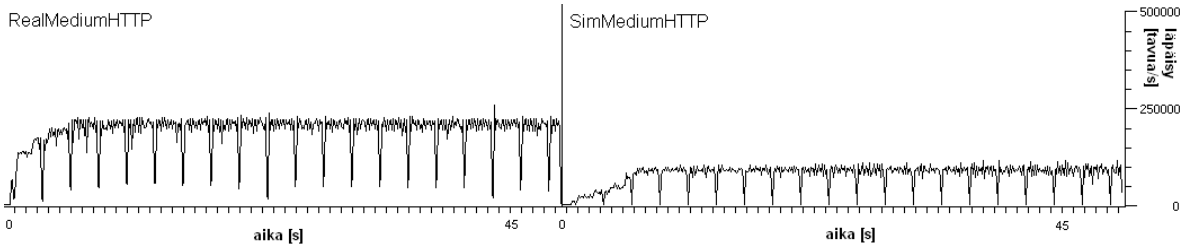
Kuva 5.4 Mittalaitteiden pakettikaappaukset: LowFTP

LowFTP-mittausten kaappaukset ovat samoin samankaltaiset. Läpäisyksi saatiin nyt n. 60 kt/s = 480 kbit/s. Kaappaukset on esitetty kuvassa 5.4.



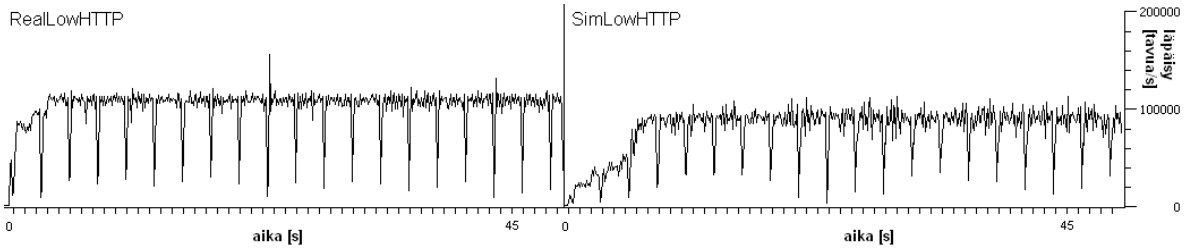
Kuva 5.5 Mittalaitteiden pakettikaappaukset: HeavyHTTP

HeavyHTTP-mittausten pakettikaappaukset kuvassa 5.5 eroavat taas samaan tapaan kuin HeavyFTP-mittauksissa. Hybridiverkko jää läpäisyssä n. 100 kt/s = 800 kbit/s, kun oikean verkon läpäisy on n. 600 kt/s = 4,8 Mbit/s.



Kuva 5.6 Mittalaitteiden pakettikaappaukset: MediumHTTP

MediumHTTP-mittausten pakettikaappaukset kuvassa 5.6 muuttuvat lähinnä vain oikean verkon osalta, jonka läpäisy on nyt n. 200 kt/s = 1,6 Mbit/s. Hybridiverkon läpäisy pysyy samana kuin HeavyHTTP-mittauksessa, eli n. 800 kbit/s.



Kuva 5.7 Mittalaitteiden pakettikaappaukset: LowHTTP

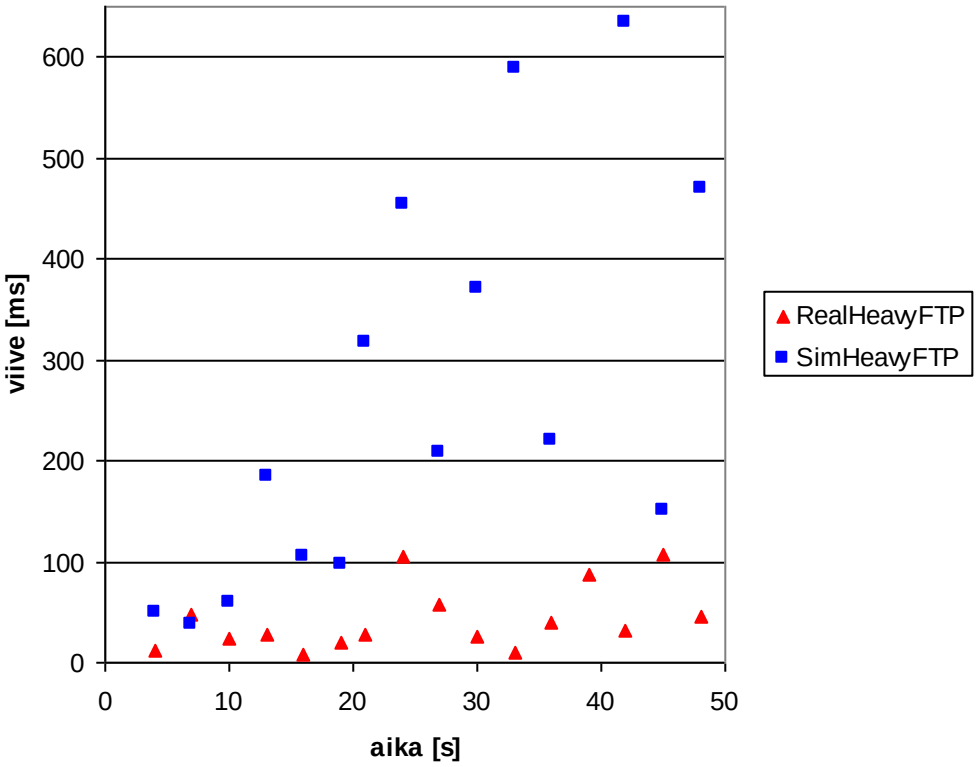
LowHTTP-mittausten pakettikaappaukset kuvassa 5.7 ovat suunnilleen samaa suuruusluokkaa. Oikealla verkolla läpäisy on vähän yli 100 kt = 800 kbit/s ja hybridillä vähän alle tämän.

Taulukossa 5.1 on esitetty mittalaitteiden antamat mittaustulokset. Tilaajien kokonaislukumäärä vaihteli 23:sta 41089:ään tilaajaan 70:n sekunnin aikana eri mittauksissa. Tilaajien lukumäärään laskettiin vain täysin onnistuneet transaktiot, eli onnistuneesti ladattu tiedosto FTP:lla tai sivu HTTP:lla. Vasteaika on keskimääräinen tilaajan pyynnöstä onnistuneeseen transaktioon kulunut aika. Lisäksi taulukossa on mittauksen aikana palvelimelta FTP:lla ladattu data. Taulukosta huomataan, että Heavy-mittauksissa on suuria eroja verkkojen kesken. LowFTP-mittausten heikko onnistumisprosentti johtunee FTP:n lyhyestä odotusajasta (*timeout*) palvelimen kuristaessa läpäisyä viiveen lisäämisen avulla.

	Tilaajien lkm.	Onnistumis-prosentti	Vasteaika (ms)	FTP-data (Mt)
RealHeavyFTP	396	100	1959	40
SimHeavyFTP	99	100	8405	10
SimEfficiency	114	100	7169	12
RealMediumFTP	68	100	13545	7
SimMediumFTP	68	100	13706	7
RealLowFTP	24	54,5	19603	4,5
SimLowFTP	23	53,4	19249	4,4
RealHeavyHTTP	41089	100	8	-
SimHeavyHTTP	6323	100	60	-
RealMediumHTTP	14928	100	26	-
SimMediumHTTP	6225	100	63	-
RealLowHTTP	7939	100	50	-
SimLowHTTP	6126	100	64	-
RealNetFail	61	95,3	13626	6
SimNetFail	68	100	13754	7

Taulukko 5.1 Mittalaitteiden avulla kerättyjä tuloksia

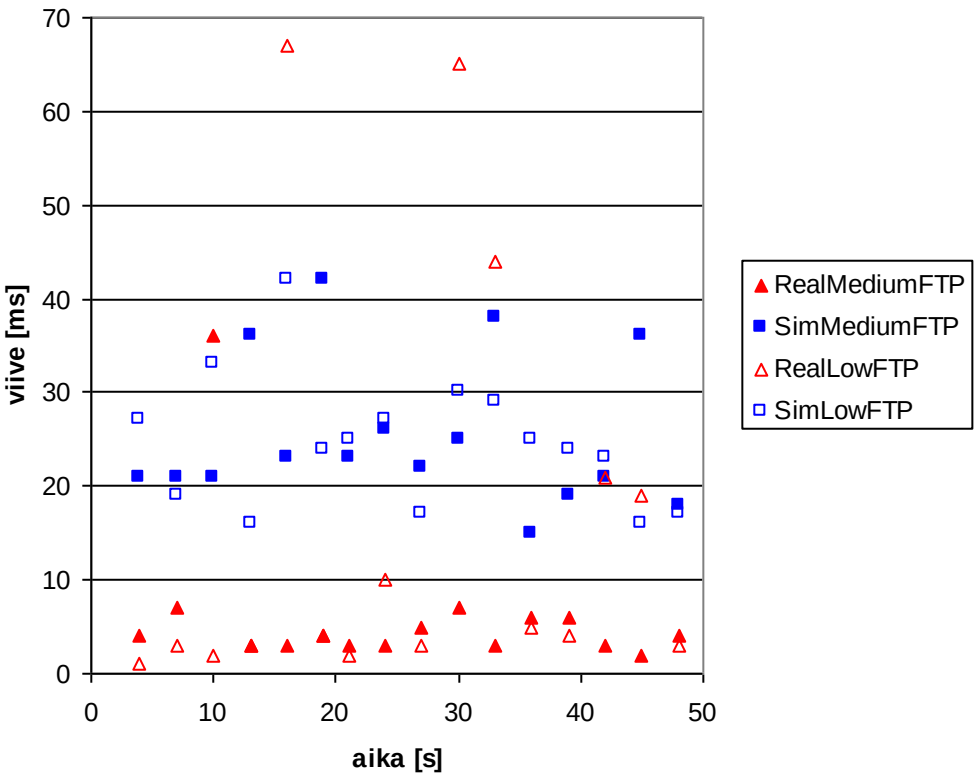
MediumFTP- ja LowFTP-mittaukset olivat pakettikaappausprofiileiltaan ja taulukossa 5.1 esitetyiltä tuloksiltaan samankaltaisia sekä hybridiverkossa että oikeassa verkossa. Kunkin mittauksen pakettikaappauksista laskettiin verkon Round-Trip Time (RTT) tilaajan lähettämälle paketille. RTT:stä vähennettiin mahdollinen palvelimen lisäämä viive, jota oli läipäisyltään rajoitetuissa mittauksissa Medium- ja Low-FTP/HTTP-mittauksissa. Viiveistä laskettiin jokaiselle mittaukselle 16 eri otosta tasaisin väliajoin 4-48 sekunnin kuluttua mittausliikenteen alkamisesta. Alkutransientti jätettiin tarkoituksella mittaustuloksiin, koska myös eroavaisuuksia transienteissa tilanteissa haluttiin tutkia.



Kuva 5.8 Viiveet HeavyFTP-mittauksissa

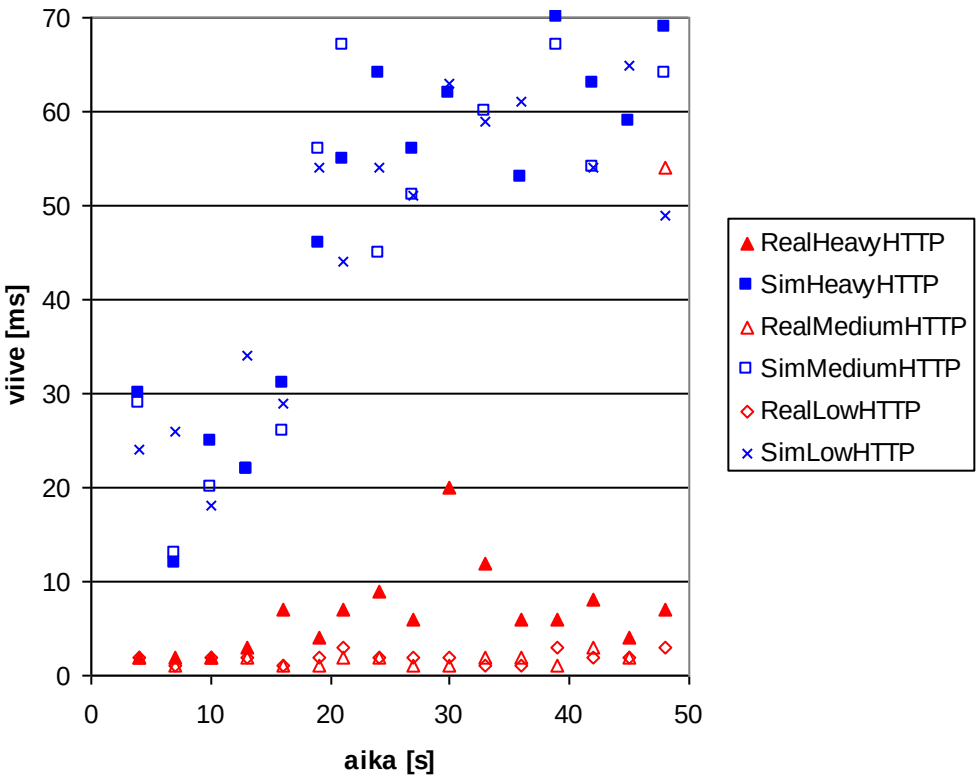
Kuvassa 5.8 on esitetty HeavyFTP-mittausten RTT eli viive, joka kului vastauksen saamiseen tilaajalta lähetettyyn pakettiin. Heavy-mittauksissa ei käytetty läpäisynrajoitusta, joten palvelin ei lisännyt ylimääräistä viivettä lainkaan. Heavy-mittaukset käyttävät siten verkon koko kapasiteetin hyväkseen ja antavat tietoa verkon maksimisuorituskyvystä. Kuvasta 5.8 havaitaan hybridiverkon viiveiden olevan moninkertaisia oikean vastaaviin.

Noin ajanjaksolla 0-20 sekuntia, oikea verkko ja hybridiverkko ovat vielä transientissa tilassa. Tämä johtuu siitä, että molemmat verkot ovat tyhjiä ennen mittausliikenteen generointia, poisluettuna OSPF:n generoima liikenne reititystaulujen ajan tasalla pitämiseen. Kuvista 5.2 ja 5.8 nähdään, että hybridiverkko ei saavuta stabiilia tilaa, vaan läpäisy- ja RTT-arvot vaihtelevat rajusti.



Kuva 5.9 Viiveet MediumFTP- ja LowFTP-mittauksissa

Kuvassa 5.9 on esitetty läpäisyltään rajoitetuiden mittausten RTT:t. Palvelin lisäsi mittauksissa viivettä tilaajien paketteihin vastaamiseen, jotta kulloistakin läpäisyrajoitetta ei ylitettäisi. Siten verkon koko kapasiteetti ei välttämättä ollut käytössä näissä mittauksissa. Palvelinten lisäämä keinotekoinen viive on poistettu kuvan 5.9 tuloksista, joten ne näyttävät pelkästään verkon aiheuttaman viiveen. Kuvan 5.3 mukaan MediumFTP-liikenteen pakettikaappausten profiileissa ja verkon läpäisyssä ei ollut merkittävää eroa. Kuvan 5.9 esittämät viiveet eroavat kuitenkin oikean verkon vastaavista. Hybridiverkon viiveet eivät koskaan alita 10 millisekuntia ja oikean verkon MediumFTP-mittauksen viiveen keskiarvo (6,2 ms) on selvästi alle 10 millisekunnin.

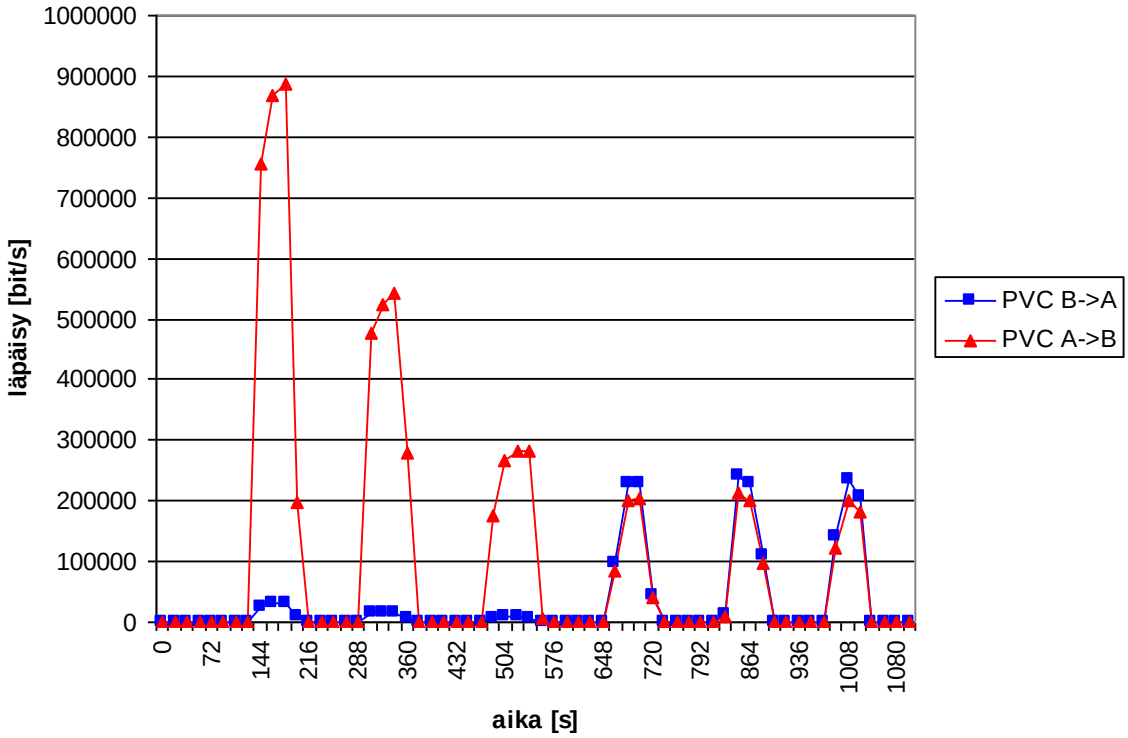


Kuva 5.10 Viiveet HTTP-mittauksissa

Kuvassa 5.10 on esitetty kaikkien HTTP-mittausten RTT:t. Läpäisyltään rajoitetuista Medium- ja Low-mittauksista on poistettu palvelimen viive samalla tavalla kuin FTP-mittauksista. Alkutransientti on HTTP-mittauksissakin ajanjaksolla 0-20 sekuntia. Oikeassa verkossa viiveen keskiarvo HeavyHTTP-mittauksessa on 6,6 millisekuntia. Hybridiverkolla viiveet ovat huomattavasti suurempia myös läpäisyltään rajoitetuissa mittauksissa.

Modelerin hybridisimulaatiossa otettiin talteen ATM:n PVC-reittien liikennemäärät, jotka on esitetty kuvissa 5.11 ja 5.12. Kuvassa 5.11 on reitti Ciscon reitittimien 7505_B:n ja 7505_A:n välillä ja kuvassa 5.12 vastaavasti 7507_C:n ja 7505_A:n välillä. Verkon topologia on esitettynä kuvassa 4.4. Huomataan, että molempien vaihtoehtoisten reittien käyttö on yhtäläistä ja kuormat jakautuneet tasaisesti hyödyntämään molempien reittien kapasiteettia. Modeler käyttää liikenteentasausta (*load balancing*) aina OSPF:n kanssa. Liikenteentasaus tulee kyseeseen silloin kun kohteeseen on olemassa useampia vaihtoehtoisia reittejä, joilla on sama hinta (*cost*). Moy ei määrittele tarkemmin liikenteentasauksen käyttöä OSPF:n kanssa, mutta toteaa useamman eri reitin samaan kohteeseen olevan mahdollisia reititystaulussa [RFC2328]. Tarkemman standardin

puutteesta johtuen eri reititinvalmistajien toteutukset OSPF:n liikenteen tasaamisesta saattavat olla erilaisia. [Cis5212, Cis18285, RFC2328]

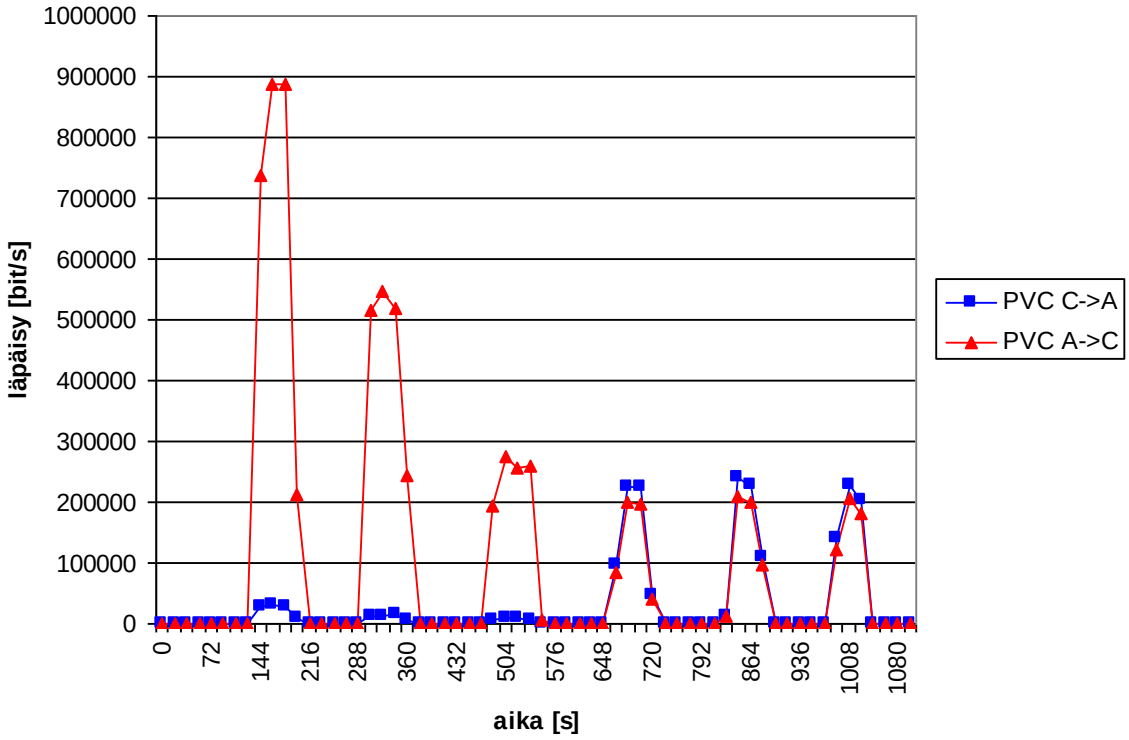


Kuva 5.11 Läpäisy molempiin suuntiin PVC-AB- ja PVC-BA-reittiä pitkin

Mittauksissa käytettiin pelkästään Ciscon valmistamia reitittimiä. Cisco on toteuttanut kaksi liikenteentasaustapaa OSPF:lle reitittimiinsä. Kohderiippuva liikenteentasaus (*per destination load balancing*) jakaa yhteydet kohteen mukaan eri linkeille. Samalle kohteelle menevät paketit samalta lähettäjältä siis reititetään aina samaa reittiä pitkin. Eri lähde-kohde-parit voidaan reitittää eri reittejä pitkin, jolloin kuorma jakautuu useammalle reitille. Pakettiriippuva liikenteentasaus (*per packet load balancing*) jakaa paketit eri reiteille pakettien saapumisen mukaan. Pakettiriippuvalla liikenteentasauksella saadaan kaikissa tilanteissa jaettua liikenne tasaisesti vaihtoehtoisille reiteille. Haittapuolena pakettiriippuvalla liikenteentasauksella on pakettien mahdollinen väärä saapumisjärjestys kohteeseen. Tästä on haittaa esimerkiksi Voice over IP (VoIP) -liikenteessä, jossa pakettien on saavuttava oikeassa järjestyksessä kohteeseen. [Cis5212, Cis18285]

Ciscon reitittimissä ja Modelerin malleissa Ciscon reitittimille liikenteen tasausta voidaan säädellä valitsemalla toinen mahdollisista liikenteentasaustavoista ja määrittämällä

enimmäismäärä vaihtoehtoisille reiteille. Liikenteentasaus voidaan kytkeä pois käytöstä määrittämällä vaihtoehtoisten reittien enimmäismääräksi yksi reitti. Modelerissa oletuksena on kohderiippuva liikenteentasaus ja kaikki vaihtoehtoiset reitit lisätään reititystauluun. [Cis5212, Cis18285]



Kuva 5.12 Läpäisy molempiin suuntiin PVC-AC- ja PVC-CA-reittiä pitkin

Simulaation talteen ottamista liikennemääristä käy hyvin ilmi eri vaiheiden kuormitus poluille ja liikenteen suunnat. FTP-liikenteessä suurin osa paketeista kulkee palvelimelta tilaajille, kun taas HTTP-liikenteessä kuormitus on suunnilleen yhtä suuri molempiin suuntiin. Tilanne on toki kärjistetty näissä mittauksissa lataamalla tyhjää HTML-sivua, mutta näin saatiin paremmin näiden kahden protokollan luonne-eron vaikutus vertailuun paremmin esille.

5.2 Muita mittauksia reaali- ja hybridiverkoilla

Tavallisten FTP- ja HTTP-mittausten lisäksi tehtiin muita hybridiverkon toimintaa kartoittavia mittauksia. Tässä kappaleessa käsitellään niiden tuloksia.

5.2.1 Tehokkuutta lisäävät yksinkertaistukset

Seuraavaksi esiteltävät yksinkertaistukset ovat Modelerin vipuja. Tarkoituksena oli selvittää niiden vaikutusta simulaatiotuloksiin. Yksinkertaistusten odotettiin vähentävän simulaatiotapahtumien määrää siten, että simulaation suoritus ei olisi liian raskas suorittimelle. Yksinkertaistukset liittyivät verkon reititysprotokolliin ja reititystaulujen muodostuksiin, joten niillä ei ole varsinaista vaikutusta hyötyliikenteeseen. Reititystaulut muodostetaan simulaatioajon alkuvaiheessa ja sen jälkeen oletetaan verkon tilan olevan staattinen. Mahdollisia vikatilanteita linkkien suhteen ei voida enää havaita. HeavyFTP-mittaus ajettiin hybridiverkolla käyttäen seuraavia yksinkertaistuksia:

“ARP Sim Efficiency”

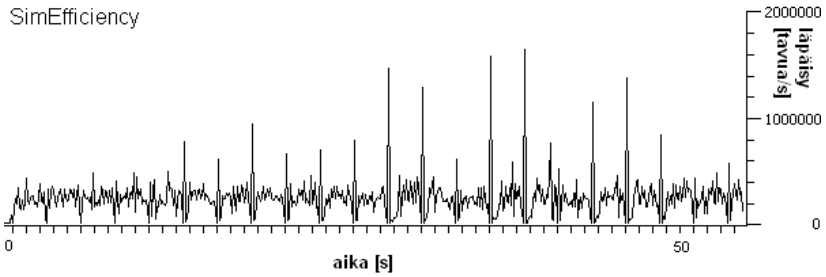
IP-osoitteiden kartoitus MAC-osoitteisiin tehdään ajanhetkellä 0 ja ARP-viestejä ei mallinneta simulaatiossa lainkaan. [Opn06]

“ATM Sim Efficiency”

ATM-solut siirretään suoraan vastaanottajalle oikeaan aikaan, eikä niitä mallinneta linkeille suoranaisesti. Siten ei saada selville linkkien statistiikkaa, mutta ATM-tuloksiin tai ylemmän tason tuloksiin tämä ei vaikuta. [Opn06]

“OSPF Sim Efficiency”

”OSPF Stop Time”-vivussa määritellään OSPF-viestien loppumisaika, jolloin reittitietoja ei enää välitetä reitittimien välillä. Tämän jälkeen OSPF-reittitiedot eivät enää muutu. Siten ei enää voida simuloida linkkien tai solmujen vikaantumista. Mittauksissa loppumisaika säädettiin siten, että liikenne generoitiin verkkoon vasta loppumisajan jälkeen. [Opn06]

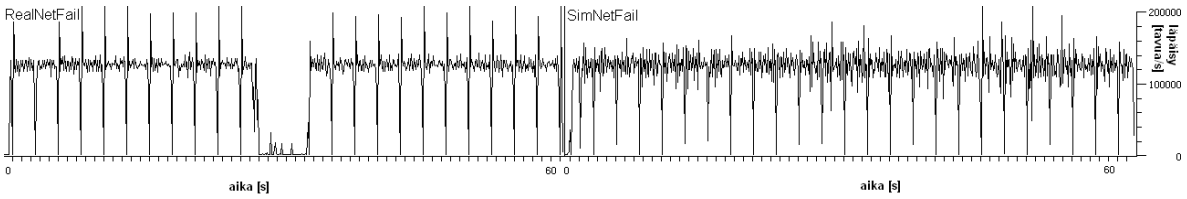


Kuva 5.13 Mittalaitteiden pakettikaappaukset: SimEfficiency

Yksinkertaistuksien tuomaa tehokkuushyötyä mitattiin HeavyFTP-mittauksella. Kuvassa 5.13 on esitetty mittauksen pakettikaappaus. Pakettikaappaus ja statistiikka eivät eroa huomattavasti ilman yksinkertaistuksia tehdystä simuloinnista. HeavyFTP-mittauksen pakettikaappaus ilman yksinkertaistuksia esitettiin kuvassa 5.2. Saadut arvot ovat hieman lähempänä oikeaa verkkoa, mutta tulokset jäävät silti kauas oikealla verkolla tehdystä vastaavasta mittauksesta.

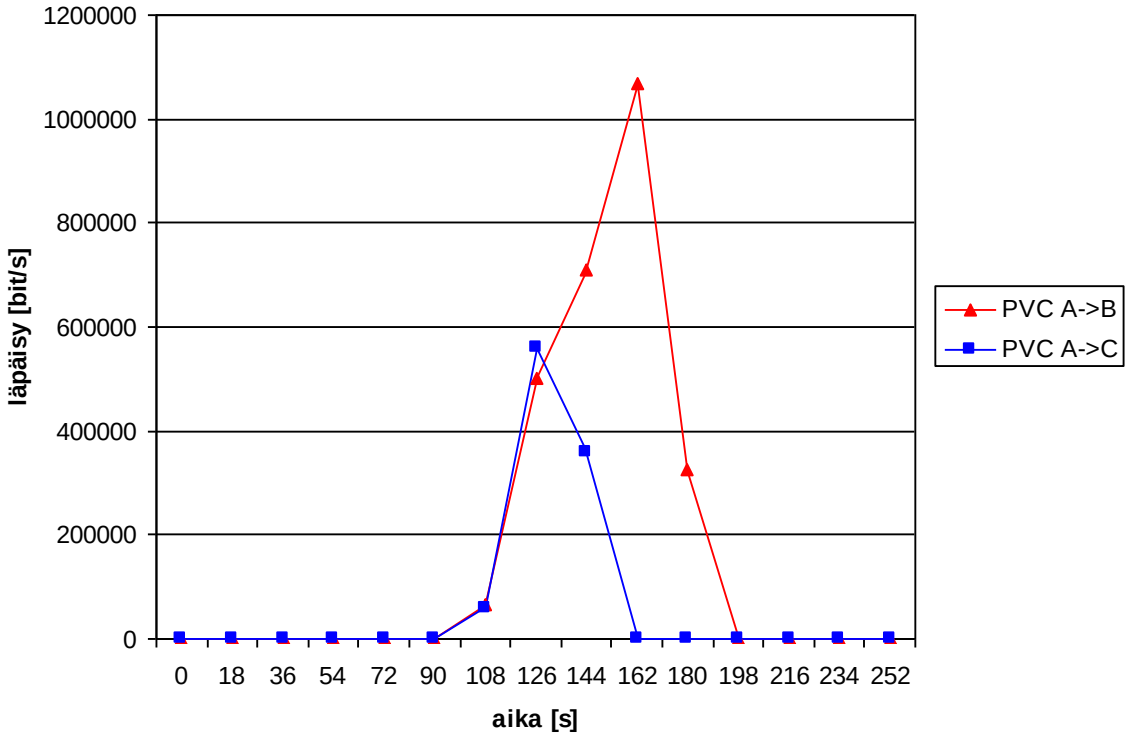
5.2.2 Vikatilanne reaali- ja hybridiverkossa

Vikatilanne luotiin oikeaan verkkoon kytkemällä Ethernet-kanava pois päältä Ciscon 7507_C-ATM-reitittimeltä. Hybridiverkossa vikatilanne generoitiin samaan tapaan määrittelemällä 7507_C-ATM- ja 2621_C-Ethernet-reitittimien välinen linkki pettämään. Verkon topologia on esitetty kuvassa 4.4. Molemmat vikatilanteet ajoitettiin suunnilleen liikenteen generoimisen puoliväliin, eli n. 30 sekunnin päähän liikenteen alkamisesta. Linkki on poissa käytöstä molemmissa tapauksissa mittauksen loppuun asti. Mittauksissa käytettiin MediumFTP-liikennettä.



Kuva 5.14 Mittalaitteiden pakettikaappaukset: NetFail

Kuvassa 5.14 esitetyistä pakettikaappauksista huomataan, ettei vikatilanne vaikuttanut hybridisimulaation liikenteeseen siten, että se olisi keskeytynyt missään vaiheessa. Oikean verkon tilanteessa liikenne keskeytyy hetkeksi, kunnes OSPF-protokolla reagoi tilanteeseen ja reitittää kaiken liikenteen toimivan 7505_B-reitittimen kautta.

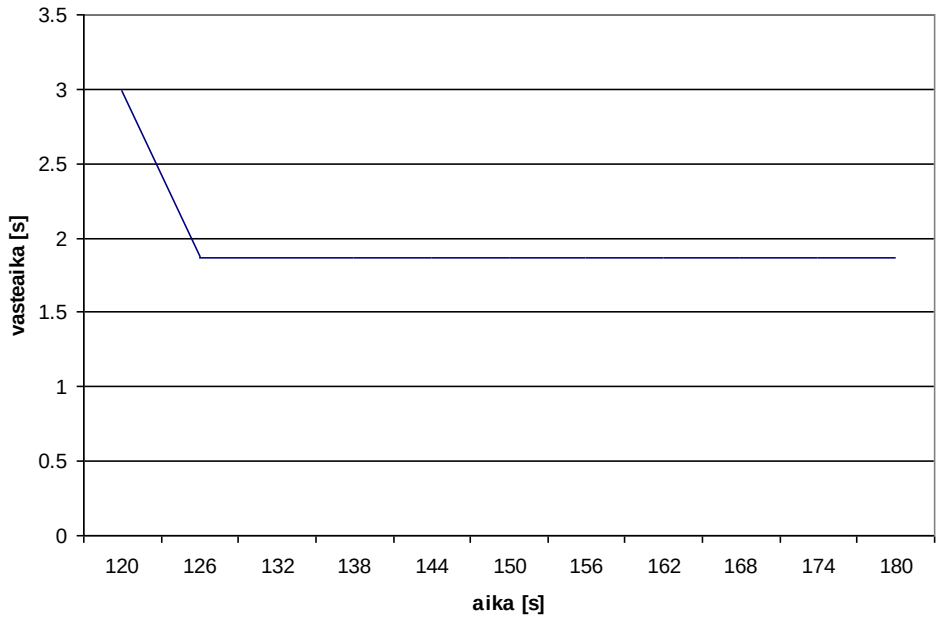


Kuva 5.15 PVC-reittien läpäisyt tilaajien suuntaan

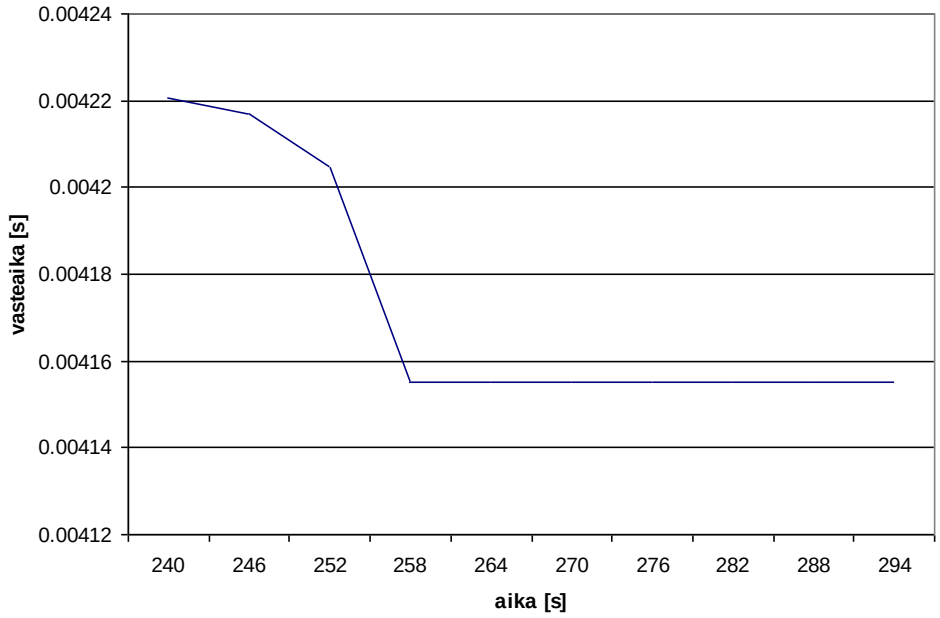
Kuvassa 5.15 esitetyssä Modelerin hybridisimulaation liikennemääristä eri reiteillä voidaan tarkastella, mitä simulaatiossa tapahtui vikatilanteessa. Nähdään, että liikenne jakautuu aluksi tasaisesti molemmille reiteille, kunnes toisen reitin pettäessä liikenne siirtyy pelkästään ehjää reittiä pitkin. Pakettikaappauksista huomattiin, ettei tämä näkynyt tilaajien päässä millään tavalla, eli liikenne ei keskeytynyt hetkeksikään.

5.3 Täysin simuloidun verkon tulokset

Seuraavaksi esitellään täysin simuloidun verkon tulokset. Kuvassa 5.16 vasteaika kuvan 4.5 FTP-pyynnöille. Vasteaika kuvaa aikaa, joka kului tiedoston lataamiseen. Stabiilin vaiheen vasteaika on n. 1864 millisekuntia. Kuvassa 5.17 on esitetty vasteaika, joka kuvaa sivun lataamiseen kulunutta aikaa kuvan 4.6 HTTP-pyynnöille. Stabiilin vaiheen vasteaika on n. 4,155 millisekuntia. Saatuja tuloksia verrataan reaali- ja hybridiverkon vastaaviin arvoihin HeavyFTP- ja HeavyHTTP-mittauksista luvussa 6.



Kuva 5.16 FTP-pyyntöjen vasteaika

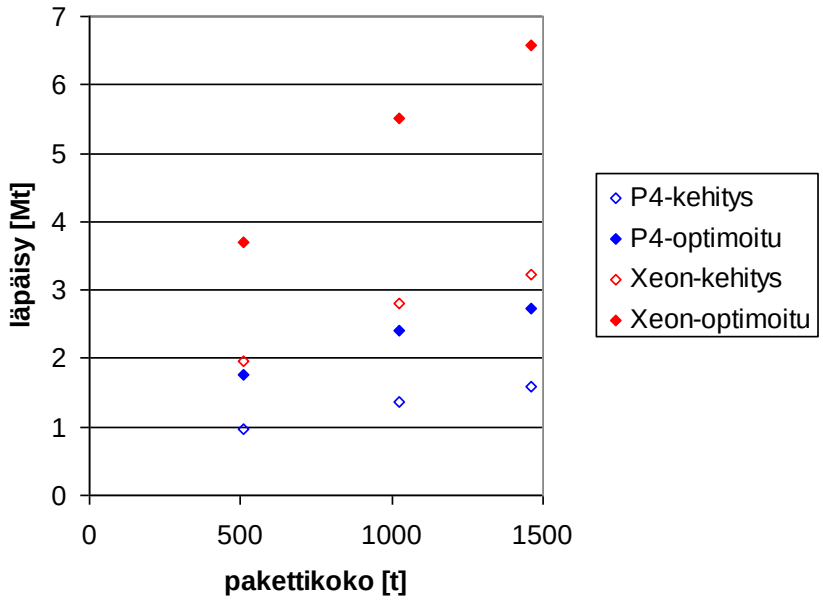


Kuva 5.17 HTTP-pyyntöjen vasteaika

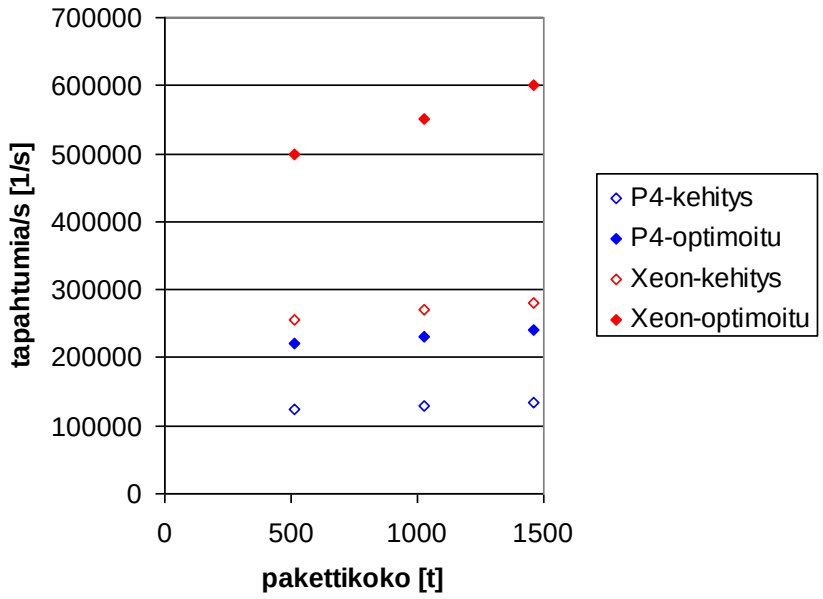
5.4 Tulokset laskentatehon vaikutuksesta hybridiverkkoon

Laskentatehon vaikutusta tarkastelevissa mittauksissa kerättiin verkon läpäisyn keskiarvo kullakin mittausjärjestelyllä. Kuvassa 5.18 on esitetty pakettikoon ja simulaatiokernelin vaikutus läpäisyyn. Kuva 5.19 esittää simulaatiotapahtumien suoritusnopeuden eri pakettikokoja ja kerneleitä käytettäessä. Tulokset on molemmissa kuvissa eritelty Pentium

4-työaseman ja Xeon-tehotyöaseman osalta. Suoritusnopeus ja läpäisyarvo ovat selvästi riippuvaisia toisistaan. Tehotyöaseman tehokkaampi suoritin pystyi huomattavasti parempaan tapahtumien suoritusnopeuteen. Suurempi pakettikoko mahdollistaa suuremman läpäisyn, koska samaan pakettimäärään mahtuu enemmän hyötykuormaa. Hyötykuorman lisäys ei lisäkuormita merkittävästi simulaatiota, kuten pakettien lukumäärän mahdollinen lisääminen. Suoritusnopeuden lisääntyminen pakettikoon kasvaessa on selitettävissä vain tapahtumien muuttumisena nopeammiksi suorittaa.



Kuva 5.18 Pakettikoon vaikutus läpäisyyn



Kuva 5.19 Tapahtumien suoritusnopeus simulaatioajossa

6. Johtopäätökset

Hybridisimulaatiolle aiheutuneiden tapahtumien määrästä voidaan päätellä, että HeavyFTP- ja kolme HTTP-mittausta olivat raskaimmat suoritettavaksi simulaatiotyöasemalle. Mittalaitteilla kerätyt tulokset vahvistavat osaltaan tätä päätelmää. HeavyFTP-mittauksella työasema kykeni käsittelemään n. 150000 tapahtumaa sekunnissa. Kaikilla HTTP-mittauksilla päästiin vain n. 130000 tapahtumaan sekunnissa. Kuitenkin näiden neljän mittauksen osalta voidaan todeta, että työasema oli suorituskykynsä äärirajoilla tai tehtävien määrä ylitti sen suorituskyvyn, kuten muista jäljempänä käsiteltävistä tuloksista voidaan todeta. Näin ollen HeavyFTP-mittauksen aiheuttamat tapahtumat olivat nopeampia suorittaa kuin HeavyHTTP-mittauksen, koska työasema ehti suorittaa enemmän tapahtumia sekunnissa HeavyFTP-mittauksen aikana kuin HeavyHTTP-mittauksen. Simulointityöasemalle ei voida siis määrittää yksiselitteistä tapahtumien suorituskykymäärää sekunnissa, koska tapahtumien suoritusaika saattaa vaihdella erilaisissa simulaatioissa.

HeavyFTP-mittauksen tulokset jäivät hybridiverkolla oikean verkon tuloksista jälkeen kaikilla tarkastelluilla arvoilla. Läpäisy hybridiverkolla jäi n. 2 Mbit/s oikean verkon siirtäessä samassa mittauksessa n. 8 Mbit/s. Hybridiverkko pystyi palvelemaan 99 tilaajaa mittauksen aikana, kun oikea suoriutui nelikertaisesta määrästä. Oikean verkon RTT-arvo jäi n. 100 millisekuntiin tai alle, kun taas hybridiverkon RTT vaihteli 100-650 millisekunnin välillä alkutransientin jälkeen. HeavyFTP-mittaus oli siis liian raskas simuloitavaksi reaaliajassa kyseisellä verkon topologialla ja käytössä olleella simulointityöasemalla.

MediumFTP-mittauksen pakettikaappausten profiileissa oli havaittavissa pientä eroa, mutta läpäisyarvot ja palveltujen tilaajien lukumäärä olivat samaa suuruusluokkaa. RTT-arvot olivat kuitenkin selvästi suuremmat hybridiverkolla. LowFTP-mittauksen pakettikaappausten profiilit olivat kaikkein lähimpänä toisiaan, kuten myös mitatut arvot olivat molemmilla verkoilla vastaavat. RTT-arvot olivat kuitenkin edelleen n. nelinkertaisia hybridiverkolla. Voidaan kuitenkin todeta, että LowFTP-mittaus oli kevyin kaikista mittauksista simuloida ja siitä saadut tulokset olivat kaikkein lähimpänä oikean verkon vastaavia.

HTTP-mittaukset olivat selvästi raskaampia simuloida kuin FTP-mittaukset. HeavyHTTP-mittausten tuloksissa hybridiverkko jäi kaikkein kauimmaksi oikean verkon tuloksista. Hybridiverkon tulokset olivat samaa suuruusluokkaa kaikilla HTTP-mittauksilla; läpäisy- ja RTT-arvot eivät suuresti eronneet toisistaan eri HTTP-mittauksissa. Voidaan sanoa, että kaikki HTTP-mittaukset olivat liian raskaita simuloitaviksi reaaliajassa työasemalla. LowHTTP-mittauksen tulokset olivat lähimpänä oikeaa, mutta senkin palvelemien tilaajien kokonaismäärä jäi n. 2000 kappaletta oikeasta. RTT-arvot nousivat hybridiverkolla kaikissa HTTP-mittauksissa paikoitellen yli 60 millisekunnin, kun oikealla verkolla melkein kaikki kerätyt RTT-arvot olivat alle 10 millisekuntia.

Liikenteentasauksen osalta havaittiin merkittävä ero Modelerin Cisco-mallien ja oikeiden Cisco-reitittimien oletusasetuksissa. Molemmissa on oletuksena käytössä kohderiippuva liikenteentasaus, mutta malleissa otetaan käyttöön kaikki mahdolliset reitit yhteen kohteeseen, kun taas oikeassa reitittimessä oletuksena reititystauluun määritellään vain yksi reitti yhdelle kohteelle, vaikka tarjolla olisi useampia. Näin ollen oikeassa Cisco-reitittimessä liikenteentasaus on poissa käytöstä oletuksena.

Simulaation yksinkertaistukset eivät vähentäneet juurikaan simulaatiotyöaseman kuormitusta ja HeavyFTP-mittauksella hybridiverkko ehti palvella vain 15 tilaajaa enemmän kuin ilman yksinkertaistuksia. Oikeasta verkosta yksinkertaistettu simulaatio jäi silti n. 300 tilaajan verran. Yksinkertaistukset eivät siis parantaneet varsinaisesti tuloksia hybridiverkon osalta, joten niiden tarpeellisuus voidaan kyseenalaistaa ainakin tämän tyyppisissä simulaatioissa.

Medium-FTP mittauksen aikana generoitu vikatilanne vaikutti eri tavalla oikeaan ja hybridiverkkoon. Oikeassa verkossa liikenne katkesi hetkeksi, kunnes paketit reititettiin vaihtoehtoista reittiä pitkin. Oikeassa verkossa oli siis aina käytössä vain toinen reiteistä, koska liikenteentasaus ei ollut kytkettynä. Hybridiverkossa oli kaikissa mittauksissa käytössä kohderiippuva liikenteentasaus, eli eri lähde-kohde-parit voitiin reitittää tasaisesti hyödyntämään molempia reittejä. Tästä ei tosin ollut verkon läpäisyn kannalta hyötyä, koska sitä kuitenkin rajoittivat vain yhdet linkit tilaajille ja palvelimelle. Virhetilanteessa liikenteentasauksesta oli kuitenkin hyötyä sen verran, että liikenne ei katkennut hybridiverkolla hetkeksikään.

	Vasteaika (ms)
RealFTP	1959
HybridFTP	8405
FullSimFTP	1864
RealHTTP	8
HybridHTTP	60
FullSimHTTP	4

Taulukko 6.1 Verkkojen vasteajat

Taulukkoon 6.1 on koottu reaali- ja hybridiverkon sekä täysin simuloidun verkon tulokset tiedoston ja sivun lataamisen vasteaikojen osalta. Täysin simuloidun verkon tulokset olivat paljon lähempänä oikean verkon tuloksia, kuin hybridiverkon tulokset. Tiedoston lataaminen FTP-protokollalla kesti stabiilissa vaiheessa n. 1864 millisekuntia ja samankokoisen tiedoston lataamiseen oikealla verkolla kului keskimäärin 1959 millisekuntia. Sivun lataamisen kestolle HTTP-protokollalla simulaatio antoi liian optimistisen arvon n. 4 millisekuntia, kun oikealla verkolla kului keskimäärin 8 millisekuntia. Arvojen eroavaisuus saattaa johtua simulaation liikenteen määrittelemisen eroavaisuudesta mittalaitteisiin nähden, jolloin molempiin on vaikea saada määriteltyä toisiaan vastaavia liikenneprofiileja. Tätä eroavaisuutta käsiteltiin luvussa 4.5. Näiden tulosten perusteella voidaan kuitenkin todeta, että hybridiverkolla saatujen tulosten jääminen jälkeen oikean verkon tuloksista johtui nimenomaan reaaliaikasimulaation aiheuttamista ongelmista, eikä varsinaisesti mahdollisista epärealistisesti mallinnetuista laitteista.

Simulointityöaseman laskentatehon vaikutus tuloksiin oli merkittävä sitä tarkastelevissa mittauksissa. Tämä johtui siitä, ettei kumpikaan tutkittavista työasemista ja kerneleistä pystynyt täyttämään hybridisimulaation reaaliaikavaatimusta, vaan simulaatiokello jäi jälkeen jopa tehokkaimmalla yhdistelmällä. Läpäisyarvot optimoidulla kernelillä ja tehotyöasemalla eivät kuitenkaan jääne kauas oikean verkon vastaavista. Tuloksia voitaneen saada vielä realistisempaan suuntaan käyttämällä vieläkin tehokkaampaa simulointityöasemaa, ottamalla työaseman kaikki ytimet käyttöön simulaatiossa tai laatimalla simuloitava verkko ja sen liikenne muulla tavalla kuin käyttäen raskasta ja koko verkon kapasiteetin haltuun ottavaa TCP-liikennettä. Läpäisyarvoja tarkasteltaessa tulee aina ottaa huomioon generoitu liikenne ja sen ominaisuudet, kuten mittaukset eri kokoisilla paketeilla osoittavat. Tästä syystä laskentatehomittaukset eivät ole täysin vertailukelpoisia aiempien reaali- ja hybridiverkon vastaaviin, koska ne tehtiin eri liikennegeneraattoreilla.

Yhteenveto

Luotettava hybridisimulaatio tarjoaa tehokkaan työkalun tietoverkon toiminnan ennakoimiseen ilman tarvetta rakentaa reaaliverkkoa kokonaan. Simulaation luotettavuus on määriteltävä tapauskohtaisesti, jolloin on otettava huomioon tarkasteltavan verkon koko ja siinä kulkeva liikenne sekä käytössä olevan simulaatiotyöaseman resurssit tehtävän suorittamiseen. Hybridisimulaation tuomat edut ovat kuitenkin ilmeisiä ja merkittäviä, jos simulaatioihin liittyvät ongelmat saadaan ratkaistua tai kierrettyä tässä työssä esitellyillä tavoilla.

Työn tavoitteena oli saada tietoa hybridiverkon simuloinnista ja sen tuottamien tulosten luotettavuudesta. Tietoa onnistuttiin saamaan mittausten ja vertailujen avulla. Työ nosti esiin hybridisimulaatioihin liittyviä ongelmakohtia ja haasteita. Tulokset täysin simuloidulla verkolla osoittavat käytettyjen mallien olevan realistisia. Ongelmat hybridiverkossa johtuvat siis nimenomaan reaaliaikavaatimuksesta. Suurilla liikennemäärillä reaaliajassa pysyminen tuottaa vaikeuksia tehokkaimmallekin simulointilaitteistolle. Simulointityöaseman suorituskykyä nostamalla voi kuitenkin simuloida reaaliaikaisesti suurempia verkkoja suuremmilla liikennemäärillä. Hybridisimuloinnin käyttökohteet tulisi silti olla liikennemäärältään pienehköjä tai suorituskykyongelma tulisi ratkaista muuten kuin pelkästään työaseman suorituskykyä nostamalla. Työasemien laskentateho kasvaa jatkuvasti, mutta myös tietoverkot monimutkaistuvat ja niiden liikenne kasvaa. Yksi ratkaisu suorituskykyongelmaan voisi olla simulaation hajauttaminen, johon on olemassa muutamia vaihtoehtoisia ratkaisuja, joita olisi syytä tutkia enemmän tulevaisuudessa.

Lähdeluettelo

[Ban01] Jerry Banks, John S. Carson, Barry L. Nelson ja David M. Nicol, Discrete-event system simulation; Prentice-Hall, 2001.

[Bro02] B. Van den Broeck, P. Leys, J. Potemans, J. Theunis, E. Van Lil ja A. Van den Capelle, Validation of router models in OPNET; Katholieke Universiteit Leuven, 2002.

[Cis00] Cisco Systems, Guide to ATM Technology; Cisco Systems Inc., 2000.

[Cis5212] Cisco Systems, DocumentID: 5212 How does load balancing work?; www-sivu: http://ww.cisco.com/warp/public/105/46.html , Cisco Systems Inc, 2005, viitattu 2.3.2007.

[Cis18285] Cisco Systems, DocumentID: 18285 Troubleshooting load balancing; www-sivu: http://ww.cisco.com/warp/public/105/loadbal_cef.html , Cisco Systems Inc, 2005, viitattu 2.3.2007.

[Fal99] Kevin Fall, Network emulation in the vint/ns simulator; UC Berkeley, 1999.

[Fal07] Kevin Fall ja Kannan Varadhan, The ns Manual; www-sivu: http://www.isi.edu/nsnam/ns/ns-documentation.html , UC Berkeley, LBL, USC/ISI, Xerox PARC, 2007, viitattu 14.2.2007.

[Gar90] Ricardo F. Garzia ja Mario R. Garzia, Network modeling, simulation, and analysis; Marcel Dekker, Inc., 1990.

[Gio05] A. Giovanardi ja G. Mazzini, Ad Hoc Routing Protocols: Emulation vs Simulation; University of Ferrara, 2005.

[Gru97] Mika Grundström, ATM-tekniikka ja monipalveluverkot; Suomen ATK-kustannus, 1997.

[Gur04] S. B. Guruprasad, Issues in integrated network experimentation using simulation and emulation; University of Utah, 2004.

[ISO7498] International Organization for Standardization, International standard 7498-1; ISO/IEC, 1996.

[Kid05] C. Kiddle, R. Simmonds ja B.Unger, Improving scalability of network emulation through parallelism and abstraction; University of Calgary, 2005.

[Las06] Pasi Lassila, S-38.3148 Simulation of data networks – lecture notes; www-sivu: http://www.netlab.tkk.fi/opetus/s383148/s06/material.shtml , Teknillinen Korkeakoulu, 2006, viitattu 1.3.2007.

[Law00] Averill M. Law ja W. David Kelton, Simulation modeling and analysis; McGraw-Hill Higher Education, 2000.

[Mac03] J. P. Macker, W. Chao ja J. W. Weston, A low-cost, IP-based, mobile network emulator (mne); Naval Research Laboratory, 2003.

[Mah04] D. Mahrenholz ja S. Ivanov, Real-time network emulation with ns-2; University of Magdeburg, 2004.

[Nic05] D. M. Nicol, M. Liljenstam ja J. Liu, Advanced concepts in large-scale network simulation; University of Illinois, Colorado School of Mines, 2005.

[Opn06] OPNET Technologies, Modeler Documentation Set V. 12.0; OPNET Technologies Inc., 2006.

[RFC793] Information sciences institute, RFC 793 – Transmission control protocol; University of southern California, 1981.

[RFC959] J. Postel ja J. Reynolds, RFC 959 – File transfer protocol; Internet Engineering Task Force, 1985.

[RFC1577] M. Laubach, RFC 1577 – Classical IP and ARP over ATM; Network Working Group, 1994.

[RFC2328] J. Moy, RFC 2328 – OSPF Version 2; Internet Engineering Task Force, 1998.

[RFC2453] G. Malkin, RFC 2453 – RIP Version 2; Network Working Group, 1998.

[RFC2616] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee, RFC 2616 – HTTP/1.1; Internet Engineering Task Force, 1999.

[RFC2684] D. Grossman, J. Heinanen, RFC 2684 – Multiprotocol Encapsulation over ATM Adaptation Layer 5; Motorola, Telia, 1999.

[Spi06] Spirent Communications, Avalanche Commander online help; Spirent Communications Inc., 2006.

[Tho06] M. Thoppian, H.T. Vu, A. Mehdian, S. Venkatesan, R. Prakash ja A.J. Anderson, Real-time simulations of mobile ad-hoc networks in Opnet Modeler; University of Dallas, Rockwell Collins Inc., 2006.

[Tir03] A. Tirumala, F. Qin, J. Dugon, J. Ferguson ja K. Gibbs, Iperf version 1.7.0; www-sivu: http://dast.nlanr.net/Projects/Iperf/ , University of Illinois, 2003.

[Wel03] R. J. Wellington ja M. D. Kubischta, Wireless network emulation for distributed processing systems; General Dynamics Advanced Information Systems, 2003.

[Zhe03] P. Zheng ja L. M. Ni, Test and evaluation of wide area networks using emulator cluster; Michigan State University, Hong Kong University of Science and Technology, 2003.

[Zhe04] P. Zheng ja L. M. Ni, EMPOWER: a cluster architecture supporting network emulation; Michigan State University, Hong Kong University of Science and Technology, 2004.