ANALISIS PERBANDINGAN PERFORMANSI METODE *FILE SHARING* BERBASIS *MULTICAST* DENGAN *PEER-TO-PEER* DALAM PROSES DISTRIBUSI KONTEN DATA

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Dimas Dwi Atmojo
NIM: 115060801111008

PROGRAM STUDI INFORMATIKA/ILMU KOMPUTER
PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2016

PENGESAHAN

ANALISIS PERBANDINGAN PERFORMANSI METODE *FILE SHARING* BERBASIS *MULTICAST* DENGAN *PEER-TO-PEER* DALAM PROSES DISTRIBUSI KONTEN DATA

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Dimas Dwi Atmojo
NIM: 115060801111008

Skripsi ini telah diuji dan dinyatakan lulus pada
18 Januari 2016
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Achmad Basuki, ST., M.MG., Ph.D	Kasyful Amron, ST., M.Sc
NIP: 19741118 200312 1 002	NIP: 19750803 200312 1 003

Mengetahui
Ketua Program Studi Informatika/Ilmu Komputer

Drs. Marji, MT
NIP: 19670801 199203 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 18 Januari 2016

Dimas Dwi Atmojo

NIM: 115060801111008

KATA PENGANTAR

Puji syukur penulis ucapkan kehadirat Allah SWT karena atas rahmat, karunia, taufik dan hidayah-Nya, penulis dapat menyelesaikan tugas akhir atau skripsi yang berjudul **"*Analisis Perbandingan Performansi Metode File Sharing Berbasis Multicast dengan Peer-to-Peer dalam Proses Distribusi Konten Data*"** untuk memenuhi gelar sarjana komputer. Penulis menyadari bahwa penyusunan laporan skripsi ini tidak akan berjalan dengan lancar tanpa bantuan dan masukan yang bermanfaat melalui dosen pembimbing serta pihak-pihak disekitar penulis. Oleh karena itu, penulis mengucapkan terima kasih kepada pihak-pihak tersebut yang telah bersedia untuk memberikan arahan, saran, dan dukungan demi kelancaran penyusunan laporan skripisi ini diantaranya:

1. Bapak Achmad Basuki, ST., M.MG., Ph.D selaku dosen pembimbing I dan Bapak Kasyful Amron, ST., M.Sc. selaku dosen pembimbing II yang telah memberikan bimbingan serta menyampaikan ilmu, saran, dan motivasi dalam penyelesaian skripsi ini.
2. Kedua orang tua penulis, Bapak Edy Prijomono dan Ibu Sumini, S.Pd, yang selalu memberikan motivasi, kasih sayang, do'a, serta dukungan baik moril maupun materiil.
3. Keluarga kakak kandung Bagus Prasetiaji, S.Pd, Yayuk Mustika Yanti, S.Pd, dan Akasha Fadhil Prasetiaji, yang selalu memberikan dukungan dan doa.
4. Teman-teman seperjuangan skripsi lintas laboratorium PTIIK khususnya angkatan 2011, yang telah memberikan semangat dan dukungan dalam pengerjaan skripsi ini.
5. Teman-teman kelas F Program Studi Informatika angkatan 2011 (SCANF), yang telah memberikan bantuan dan dukungannya selama masa perkuliahan.
6. Segenap dosen dan karyawan PTIIK Universitas Brawijaya yang telah membantu pelaksanaan skripsi ini.
7. Semua pihak yang tidak dapat penulis sebutkan satu per satu yang terlibat baik secara langsung maupun tidak langsung demi terselesaikannya skripsi ini.

Semoga jasa dan amal baik dari seluruh pihak yang bersangkutan mendapatkan balasan dan barakah dari Allah SWT. Kritik dan saran yang membangun sangat penulis harapkan dari pembaca mengingat laporan yang penulis susun ini masih jauh dari kata sempurna. Akhir kata, penulis ucapkan terima kasih dan semoga laporan skripsi ini dapat memberikan manfaat bagi pembaca terutama mahasiswa PTIIK Universitas Brawijaya.

Malang, 18 Januari 2016

Penulis
atmojo.dimas@gmail.com

ABSTRAK

Salah satu pemanfaatan fungsi dari jaringan komputer yaitu sebagai media distribusi konten data atau *file sharing*. *File sharing* banyak dimanfaatkan untuk menunjang mekanisme distribusi konten data secara terpadu dengan memanfaatkan jaringan internet ketika menghadapi kendala waktu, biaya dan lokasi dari pengguna yang tersebar secara geografis. Sebagian besar pengguna ketika memanfaatkan media distribusi konten data melihat kecepatan dan waktu transfer sebagai parameter efektifitas dari metode yang digunakan. Sedangkan disisi lain parameter tersebut dipengaruhi oleh banyak faktor, seperti ukuran konten data yang didistribusikan, kondisi lingkungan jaringan komputer yang dilalui dan kinerja metode *file sharing* yang digunakan.

Saat ini metode yang banyak dimanfaatkan dalam mekanisme distribusi konten data yaitu melalui pendekatan *multicast* dan *Peer-to-Peer* (Costa-Montenegro, 2012). Setiap metode tersebut menawarkan solusi dalam menunjang mekanisme distribusi konten data. Sehingga dibutuhkan analisis yang mendalam untuk mengetahui metode mana yang lebih efektif dan efisien untuk melakukan proses distribusi konten data. Untuk dapat mengetahui performansi dari metode *multicast* dan *Peer-to-Peer*, diperlukan contoh aplikasi *file sharing* yang menerapkan masing-masing metode tersebut.

Dalam penelitian ini contoh aplikasi *file sharing* berbasis *multicast* yang digunakan yaitu UFTP. Sedangkan contoh aplikasi berbasis *Peer-to-Peer* yang digunakan yaitu BitTorrent Sync. Kemudian untuk mendukung pengujian dan analisis perbandingan performansi dari setiap metode *file sharing* dan contoh aplikasi terkait pengaruh kondisi lingkungan jaringan komputer lokal yang dilalui dalam proses distribusi konten data, diperlukan bantuan emulator jaringan seperti *Wide Area Network emulator* (WANem). Untuk dapat meminimalkan faktor lain yang dapat menggangu akurasi pengukuran perbandingan performansi dari masing-masing aplikasi *file sharing* dalam proses distribusi konten data seperti *network policy* yang terdapat pada suatu jaringan *physical*, maka lingkungan pengujian sistem diimplementasikan secara virtual menggunakan bantuan aplikasi Virtualbox. Kriteria performansi metode *file sharing* dikatakan baik ketika mampu menghasilkan nilai *throughput* atau *transfer rate* yang tinggi dengan nilai waktu proses yang rendah ketika digunakan untuk mendistribusikan berbagai macam ukuran konten data dalam kondisi lingkungan jaringan komputer tertentu.

Kata kunci: *File Sharing*, *Multicast*, *Peer-to-Peer*, UFTP, BitTorrent Sync.

ABSTRACT

One of the used of computer network function is as content data distribution media or known as file sharing. File sharing much used to support content data distributed integration using internet network when deal with time, cost and the spread of user in geographical location. When using content data distribution media, most of user assuming that transfer rate and transfer time as the factors of effectiveness from working method. While on the other side, transfer rate and transfer time are influenced by many factor like size of data content, network environment condition and capability of file sharing method.

Nowadays, the most of content data distribution method is multicast and Peer-to-Peer (Costa-Montenegro, 2012). But there is no research to discuss deeper relating with performance of each method when used for distributing data content in various sizes and in certain environment conditions. So it is not known which method more effective and efficient for content data distribution process. In order to know performance of multicast and peer-to-peer method, require an example application that apply each method.

In our research, example of multicast based file sharing application were used is UFTP. While example of Peer-to-Peer based file sharing application were used is BitTorrent Sync. In order to support testing and analysis comparation performance from each approach and application were used, relating to the passed network condition for content data distribution process, need support of network emulator tools such as Wide Area Network emulator (WANem). And then, in order to eliminate the other factor that can interfere accuracy of performance testing from each file sharing application such as network policy in physical network, then used virtual testing method with support of Virtualbox tools. Performance criteria of file sharing method said well if having high throughput or transfer rate and low transfer time result when used for distributing data content in various size on certain network environment condition.

***Key words**: File Sharing, Multicast, Peer-to-Peer, UFTP, BitTorrent Sync.*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1 PENDAHULUAN

Pada bab ini dideskripsikan mengenai latar belakang penulis memilih topik penelitian yang membahas mengenai perbandingan performansi metode *file sharing* berbasis *multicast* dengan *Peer-to-Peer* dalam proses distribusi konten data beserta rumusan masalah, tujuan, batasan masalah, manfaat, serta sistematika pembahasan skripsi.

1.1 Latar Belakang

Pemanfaatan fungsi jaringan komputer seiring dengan perkembangan teknologi saat ini semakin beragam dan kompleks. Salah satunya adalah sebagai media distribusi konten data atau *file sharing*. *File sharing* merupakan fungsi internet yang paling populer dan tercatat hampir 40% dari seluruh trafik data yang ada pada tahun 2010 digunakan untuk kegiatan tersebut (Cisco Systems, 2011 disitasi dalam Klumpp, 2013). *File sharing* adalah pendekatan teknologi yang digunakan untuk melakukan distribusi konten data *digital* seperti dokumen, multimedia, grafis, program komputer, gambar dan *e-book* dari satu perangkat ke perangkat lain dengan memanfaatkan protokol jaringan sebagai media transfer. Saat ini, aktifitas *file sharing* telah menjadi bagian penting dalam menunjang produktifitas kerja secara terpadu antar pengguna ketika menghadapi kendala tempat, waktu dan biaya (Osterman Research, Inc., 2014). Dengan semakin tingginya kebutuhan distribusi konten data, lokasi pengguna yang tersebar di wilayah geografis yang luas, serta peningkatan ukuran konten data yang didistribusikan melalui internet, maka diperlukan metode yang efektif dan efisien agar dapat memaksimalkan kinerja *file sharing* dalam mendukung proses distribusi konten data untuk meningkatkan produktifitas kerja secara terpadu melalui jaringan internet.

Salah satu parameter penilaian performansi metode *file sharing* yang efektif dan efisien dapat dilihat berdasarkan kecepatan dan waktu transfer yang dihasilkan dalam proses distribusi konten data (Zhang, 2005). Hal ini karena pada umumnya pengguna awam secara mudah melihat kemampuan mekanisme *file sharing* dari kecepatan dan waktu yang dihasilkan ketika melakukan proses transfer data terlepas dari kondisi jaringan komputer yang dilalui. Sedangkan parameter kecepatan dan waktu transfer pada proses distribusi data sendiri dipengaruhi oleh implementasi protokol komunikasi serta kondisi lingkungan jaringan komputer yang digunakan (Zhang, 2005).

Terdapat beberapa pemanfaatan teknologi dan protokol jaringan yang digunakan untuk menunjang aktifitas *file sharing* dalam proses distribusi konten data. Saat ini metode yang banyak dimanfaatkan dalam mekanisme distribusi konten data yaitu melalui pendekatan *multicast* dan *Peer-to-Peer* (Costa-Montenegro, 2012). *Multicast* adalah metode komunikasi dimana data dikirim dari satu atau lebih *host* ke beberapa *host* lainnya dengan terlebih dahulu membentuk kelompok komunikasi tertentu. Teknologi *multicast* memiliki kelebihan dalam mendukung komunikasi pada jaringan komputer karena tidak membebani trafik

jaringan yang dilalui ketika dilakukan proses distribusi konten data jika dibandingkan dengan metode komunikasi melalui mekanisme *unicast* (Jacobsen, eds., 1998). *Multicast* memanfaatkan protokol UDP pada *transport layer*. Protokol UDP memiliki kemampuan untuk mengeliminasi nilai *delay* propagasi yang dihasilkan pada protokol TCP, dimana hal tersebut merupakan salah satu faktor pembatas dalam proses distribusi konten data (Zhang dan McLeod, 2003).

Sedangkan *Peer-to-Peer* adalah pendekatan arsitektur jaringan yang menerapkan komunikasi secara terdesentralisasi (Camarillo, ed., 2009). Metode ini digunakan untuk mengatasi permasalahan dalam proses distribusi konten data pada arsitektur *client-server*. Pendekatan yang digunakan dalam arsitektur *Peer-to-Peer* memiliki karakteristik seperti kontrol secara terdesentralisasi, mendukung skalabilitas, dan *robustness*. Dalam jaringan *Peer-to-Peer* setiap *endnode* yang ikut serta dalam proses distribusi konten data disebut dengan *peer*. Setiap *peer* mampu berperan sebagai *client* maupun *server* terhadap *peer* lainnya. Sehingga seluruh bagian dari jaringan *Peer-to-Peer* ikut serta dalam mendistibusikan konten data.

Untuk dapat membandingkan kinerja atau performansi dari metode *multicast* dan *Peer-to-Peer* dalam proses distribusi konten data, maka digunakan contoh aplikasi yang menerapkan teknologi dari masing-masing metode tersebut. Contoh aplikasi yang digunakan yaitu UFTP dan BitTorrent Sync. UFTP adalah aplikasi transfer konten data berbasis *open source* yang menggunakan pendekatan *multicast* dan didesain untuk menunjang proses distribusi konten data dalam ukuran besar pada beberapa *receiver* secara simultan (Bush, 2015). UFTP juga menerapkan teknologi transfer data melalui mekanisme *TCP-Friendly Multicast Congestion Control* (TFMCC). TFMCC adalah mekanisme yang digunakan untuk mendukung stabilitas dan responsifitas dari setiap *host* yang terkait dengan proses distribusi konten data melalui mekanisme *congestion control*.

Sedangkan BitTorrent Sync (BTSync) adalah aplikasi *file sharing* berbasis *Peer-to-Peer* yang dikembangkan oleh BitTorrent Inc., dan dirilis pertama kali dalam versi *alpha* pada bulan April 2013 (BitTorrent Inc., 2013 disitasi dalam Farina, Scanlon dan Kechadi, 2014). Konten data yang didistribusikan pada sistem, dapat diperoleh dari satu atau beberapa *peer* yang telah memiliki seluruh atau sebagian konten data yang didistribusikan, kemudian dilakukan proses distribusi lanjut secara simultan pada satu atau lebih *peer* lainnya. BitTorrent Sync memanfaatkan mekanisme *Distributed Hash Table* (DHT) untuk memastikan *peer* mana saja yang dapat berpartisipasi dalam distribusi konten data. Mekanisme DHT dilakukan melalui pertukaran tabel yang terdiri dari *key* dan *value* dari bagian-bagian konten data yang didistribusikan secara terdesentralisasi (Farina, Scanlon dan Kechadi, 2014). Protokol BitTorrent pada BTSync memungkinkan seluruh *peer* dalam sistem ikut serta dalam mendistribusikan konten data, sehingga semakin banyak *peer* yang terhubung dalam sistem, maka proses transfer yang berjalan akan semakin cepat.

Setiap metode *file sharing* dengan masing-masing contoh aplikasi yang dijelaskan sebelumnya memiliki keunggulan dalam mendukung proses distribusi

konten data. Sehingga perlu dilakukan penelitian lebih lanjut yang membahas mengenai analisis perbandingan performansi dari masing-masing aplikasi *file sharing* tersebut ketika digunakan untuk melakukan proses distribusi konten data pada berbagai kondisi lingkungan jaringan komputer tertentu. Dari penjelasan diatas maka penulis melakukan penelitian yang berjudul "Analisis Perbandingan Performansi Metode *File Sharing* berbasis *Multicast* dengan *Peer-to-Peer* dalam Proses Distribusi Konten Data". Hasil penelitian diharapkan mampu menjadi referensi dalam menentukan metode *file sharing* yang sesuai untuk mendukung proses distribusi konten data khususnya dalam jaringan komputer lokal.

1.2 Rumusan Masalah

Berdasarkan uraian latar belakang diatas, maka rumusan masalah dari penelitian ini adalah sebagai berikut:

1. Bagaimana mengimplementasikan lingkungan pengujian sistem jaringan komputer lokal yang digunakan dalam proses distribusi konten data melalui metode *file sharing*?
2. Bagaimana prosedur pengumpulan data perbandingan performansi pada contoh aplikasi metode *file sharing* dalam proses distribusi konten data?
3. Bagaimana melakukan analisis perbandingan performansi dari contoh aplikasi metode *file sharing* pada lingkungan pengujian sistem jaringan komputer lokal?
4. Bagaimana hasil analisis perbandingan performansi *throughput* yang dihasilkan metode *file sharing* berbasis *multicast* dengan *Peer-to-Peer* ketika digunakan dalam proses distribusi konten data?

1.3 Tujuan

Tujuan dari penelitian ini antara lain sebagai berikut:

1. Mengimplementasikan lingkungan pengujian sistem yang digunakan untuk mengukur performansi aplikasi *file sharing* dalam proses distribusi konten data.
2. Melakukan analisis perbandingan performansi metode *file sharing* berbasis *multicast* dan *Peer-to-Peer* dengan masing-masing contoh aplikasi yang digunakan yaitu UFTP melalui mekanisme TFMCC dan BitTorrent Sync melalui mekanisme DHT dalam proses distribusi konten data pada lingkungan pengujian sistem.
3. Mengetahui pengaruh dari kondisi jaringan komputer lokal yang digunakan dan ukuran konten data yang didistribusikan terhadap performansi aplikasi UFTP dan BitTorrent Sync.
4. Mengetahui performansi yang dihasilkan dari masing-masing aplikasi *file sharing* ketika digunakan dalam proses distribusi konten data pada sistem jaringan komputer lokal.

1.4 Manfaat

Manfaat dari penelitian ini adalah sebagai berikut:

1. Memperoleh hasil analisis perbandingan performansi dari metode *file sharing* berbasis *multicast* dengan *Peer-to-Peer* berdasarkan contoh aplikasi UFTP melalui mekanisme TFMCC dan BitTorrent Sync melalui mekanisme DHT dalam proses distribusi konten data pada jaringan komputer lokal.
2. Menghasilkan acuan pemilihan aplikasi *file sharing* yang efektif dan efisien untuk melakukan proses distribusi konten data.

1.5 Batasan Masalah

Batasan masalah yang digunakan dalam penelitian ini antara lain sebagai berikut:

1. Metode *file sharing* yang dianalisis yaitu *multicast* dan *Peer-to-Peer*.
2. Contoh aplikasi *file sharing* yang diujikan pada penelitian ini adalah UFTP dan BitTorrent Sync.
3. Mekanisme transfer data yang digunakan pada aplikasi UFTP adalah *TCP-Friendly Multicast Congestion Control* (TFMCC).
4. Mekanisme distribusi konten data yang digunakan pada aplikasi BitTorrent Sync adalah *Distributed Hash Table* (DHT).
5. Lingkungan pengujian sistem yang digunakan dalam penelitian adalah jaringan komputer lokal dalam satu *subnet*.
6. Kriteria yang diujikan dalam penelitian yaitu ukuran konten data yang didistribusikan serta kondisi lingkungan jaringan komputer lokal yang terdiri dari parameter *Round Trip Time* (RTT) *delay* atau latensi dan *packet loss*.
7. Spesifikasi *hardware* yang digunakan pada masing-masing *host* yang terkait dalam sistem distribusi konten data terdiri dari memori (RAM) sebesar 512 MB dan media penyimpanan *(storage)* sebesar 25 GB.
8. Sistem operasi yang digunakan pada setiap *host* yaitu Windows 7 Ultimate 64-bit.
9. Perbandingan performansi yang dianalisis dari masing-masing aplikasi *file sharing* yaitu *throughput* atau kecepatan transfer data.

1.6 Sistematika Pembahasan

Pembuatan skripsi ini disusun dengan sistematika pembahasan sebagai berikut:

BAB I PENDAHULUAN

Berisi latar belakang penulisan dan pemilihan topik penelitian yang dilakukan oleh penulis, termasuk rumusan masalah, tujuan, batasan masalah, manfaat, serta sistematika pembahasan skripsi mengenai analisis perbandingan metode *file sharing* berbasis *multicast* dan *Peer-to-Peer* dengan contoh aplikasi UFTP dengan BitTorrent Sync dalam proses distribusi konten data.

BAB II LANDASAN KEPUSTAKAAN

Berisi teori tentang kajian pustaka dari penelitian sebelumnya yang terkait dengan penelitian yang dilakukan penulis serta beberapa teori digunakan untuk menunjang penelitian ini seperti *file sharing*, *multicast*, *Peer-to-Peer*, UFTP serta TFMCC, BitTorrent Sync serta DHT, Virtualbox, WANem dan Wireshark.

BAB III METODOLOGI PENELITIAN

Berisi metode atau prosedur yang digunakan dalam penelitian ini yang terdiri dari studi literatur mengenai analisis performansi metode *file sharing* dalam proses distribusi konten data, rancangan lingkungan pengujian sistem, analisis kebutuhan lingkungan pengujian sistem, implementasi lingkungan pengujian sistem, pengumpulan data perbandingan performansi, analisis hasil perbandingan performansi dan pengambilan kesimpulan.

BAB IV IMPLEMENTASI DAN PENGUJIAN

Berisi prosedur implementasi dari rancangan ruang lingkup pengujian sistem yang digunakan untuk menunjang proses distribusi konten data melalui aplikasi *file sharing*. Selain itu juga dijelaskan mengenai prosedur dalam melakukan perbandingan performansi dari masing-masing aplikasi *file sharing*.

BAB V HASIL DAN ANALISIS

Berisi tentang hasil perbandingan performansi dari masing-masing aplikasi *file sharing* berdasarakan prosedur yang dilakukan pada bab sebelumnya dan analisis terhadap hasil perbandingan performansi yang didapatkan.

BAB VI PENUTUP

Berisi kesimpulan dari penelitian mengenai perbandingan performansi metode *file sharing* berbasis *multicast* dan *Peer-to-Peer* dengan contoh aplikasi UFTP melalui mekanisme TFMCC dan BitTorrent Sync melalui mekanisme DHT dalam proses distribusi konten data serta saran pengembangan penelitian berikutnya oleh penulis.

BAB 2 LANDASAN KEPUSTAKAAN

Pada bab ini dipaparkan mengenai kajian pustaka dan dasar teori terkait dengan penelitian yang dilakukan oleh penulis. Kajian pustaka yang digunakan berdasarkan pada penelitian sebelumnya terkait dengan metode *file sharing* untuk melakukan distribusi konten data, yang melibatkan aplikasi UFTP dan BitTorrent Sync serta konsep mengenai faktor-faktor yang terlibat dalam mekanisme distribusi konten data. Sedangkan dasar teori yang digunakan untuk mendukung penelitian ini antara lain *file sharing*, *multicast*, *Peer-to-Peer*, UFTP serta TFMCC, Bittorent Sync serta DHT, WANem, Virtualbox dan Wireshark.

2.1 Kajian Pustaka

Berdasarkan judul skripsi yang dibahas, penulis menggunakan penelitian relevan yang telah dilakukan sebelumnya. Selain itu juga dibahas mengenai konsep mengenai faktor-faktor yang berpengaruh terhadap mekanisme distribusi konten data. Beberapa penelitian yang digunakan sebagai acuan, membahas mengenai teknologi *file sharing* yang melibatkan aplikasi UFTP dan BitTorrent Sync.

Penelitian pertama adalah penelitian yang dilakukan oleh Leo Truksans, Edgar Znots, dan Guntis Barzadins dengan judul "*File Transfer Protocol Performance Study for EUMSTAT Meteorological Data Distribution*". Penelitian tersebut bertujuan untuk melakukan analisis perbandingan terhadap beberapa teknologi *file transfer protocol* yang mampu mendukung standar *European Meteorological Union* (EUMSTAT) dan digunakan dalam proses distribusi data meteorologi. Standar yang ditetapkan dalam proses distribusi data meteorologi yaitu metode yang mampu menghasilkan *data rate* antara 300-400 Mbps pada jaringan heterogen. Fokus yang dilakukan penulis dalam penelitian tersebut yaitu membangun lingkungan pengujian sistem yang dapat mensimulasikan kondisi jaringan heterogen untuk mengukur performansi dari masing-masing *file transfer protocol*. Dimana kondisi dari jaringan heterogen sendiri dipengaruhi oleh nilai *RTT delay* dan *packet loss* yang beragam. *File transfer protocol* yang diuji pada penelitian ini antara lain FTP, UFTP, bbFTP, GridFTP dan RSYNC. Hasil dari penelitian tersebut yaitu empat dari lima aplikasi yang diujikan menghasilkan performansi yang maksimal pada kondisi skenario tertentu. Sehingga dapat digunakan untuk mendukung program EUMSTAT sesuai kondisi jaringan heterogen yang ada. *File transfer protocol* yang tidak mencapai target performansi yang diharapkan yaitu bbFTP (Truksans, Znots dan Barzadins, 2011).

Penelitian kedua dilakukan oleh Aleksandr Bakharev dan Eduard Siemens dengan judul penelitian "*Evaluation of Reliable Multicast Implementations with Proposed Adaption for Delivery of Big Data in Wide Area Networks*". Penelitian ini membahas mengenai beberapa solusi distribusi konten data berbasis *open source* menggunakan teknologi *reliable multicast* dan integrasi dari masing-masing solusi tersebut untuk menunjang proses transfer konten data dalam ukuran besar di lingkungan *Content Delivery Network* (CDN). Evaluasi dilakukan pada tiga solusi

pendekatan yaitu UFTP, *NACK-oriented Reliable Multicast* dan *Pragmatic General Multicast* (PGM). Tujuan dalam penelitian ini yaitu untuk membangun solusi baru yang dikembangkan dari ketiga solusi yang dibahas sebelumnya dengan parameter mampu menghasilkan performansi *data rate* diatas 1 Gbps per *stream* dari sebuah konten data ketika dilakukan sebanyak sepuluh kali *streaming* secara simultan. Pengujian dilakukan dalam kondisi jaringan dengan nilai *RTT delay* dan *packet loss* tinggi. Hasil dari penelitian tersebut yaitu *troughput* yang dicapai dari integrasi ketiga solusi tidak lebih dari 250 Mbps meskipun ketika jumlah penerima yang ikut serta dalam proses distribusi konten data sedikit. Selain itu dapat dibuktikan bahwa *packet loss* berpengaruh pada kecepatan dari proses *transmission rate* yang berjalan (Bakharev dan Siemens, 2013).

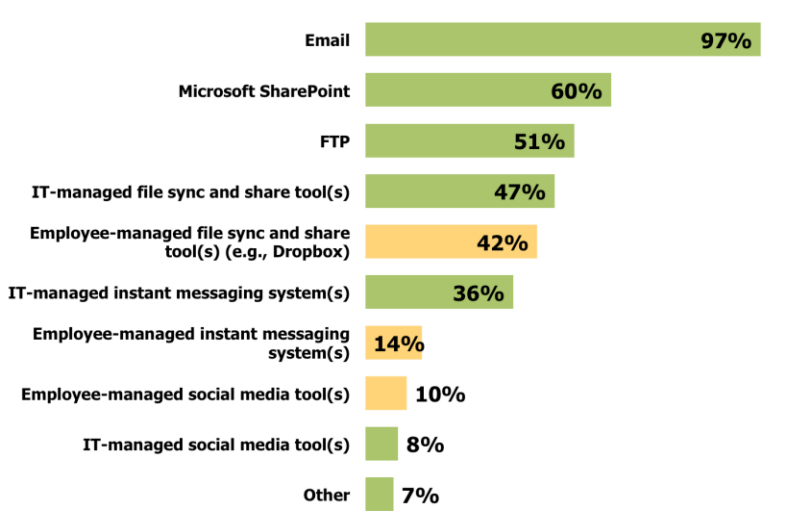
Penelitian ketiga adalah penilitian yang dilakukan oleh Jason Farina, Mark Scanlon, dan M-Tahar Kechadi dengan judul penelitian "*BitTorrent Sync: First Impressions and Digital Forensic Implications*". Tujuan dalam penelitian tersebut yaitu melakukan analisis *digital* forensik terhadap performansi aplikasi *file sharing client* BitTorrent Sync. Performansi yang dianalisis yaitu proses transfer konten data, artefak yang dihasilkan selama proses instalasi dan ketika program sedang berjalan, dan bagian yang masih ditinggalkan ketika dilakukan proses *uninstall* pada aplikasi BitTorrent Sync. Pemilihan aplikasi BitTorrent Sync dalam kajian penelitian tersebut karena BitTorrent Sync mendukung mekanisme seperti *high availability*, *offsite backup* dan *resilent storage* dalam proses distribusi konten data. BitTorrent Sync menyediakan fitur yang terdiri dari *compatibility and availability*, *synchronization*, *no limitations or cost*, *automated backup*, *decentralised technology*, *encrypted data transmission* dan *proprietary technology*. Hasil dari penelitian tersebut yaitu aplikasi BitTorrent Sync dikembangkan tidak untuk menggantikan teknologi BitTorrent dalam hal distribusi konten data, namun lebih kearah pemanfaatan lanjut dari protokol BitTorrent untuk kegiatan distribusi konten data *digital* secara aman dan efisien. Untuk dapat melakukan analisis terhadap proses komunikasi yang berlangsung pada aplikasi BitTorrent Sync dapat menggunakan bantuan aplikasi Wireshark, *tcpdump* dan *libcap* (Farina, Scanlon dan Kechadi, 2014).

Sedangkan dalam penelitian yang dilakukan oleh penulis, dibahas mengenai perbandingan performansi dari metode *file sharing* berbasis *multicast* dengan *Peer-to-Peer* dengan contoh aplikasi yang digunakan yaitu UFTP melalui mekanisme TFMCC dan BitTorrent Sync melalui mekanisme DHT. Penelitian ini bertujuan untuk melakukan analisis perbandingan performansi dari metode *file sharing* berbasis *multicast* dan *Peer-to-Peer* berdasarkan contoh aplikasi UFTP dan BitTorrent Sync, serta solusi metode distribusi konten data yang efektif dan efisien ketika digunakan dalam berbagai kemungkinan kondisi di lingkungan jaringan komputer lokal. Penelitian dilakukan pada lingkungan pengujian sistem jaringan komputer secara virtual dengan menggunakan aplikasi Virtualbox. Untuk melakukan skenario kondisi lingkungan pengujian maka digunakan aplikasi *Wide Area Network Emulator* (WANem) yang berfungsi mengemulasikan variabel-variabel kondisi yang terdapat pada sebuah jaringan komputer dan berpengaruh terhadap proses distribusi konten data seperti *Round Trip Time* (RTT) *delay* atau

latensi dan *packet loss* (Bakharev dan Siemens, 2013). *RTT delay* adalah waktu yang dibutuhkan sebuah paket ketika ditransmisikan dari *client* ke *server* dan kembali ke *client* kembali. Nilai *RTT delay* sendiri diperoleh dari penjumlahan beberapa jenis *delay* yaitu *propagation delay*, *queuing delay* dan *processing delay* (Kurose dan Ross, 2000). Sedangkan *packet loss* adalah hilangnya suatu paket data ketika dilakukan transmisi yang disebabkan berbagai macam kondisi yang terdapat pada jaringan yang dilalui. Seperti ketika paket yang datang lebih cepat dibandingkan kemampuan perangkat komunikasi untuk memproses paket data yang ditransmisikan, sehingga menimbulkan *congestion* atau kemacetan yang mengakibatkan paket data tidak sampai ke tujuan dan diperlukan mekanisme pengiriman ulang atau retransmisi (Kurose dan Ross, 2000). Untuk mengetahui pengaruh dari setiap variabel-variabel kondisi lingkungan pengujian sistem, maka digunakan variabel *throughput* sebagai parameter performansi yang dibandingkan dari masing-masing metode *file sharing* ketika melakukan proses distribusi konten data. *Throughput adalah* kecepatan atau *rate* yang berjalan pada jaringan komputer ketika terjadi transmisi data dari *sender* ke *receiver*. *Throughput* merupakan parameter utama yang digunakan dalam menentukan performansi pada komunikasi TCP/IP (Kurose dan Ross, 2000). *Throughput* dari masing-masing metode *file sharing* dianalisis menggunakan bantuan aplikasi Wireshark.

2.2 *File Sharing*

File sharing adalah pendekatan teknologi yang digunakan untuk melakukan distribusi konten data *digital* seperti dokumen, multimedia, grafis, program komputer, gambar dan *e-book* dari satu perangkat ke perangkat lain dengan memanfaatkan protokol jaringan komputer sebagai media transfer konten data. Pada awal tahun 1990 metode *file sharing* mulai banyak dikembangkan seperti *File Transfer Protocol* (FTP), *hotline* dan *Internet Relay Chat* (IRC). Selain itu juga terdapat metode *file sharing* berbasis sistem operasi seperti *Network File System* (NFS).



Gambar 2.1 Grafik pemanfaatan teknologi *file sharing* pada perusahaan

Sumber: Oesterman, Inc. (2014)

Saat ini teknologi *file sharing* telah dikembangkan lebih lanjut untuk menunjang produktifitas pengguna terutama dalam sistem jaringan informasi di perusahaan atau instansi. Salah satunya untuk menunjang proses distribusi konten data perusahaan yang bersifat penting terkait kerja terpadu yang dilakukan antar karyawan (Osterman Research, Inc., 2014). Terdapat banyak teknologi *file sharing* yang dimanfaatkan pada jaringan lokal perusahaan hingga saat ini. **Gambar 2.1** menunjukkan beberapa teknologi *file sharing* yang pada umumnya digunakan di suatu perusahaan atau instansi saat ini.

Meskipun telah banyak teknologi *file sharing* yang digunakan sebagai media distribusi konten data, masih banyak kelemahan yang muncul dari setiap teknologi yang ada ketika proses distribusi konten data digunakan pada kondisi lingkungan jaringan komputer tertentu, antara lain sebagai berikut (Osterman Research, 2014):

- Kebanyakan metode *file sharing* tidak terintegrasi dengan sistem jaringan komputer perusahaan dan bekerja diluar kontrol dari staff IT sehingga sulit untuk dapat dikontrol secara terpusat melalui jaringan internal perusahaan.
- Kebanyakan teknologi *file sharing* tidak menyediakan fitur pencarian konten data secara terpusat. Terlebih lagi ketika dilakukan distribusi konten data melalui sistem *file sharing* yang berbeda-beda, sehingga akan menyulitkan ketika dibutuhkan proses audit data.
- Sistem FTP yang diterapkan pada kebanyakan perusahaan, menawarkan kelebihan seperti mampu mendistribusikan konten data dalam ukuran besar. Namun sistem FTP memiliki kelemahan seperti tingkat keamanan yang rendah karena kebanyakan pengguna saling berbagi akun *login credential*. Sehingga rentan terhadap kebocoran data.
- Permasalahan lain dengan FTP adalah realibilitas yang buruk, karena FTP tidak dilengkapi dengan mekanisme *recovery error* ketika proses transfer konten data sedang berlangsung. Selain itu FTP juga memiliki kelemahan lain seperti proses transfer konten data dalam ukuran besar melalui FTP kurang *reliable* pada beberapa kondisi jaringan tertentu, keterbatasan penggunaan *bandwidth* dan kecepatan transfer data yang rendah. Sehingga proses *download* akan berlangsung lama.
- Beberapa perusahaan atau instansi memanfaatkan aplikasi Microsoft SharePoint untuk mengatasi permasalahan manajemen *file sharing* pada jaringan internal perusahaan. Menurut riset yang dilakukan oleh Osterman Research, Inc., SharePoint merupakan aplikasi yang mampu mengatasi permasalahan distribusi konten data dalam jaringan internal perusahaan. Namun dibutuhkan waktu dan pemahan yang mendalam agar seluruh pengguna atau karyawan benar-benar dapat memanfaatkan aplikasi SharePoint secara maksimal untuk menunjang proses distribusi konten data di perusahaan.
- Proses manajemen *file sharing* membutuhkan alokasi waktu khusus bagi staff IT. Jika sistem *file sharing* yang digunakan adalah FTP, staff IT harus

selalu melakukan pemantauan terhadap *server* dan berhadapan dengan permasalahan yang muncul dari komplain pengguna setiap saat. Jika sistem *file sharing* yang digunakan adalah *email*, staff IT harus berhadapan dengan permasalahan yang menyebabkan kuota sistem penyimpanan pada *mailbox* cepat penuh.

- Pada umumnya sistem *email* dengan fitur *attachment* tidak dilengkapi dengan mekanisme enkripsi. Hal tersebut dapat meningkatkan potensi kebocoran data oleh pihak-pihak yang tidak memiliki kepentingan terhadap data penting perusahaan.
- Kebanyakan pengguna juga masih memanfaatkan *physical media* seperti CD-ROM atau DVD-ROM untuk mendistribusikan konten data dalam ukuran besar. Dengan pendekatan seperti ini tidak hanya meningkatkan kerentanan terhadap keamanan distribusi konten data, namun juga meningkatkan biaya yang harus dikeluarkan oleh perusahaan untuk dapat memanfaatkan *physical device* tersebut.
- Beberapa perusahaan membutuhkan sistem *file sharing* yang mampu bekerja dalam daur proses secara berkala. Jika sistem *file sharing* yang digunakan masih menerapkan sistem *email* atau *physical device* maka daur proses *file sharing* tidak dapat berjalan dengan efektif. Sehingga perlu suatu sistem *file sharing* yang menyediakan fitur sinkronisasi data secara terintegrasi.

Terdapat beberapa macam motivasi yang digunakan sebagai dasar implementasi sistem *file sharing* pada perusahaan. Berikut tujuan atau motivasi yang digunakan sebagai dasar pengembangan sistem *file sharing* (Whitney, 1969):

- *File sharing* mampu untuk digunakan untuk melakukan proses distribusi konten data tanpa ada batasan terhadap ukuran dari konten yang didistribusikan, proses transfer konten data tetap berjalan dengan baik dan tidak menggangu stabilitas performansi dari seluruh bagian sistem jaringan yang dilalui.
- Mudah untuk digunakan dan tidak membutuhkan waktu yang lama untuk dipelajari oleh seluruh elemen yang terkait dengan aktifitas *file sharing*.
- Memastikan proses distribusi konten data dapat berjalan dengan aman dengan menerapkan sistem enkripsi pada setiap bagian dari konten data yang didistribusikan.
- Mudah untuk dilakukan implementasi pada sistem yang terkait dengan proses distribusi konten data.
- Mudah untuk diintegrasikan dengan lingkungan jaringan yang sudah ada sebelumnya.

Dari beberapa kriteria yang dijelaskan diatas, pada penelitian ini penulis berfokus terhadap motivasi atau tujuan pada poin pertama. Yaitu terkait performansi yang dihasilkan oleh metode *file sharing* ketika menghadapi kebutuhan transfer konten data dalam ukuran besar serta dalam berbagai kemungkinan kondisi lingkungan jaringan komputer ketika dilakukan proses distribusi konten data.

2.3 *Multicast*

Multicast atau juga disebut dengan *IP-multicast* adalah mekanisme transmisi data yang digunakan untuk mendukung skalabilitas dalam sistem jaringan komputer dengan memanfaatkan mekanisme *datagram*. *Datagram* dikirimkan ke *host group* yang terdiri dari beberapa *host* dan masing-masing diidentifikasikan dengan *IP destination address* tertentu. Datagram yang dikirim ke *host group* diteruskan ke seluruh *host* lain yang tergabung dalam *host group* dengan mekanisme *best-effort* dengan tidak memberikan jaminan bahwa seluruh *datagram* yang dikirimkan dapat diterima secara utuh dan berurutan pada seluruh *host* yang terkait (Deering, 1989). *Multicast* bertujuan untuk meningkatkan fungsionalitas komunikasi pada internet dengan meminimalkan pengaruh yang dihasilkan terhadap beban kerja jaringan yang dilalui (Quinn dan Almeroth, 2001). *Multicast* memungkinkan pengiriman data oleh *server* pada *IP packet-switched network* lebih efisien dengan menggunakan *single data stream* pada *local multicast router* yang kemudian didistribusikan lanjut sesuai dengan jumlah *receiver host* yang ada. Sehingga *sender host* hanya melakukan proses *single transmission* untuk mendistribusikan data ke beberapa *receiver host* lain yang tergabung dalam *host group*.

Berdasarkan mekanisme transmisi yang dilakukan, *multicast* dibagi menjadi tiga tipe, antara lain (IXIA, 2005):

- *One-to-many*
  Mekanisme transmisi *multicast* yang dilakukan dari satu *single host* yang berperan sebagai *sender* ke lebih dari satu *host* lainnya yang berperan sebagai *receiver* serta tergabung pada satu *group multicast* yang sama. Contoh penerapan dari model *one-to-many* yaitu sistem *broadcasting* pada televisi dan radio. Model ini juga digunakan dalam sistem distribusi konten berbasis *push media* untuk mendapatkan informasi seperti berita dan prakiraan cuaca secara aktual.
- *Many-to-many*
  Model ini pada umumnya diterapkan untuk mendukung sistem simulasi interaktif terdistribusi. Contoh penerapan model *many-to-many* yaitu pada *interactive multi-player game*, *jam session*, *multimedia teleconference*, *group chat*, *shared editing and collaborating tools* dan *parallel computing*.
- *Many-to-one*
  Model *many-to-one* pada umumnya digunakan untuk mendukung mekanisme komunikasi *require and response*. Contoh penerapan model ini yaitu pada sistem yang menerapkan mekanisme poling dan pengumpulan data.

2.3.1 *Multicast Group*

Untuk dapat berkomunikasi melalui mekanisme *multicast*, suatu *host* harus bergabung dalam *host group* atau *Source-Spesific Multicast* (SSM) melalui *local multicast router*. *Host group* bersifat dinamis, dimana setiap anggota *host group* memungkinkan untuk dapat bergabung dan keluar dari *host group* sewaktu-

waktu. *Host group* dalam *multicast* tidak dipengaruhi oleh lokasi atau jumlah dari *host* yang tergabung, sehingga anggota dari suatu *host group* dapat menjadi anggota *host group* lainnya. Namun untuk dapat menerima *datagram* dari *host group* tertentu, suatu *host* harus menjadi anggota dari *host group* tersebut. Dalam proses transmisi, mekanisme *multicast* memanfaatkan *IP address* pada *class D*. Rentang *IP address* pada *class D* yaitu antara 224.0.0.0 sampai dengan 239.255.255.255.

Terdapat tiga *level* penyesuaian terhadap *host* yang terkait dalam mekanisme *multicast*, antara lain sebagai berikut (Deering, 1989):

- *Level 0*: pada *level* ini, IP yang digunakan tidak mendukung *IP-multicasting*. *Host* yang terdapat pada *level 0*, secara umum tidak terpengaruh terhadap aktivitas *multicasting* yang berjalan pada jaringan komputer.
- *Level 1*: *host* pada *level 1* ikut serta dalam beberapa proses *multicasting* seperti *resource location* atau *status reporting*, namun *host* tersebut tidak diijinkan untuk bergabung ke dalam *host group*.
- *Level 2*: pada *level 2 host* dapat bergabung maupun keluar dari *host group* untuk melakukan transmisi *datagram*. *Host* yang berada pada *level 2* menggunakan ekstensi IP tambahan seperti *Internet Group Management Protocol* (IGMP) untuk dapat ikut serta secara penuh dalam proses *multicasting*.

2.3.2 *Multicast Addresing*

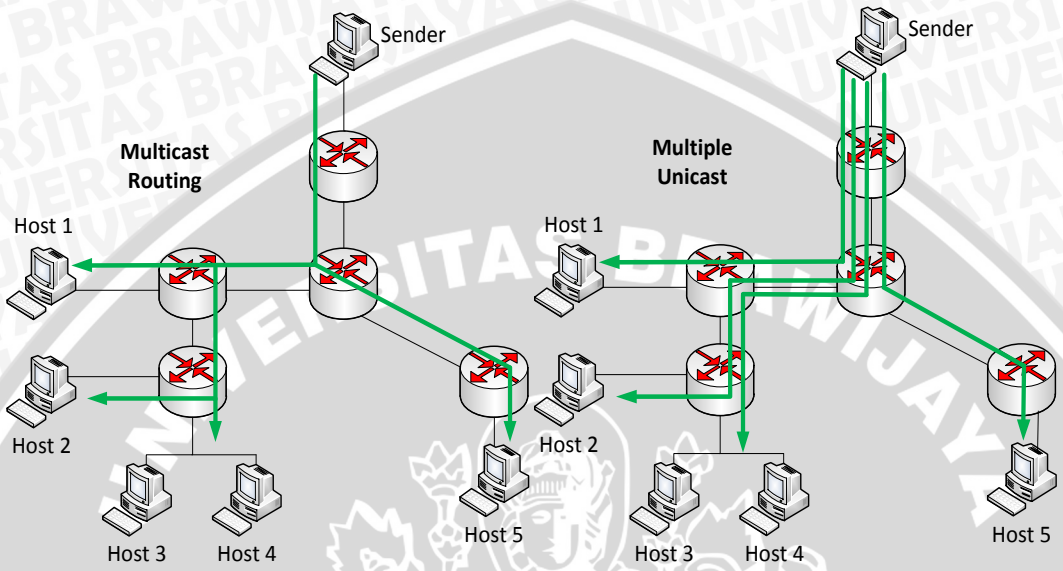
Multicast menggunakan *IP address* pada *class D* untuk berkomunikasi. Rentang *IP address* pada *class D* dimulai dari 224.0.0.0 sampai dengan 239.255.255.255. Pengalamatan IP pada *multicast group* dapat bersifat permanen atau sementara. *IP multicast address* yang bersifat permanen, ditetapkan dan distandarisasikan oleh *Internet Assigned Numbers Authority* (IANA) untuk seluruh *multicast group* yang ada (IXIA, 2005). Contoh *IP multicast address* permanen yaitu *IP address* 224.0.0.1 yang ditetapkan untuk seluruh *host* dan *router* pada suatu jaringan, 224.0.0.2 ditetapkan sebagai alamat seluruh *router group multicast*. Selain itu, terdapat *IP multicast address* lain yang ditetapkan sebagai alamat permanen pada masing-masing *group multicast* tertentu.

Alamat dengan *prefix* 233.0.0.0/8 digunakan untuk *owner* atau pemilik dari *autonomous system*. Alamat dari *autonomous system* dikonversikan ke dalam *16-bit number* dan digunakan untuk mengidentifikasi *byte* kedua dan ketiga dari *IP multicast address*. Kemudian *autonomous system* dapat memanfaatkan 256 *IP address* pada *byte* keempat sebagai *IP address* dinamis pada masing-masing anggota *multicast group* dalam melakukan transmisi data. Seluruh *IP address* yang tidak termasuk dalam *IP address* permanen, dapat berubah-ubah secara dinamis dan bersifat sementara dalam suatu *multicast group*.

2.3.3 *Multicast Routing*

Multicast routing adalah mekanisme yang digunakan untuk melakukan pertukaran pesan melaui *forward packet multicast* pada *router* yang tergabung

dalam distribution tree (IXIA, 2005). Jaringan komputer melakukan *routing* terhadap trafik dari *source host* ke *receiver group host* yang memiliki *IP address multicast group* yang sama, *receiver host* yang tergabung pada *multicast group* dapat tersebar pada jaringan komputer yang luas. Setiap *receiver host* dapat bergabung dan meninggalkan *multicast group* setiap saat dalam kondisi-kondisi tertentu secara sepihak.



Gambar 2.2 Perbedaan *unicast routing* dan *multicast routing*

Sumber: IXIA (2005)

Dengan *multicast routing*, *router* atau *switch* mengirimkan paket ke beberapa *host* tertentu tanpa melakukan proses *broadcast* atau transmisi *unicast* secara berulang-ulang, sehingga tidak terjadi *routing loop* yang membuat trafik jaringan semakin padat seperti yang pada **Gambar 2.2**. *Receiver host* yang menjadi anggota suatu *group multicast* dapat berada pada satu jaringan lokal LAN yang sama, di suatu *private network* tertentu, maupun dilokasi lain yang tersebar dalam jaringan internet.

Suatu jaringan yang mendukung *multicast* harus mampu membentuk *distribution tree* yang menghubungkan antara *source* ke seluruh *destination*. Tujuan utama dari metode *distribution tree* pada *multicast* yaitu untuk memastikan paket dapat diterima oleh seluruh *receiver host* melalui *link* yang ada sebanyak satu kali. Sedangkan tujuan dari *multicast routing* yaitu untuk menentukan *distribution tree* dari setiap *router* atau *switch* yang menghubungkan *source* ke seluruh *destination host group* yang ada (IXIA, 2005).

2.4 *Peer-to-Peer*

Peer-to-Peer memiliki berbagai macam definisi sesuai dimana sistem tersebut digunakan (Camarillo, ed., 2009). Dalam sistem terdistribusi, *Peer-to-Peer* didefinisikan sebagai model komunikasi terdesentralisasi dimana setiap *endpoint* memiliki peran yang sama sebagai *client* sekaligus *server* dari *endpoint* lainnya

yang terkait pada suatu sistem komunikasi. Istilah *peer* digunakan untuk setiap *endpoint* yang tergabung dalam sistem *Peer-to-Peer* itu sendiri.

Terdapat dua fungsi yang digunakan dalam proses bagaimana suatu *endpoint* dapat bergabung menjadi *peer* pada suatu sistem *Peer-to-Peer,* antara lain (Camarillo, ed., 2009):

- *Enrollment function*: fungsi yang memanfaatkan proses authentifikasi dan authorisasi terlebih dahulu pada setiap *endpoint* yang akan bergabung dalam sistem *Peer-to-Peer*.
- *Peer discovery function*: fungsi yang mengijinkan suatu *endpoint* agar dapat terhubung dalam sistem *Peer-to-Peer* melalui koneksi dengan satu atau lebih *endpoint* lain yang lebih dahulu tergabung sebagai anggota *peer* pada suatu sistem.

Berdasarkan *service* yang disediakan, sistem *Peer-to-Peer* dapat terdiri dari beberapa fungsi, antara lain (Camarillo, ed., 2009):

- *Data indexing function*: fungsi yang digunakan untuk melakukan proses pengindeksan terhadap data yang disimpan dalam sistem.
- *Data storage function*: fungsi yang digunakan untuk melakukan proses *storing* dan *retrieving* data dari sistem.
- *Computation function*: fungsi yang digunakan untuk melakukan proses komputasi pada sistem, seperti pemrosesan data atau *real-time media processing*.
- *Message transport function*: fungsi yang digunakan dalam proses pertukaran pesan antar *peer*.

Suatu sistem *Peer-to-Peer* dapat terdiri dari satu atau lebih fungsi diatas, tergantung pada servis yang disediakan oleh sistem itu sendiri. Dalam hal distribusi konten data atau *file sharing*, *Peer-to-Peer* banyak dimanfaatkan karena memiliki keunggulan dalam hal skalabilitas (Camarillo, ed., 2009). Semakin besar cakupan dari sistem dalam melakukan distribusi konten data tertentu, maka tingkat skalabilitas dari sistem *Peer-to-Peer* juga akan semakin tinggi. Ketika terdapat *peer* yang menjadi pemilik dari suatu konten data tertentu melakukan distribusi pada *peer* lainnya, maka *peer* yang menerima konten data yang didistribusikan tersebut juga akan ikut serta dalam proses distribusi pada *peer* berikutnya.

2.5 UFTP

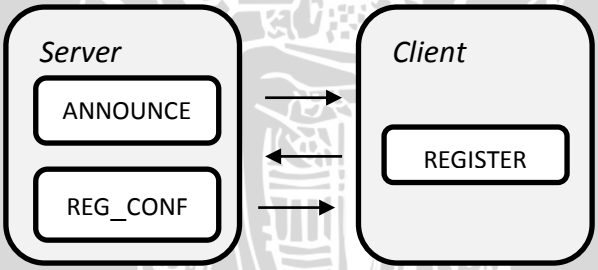
UFTP adalah aplikasi *transfer file* berbasis *multicast* terenkripsi yang digunakan untuk mendistribusikan konten data secara aman, *reliable* dan efisien pada beberapa *receiver* secara simultan dengan memanfaatkan protokol UDP pada *transport layer*. UFTP juga memungkinkan proses distribusi konten data pada beberapa *receiver* melalui *satellite link* dengan komunikasi dua arah, dimana penggunaan protokol TCP pada kasus ini sangat tidak efisien karena akan menghasilkan *delay* yang sangat tinggi. Fitur enkripsi *multicast* pada UFTP memanfaatkan ekstensi *Transport Layer Security* (TLS) yang memungkinkan

beberapa *receiver* saling bertukar *common key* untuk meningkatkan keamanan distribusi konten data (Bush, 2015).

UFTP juga memiliki kemampuan untuk dapat berjalan melalui beberapa jaringan yang dipisahkan oleh satu atau lebih *firewall (NAT transversal)* dengan bantuan *UFTP proxy server* tanpa menggunakan mekanisme *end-to-end tunneling* secara menyeluruh pada *link* yang dilewati. *UFTP proxy* menghimpun respon dari setiap *group receiver* untuk mendukung skalabilitas proses distribusi konten data.

UFTP bekerja melalui 3 fase tahapan, berikut penjelasan dari masing-masing fase pada aplikasi UFTP (Bush, 2015):

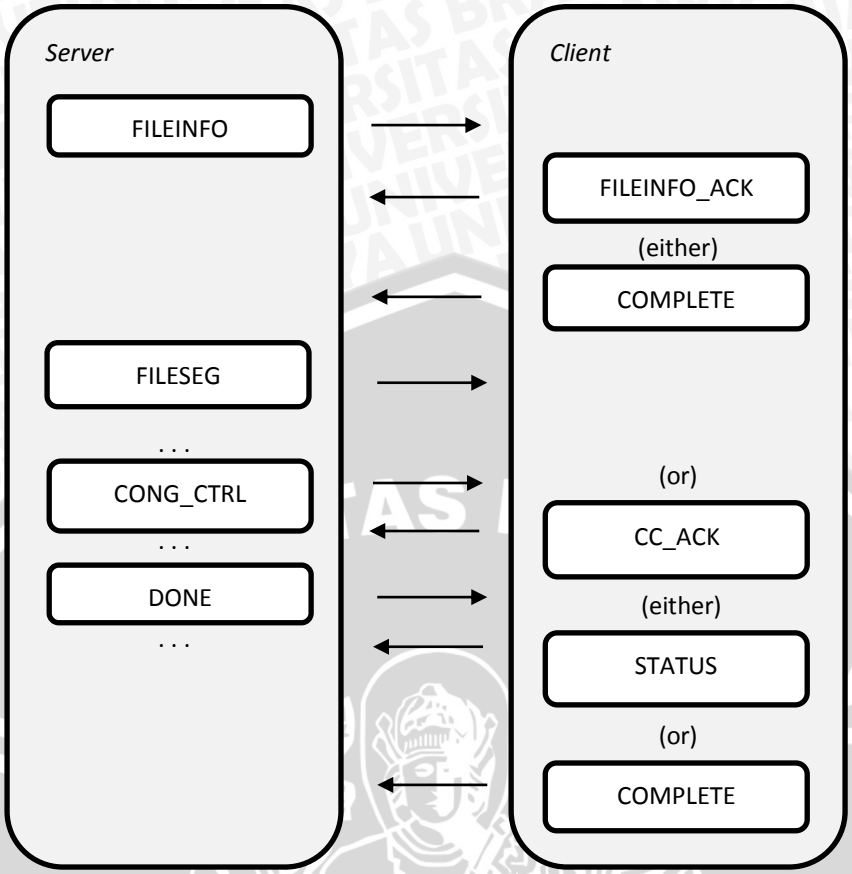
- *Announce/register phase*
  Fase dimana proses inisialisasi *session file transfer* dan parameter enkripsi dilakukan. Aliran proses pertukaran message yang berjalan pada fase ini dijelaskan pada **Gambar 2.3**. *Server* mengirim *announce message* melalui *public multicast address* dimana seluruh *client* yang terhubung dapat menerima *announce message* yang dikirimkan. Pesan yang dikirimkan oleh *server* melalui proses *announcement* dispesifikasikan sesuai dengan parameter tertentu yang akan digunakan ketika proses transfer data berlangsung melalui *private multicast address*. Kemudian *client* mengirimkan *register message* untuk merespon *announce message* yang dikirim. Proses berikutnya *server* akan mengirimkan pesan konfirmasi berupa *reg_conf message* untuk memastikan *client* tergabung dalam proses distribusi konten data yang akan dilakukan.



Gambar 2.3 Aliran *message* pada *announce/register phase* dalam aplikasi UFTP

Sumber: Bush (2015)

- *File transfer phase*
  File transfer phase dibagi menjadi dua bagian yaitu *file info phase* dan *data transfer phase*. Sesuai dengan **Gambar 2.4**, *file transfer phase* dimulai dengan *file info phase* yang berfungsi untuk mengawali *file* atau konten data pertama yang dikirimkan. *Server* mengirim sebuah *fileinfo message* yang menjabarkan informasi mengenai konten data yang akan dikirim seperti nama, ukuran dan bagaimana konten data akan dipecah menjadi beberapa bagian. Konten data dipecah menjadi beberapa bagian dan disebut dengan *block*, kemudian dari beberapa *block* yang ada dikelompokkan menjadi beberapa *section*.



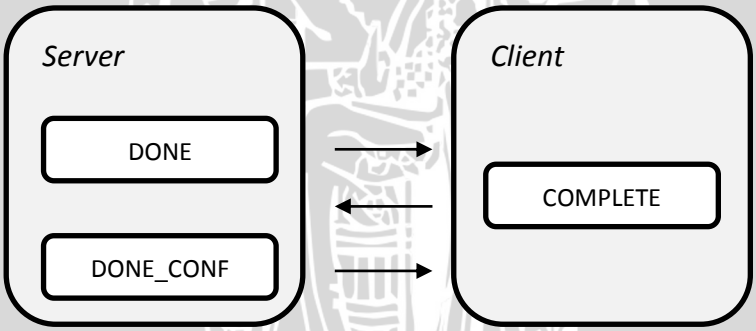
Gambar 2.4 Aliran *message* pada *file transfer phase* dalam aplikasi UFTP

Sumber: Bush (2015)

Selanjutnya jumlah *block* dan *section* akan disertakan pada informasi awal ketika konten data akan dikirimkan. Pada *data transfer phase*, paket data yang terdiri dari beberapa *block* dan *section* dikirimkan oleh *server* dengan kecepatan *transfer rate* yang dapat disesuaikan secara manual oleh pengguna. Selain itu kecepatan *transfer rate* juga dapat dilakukan secara otomatis menyesuaikan kondisi dari sistem jaringan yang ada melalui mekanisme *TCP-Friendly Multicast Congestion Control* (TFMCC). Fitur ini yang nantinya akan dianalisis performansinya ketika diterapkan dalam proses ditribusi konten data. Karena UDP tidak menyediakan mekanisme untuk memastikan apakah *file* yang dikirim benar-benar sampai ke tujuan secara urut, setiap *block* dilabeli dengan nomor urut sehingga pada sisi *client* dapat dilakukan proses penyusunan ulang dari *block-block* tersebut. Ketika *server* telah selesai melakukan proses pengiriman seluruh paket data, *server* mengirimkan sebuah pesan konfirmasi berupa *done message* bahwa seluruh bagian paket yang didistribusikan telah selesai dan berhasil dikirimkan. Ketika *client* menerima konfirmasi dari *server* bahwa seluruh paket data telah dikirimkan dan terdapat beberapa *block* yang belum lengkap, maka *client* akan mengirimkan ulang sebuah pesan ke *server* yang berisikan daftar *Negative*

Acknowledgements (NAKs). Ketika *server* menerima NAK dari satu atau lebih *client*, *server* akan mengirimkan ulang *block-block* yang masuk dalam daftar NAK dari seluruh *client* tersebut. Setelah *client* telah menerima seluruh bagian dari *block* yang hilang, *client* mengirimkan *completion message* untuk merespon pesan pengiriman paket data terakhir yang dilakuan oleh *server* bahwa seluruh konten data telah lengkap diterima di sisi *client*. Proses ini berlangsung secara terus menerus hingga seluruh *client* mengirimkan *completion message* ke *server* atau ketika telah mencapai batas waktu yang diijinkan setelah *server* mengirimkan *done message*. *File info phase* dan *data transfer phase* diulang kembali sesuai dengan jumlah *session* dari konten data yang didistribusikan.

- *Completion/confirmation phase*
  Pada fase ini dilakukan proses penghentian *session* antara *server* dan *client*. Proses pertukaran pesan yang berjalan pada *completion/confirmation phase* sesuai dengan **Gambar 2.5**. Fase ini diawali dengan *done message* dari *server* ke *client* yang mengindikasikan akhir dari *session* yang ada. Kemudian *client* merespon dengan mengirimkan *complete message* pada *server*. Apabila *server* telah menerima *completion message* dari *client*, *server* akan mengirimkan *done_conf message* kembali ke *client*.



Gambar 2.5 Aliran *message* pada *file completion/confirmation phase* dalam aplikasi UFTP

Sumber: Bush (2015)

2.6 *TCP-Friendly Multicast Congestion Control* (TFMCC)

TCP-Friendly Multicast Congestion Control (TFMCC) adalah mekanisme *congestion control single-rate* dan *source based* pada *multicast* yang dikembangkan dari metode *unicast* dalam mekanisme *TCP-Friendly Rate Control* (TRFC). Mekanisme TFMCC memungkinkan kondisi jaringan agar tetap stabil dan responsif ketika komunikasi yang berjalan mencakup skala yang besar dan luas (Widmer dan Handley, 2006). Untuk mendukung skalabilitas dari sistem yang berjalan, TFMCC memanfaatkan mekanisme *congestion control* semaksimal mungkin pada setiap *receiver*. Setiap *receiver* secara kontinyu mengirimkan kemampuan penerimaan paket data pada *sender*, sehingga *sender* dapat

menentukan kecepatan transfer paket data yang sesuai pada setiap *receiver* secara proporsional.

TFMCC melakukan perhitungan terhadap kemungkinan *packet loss* untuk menentukan nilai *sending rate* yang sesuai pada setiap *receiver*. Perhitungan tersebut tidak akan berjalan sebagaimana mestinya apabila paket yang berjalan melalui mekanisme TFMCC dalam jumlah yang sangat sedikit per *Round Trip Time* (RTT) *delay*. Pada aplikasi *multicast* ketika digunakan untuk mendistribusikan konten data, nilai *transfer rate* yang berjalan akan berubah-ubah menyesuaikan kemampuan dari kemampuan *receiver.* Hal ini karena mekanisme TFMCC berfungsi untuk menghindari terjadinya *congestion* pada jaringan komputer (Widmer dan Handley, 2006).

Secara umum, mekanisme *congestion control* pada TFMCC bekerja dalam beberapa tahap seperti dibawah ini (Widmer dan Handley, 2006):

- Setiap *receiver* mengirimkan informasi kepada *sender* nilai *packet loss* dan *RTT delay* masing-masing.
- Setiap r*eceiver* kemudian menggunakan informasi tersebut, bersama dengan perhitungan dari *TCP throughput* untuk mendapatkan nilai *TCP-friendly rate* yang sesuai.
- TFMCC menerapkan mekanisme *distributed feedback suppression*, dimana hanya beberapa dari *receiver* yang dapat mengirimkan *feedback* informasi ke *sender*. Hal ini mencegah agar tidak terjadi kepadatan trafik yang mengarah ke *sender* akibat pengiriman *feedback* berlebih. Mekanisme ini juga memungkinkan *receiver* dengan kemampuan penerimaan yang rendah memiliki prioritas yang tinggi dalam mengirimkan *feedback* informasi ke *sender*.
- *Receiver* yang tidak memiliki kesempatan untuk mengirimkan *feedback* informasi dapat tetap melakukan perhitungan *transmission rate* secara berkala ke *sender* dalam bentuk *receiver report*. *Receiver report* berfungsi untuk menginformasikan kepada *sender* mengenai nilai *transmit rate* yang sesuai dan pengukuran terhadap nilai *RTT delay* masing-masing *receiver*.
- *Sender* memilih *receiver* dengan nilai *rate* terendah sebagai nilai *Current Limiting Rate* (CLR). Ketika terdapat nilai *transmission rate* yang lebih rendah diterima oleh *sender*, maka *receiver* yang bersangkutan akan menggantikan nilai CLR dari *receiver* sebelumnya. Sehingga nilai *sending rate* akan dikurangi sesuai dengan nilai perhitungan *rate* dari *receiver* tersebut. Nilai *sending rate* akan meningkat ketika *CLR report* mengirimkan pesan bahwa nilai *sending rate* dari *receiver* tersebut juga ikut meningkat.

Nilai TFMCC bersifat dinamis dan tergantung pada bagaimana proses pengukuran dilakukan dan diaplikasikan serta pada mekanisme *feedback suppression* yang digunakan.

2.7 BitTorrent Sync

BitTorrent Sync (BTSync) adalah aplikasi replikasi yang dikembangkan oleh BitTorrent Inc. dan pertama kali di rilis pada bulan April 2013. BTSync merupakan aplikasi *file transfer* yang tersinkronisasi dan terdesentralisasi pada satu atau lebih *peer* yang terhubung dalam kelompok distribusi konten data. BTSync menerapkan mekanisme transfer data menggunakan *Distributed Hash Table* (DHT) sehingga tidak terdapat kewenangan secara terpusat untuk melakukan manajemen terhadap autentifikasi atau *log data access*. Konten data yang ditransfer melalui BTSync di*download* dari beberapa *peer* dan didistibusikan ke beberapa *peer* lainnya (BitTorrent Inc., 2013 disitasi dalam Farina, Scanlon dan Kechadi, 2014).

Sistem *storage* yang digunakan dalam BTSync dibatasi dengan ukuran *folder* yang digunakan sebagai *destination folder* atau tergantung pada ketersedian *disk space* dari setiap *peer* yang terhubung pada sistem. Konten data yang terdapat pada *storage* masing-masing *peer* dapat di*download* oleh *peer* lainnya tanpa sepengetahuan dari pengguna, selama terhubung dengan jaringan komputer. Kecuali pada *peer* yang berperan sebagai *creator* dari konten data yang didistribusikan. Konten data yang didistribusikan melalui BTSync, dapat diakses apabila seluruh bagian konten data telah berhasil di*download* oleh suatu *peer* (Farina, Scanlon dan Kechadi, 2014).

Dalam melakukan proses distribusi konten data, BTSync memanfaatkan protokol BitTorrent pada *layer* aplikasi. Protokol BitTorrent melakukan proses distribusi konten data dengan memecah konten data menjadi beberapa bagian, dimana setiap pecahan konten data tersebut dapat di manipulasi dan didistribusikan secara terpisah. Ketika terjadi proses transfer konten data antar *peer*, *peer* yang telah selesai melakukan proses *download* dapat ikut serta melakukan proses *upload* dan mendistribusikan data ke *peer* lain yang tergabung dalam *group peer*. Dalam kasus ini, antara satu *peer* dengan *peer* lainnya juga dapat saling bertukar bagian data yang masih belum lengkap sebelum seluruh konten data berhasil di*download* oleh masing-masing *peer* (Farina, Scanlon dan Kechadi, 2014).

2.8 *Wide Area Network Emulator* (WANem)

Wide Area Network emulator (WANem) adalah aplikasi yang digunakan untuk menghasilkan emulasi berbagai macam faktor yang mempengaruhi kondisi lingkungan sistem jaringan komputer sesuai dengan yang diinginkan oleh pengguna. WANem diimplementasikan dan didistribusikan pada sistem operasi Linux Knoppix. Tujuan dari WANem adalah untuk menguji pengaruh dan performansi yang dihasilkan suatu aplikasi terhadap kondisi jaringan komputer tertentu (Kalita dan Rane, 2008).

Kondisi dalam suatu jaringan komputer cenderung bersifat dinamis dan dapat berubah-ubah dalam setiap waktu. Faktor yang berpengaruh pada suatu jaringan komputer terdiri dari beberapa hal seperti dibawah ini (Kalita dan Rane, 2008):

- Waktu transfer yang dihasilkan untuk mengirimkan paket data dari satu *endnode* ke *endnode* lainnya
- Kemungkinan terjadinya paket yang hilang ketika transmisi
- Ukuran *bandwidth* yang tersedia pada suatu *link*
- Kemungkinan terjadinya proses *disconnect* ketika proses transmisi sedang berlangsung

Faktor-faktor tersebut dapat berpengaruh terhadap keseluruhan nilai waktu transfer dan *throughput* yang dihasilkan oleh suatu aplikasi. Sehingga diperlukan pengujian performansi aplikasi khusunya aplikasi *file sharing* terhadap pengaruh dari berbagai macam kondisi jaringan komputer melalui mekanisme emulasi.

Gambar 2.6 Tampilan GUI pada WANem

WANem didukung dengan tampilan *Graphical User Interface* (GUI) untuk memudahkan pengguna dalam melakukan *emulasi* kondisi lingkungan jaringan komputer yang dibutuhkan. **Gambar 2.6** menunjukkan tampilan GUI dari WANem. WANem memiliki beberapa fitur yang dapat digunakan untuk membantu melakukan proses emulasi kondisi lingkungan jaringan komputer, antara lain (Kalita dan Rane, 2008):

- WANalyzer: WANalyzer merupakan fitur yang digunakan untuk melakukan kalibrasi terhadap karakteristik *Wide Area Network*. WANalyzer melakukan proses pengukuran terhadap *bandwitdh* yang tersedia, nilai *delay*, *packet loss* dan *jitter* pada suatu jaringan komputer. Kemampuan dari suatu *host* diukur dengan cara memasukkan *IP address* dari *remote host* pada WANalyzer. Kemudian WANalyzer akan menampilkan hasil pengukuran karakteristik dari *host* tersebut secara akurat.
- *Basic Mode*: *Basic Mode* merupakan fitur pada WANem yang memungkinkan pengguna untuk melakukan emulasi kondisi jaringan komputer hanya pada karakteristik umum seperti *bandwidth* dan *delay* di setiap *network interface* yang terhubung pada WANem.

- *Advanced Mode*: *Advanced Mode* merupakan fitur WANem yang digunakan untuk melakukan emulasi kondisi jaringan dengan berbagai macam karakteritik yang lebih spesifik pada suatu jaringan komputer. Karakteristik yang terdapat pada *Advanced Mode* dapat digunakan pada lebih dari satu *network interface* yang terhubung pada WANem.

Selain menggunakan GUI, WANem juga mendukung operasi melalui *console command* dan dapat diakses melalui *remote host* menggunakan aplikasi *putty* atau aplikasi *Secure Shel* (SSH) lainnya.

2.9 Virtualbox

Virtualbox adalah aplikasi berbasis *cross-platform* yang mendukung virtualisasi terhadap berbagai macam sistem operasi. Virtualbox memungkinkan proses virtualisasi terhadap satu atau lebih sistem operasi secara bersamaan dan saling terhubung satu dengan lainnya (Oracle Corporation, 2015). Sistem operasi yang divirtualisasikan menggunakan Virtualbox disebut dengan *virtual machine*. Jumlah *virtual machine* yang dapat divirtualisasikan melalui Virtualbox dibatasi dengan kapasitas *disk space* dan memori dari komputer *host* yang digunakan.

Di dalam virtualisasi khususnya menggunakan aplikasi Virtualbox, digunakan istilah-istilah yang digunakan untuk menjelaskan bagian-bagian dari kerja sistem, antara lain (Oracle Corporation, 2015):

- *Host operating system* (*host OS*). *Host operating system* atau *host OS* adalah sistem operasi yang digunakan pada *physical computer* dimana aplikasi Virtualbox tersebut diinstal. Virtualbox mendukung *host OS* seperti Windows, Mac OS X, Linux dan Solaris.
- *Guest operating system* (*guest OS*). *Guest operating system* atau *guest OS* adalah sistem operasi yang dijalankan atau divirtualisasikan menggunakan Virtualbox. Virtualbox mendukung *guest operating system* berbasis DOS, Windows, OS/2, FreeBSD dan OpenBSD.
- *Virtual machine* (VM). *Virtual machine* adalah lingkungan virtualisasi yang dibuat didalam Virtualbox dimana *guest operating system* dijalankan. Virtualbox menyediakan beberapa parameter untuk menentukan spesifikasi dari *virtual machine* yang akan dibuat. Spesifikasi tersebut terdiri dari pengaturan kapasitas *hardware*, *CD installer* dan *state information* yang digunakan.
- *Guest additions. Guest additions* merupakan *software* khusus yang terpisah dengan Virtualbox dan dapat ditambahkan untuk meningkatkan performansi dan fitur tambahan pada *guest OS*.

2.10 Wireshark

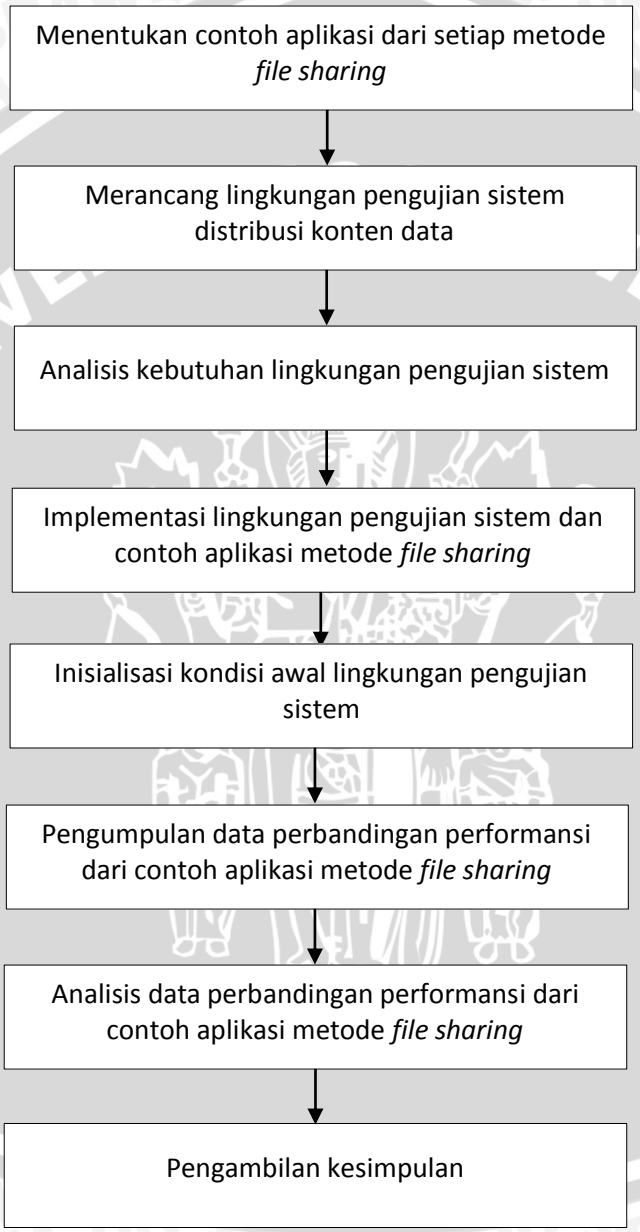
Wireshark adalah aplikasi yang berfungsi untuk menampilkan dan melakukan analisis terhadap paket data yang berjalan pada suatu jaringan komputer. Secara umum, Wireshark digunakan dalam berbagai aktifitas yang berhubungan dengan jaringan komputer seperti *troubleshooting*, melakukan pemeriksaan terhadap

permasalah *security*, melakukan *debug* terhadap implementasi suatu protokol, dan mempelajari cara kerja protokol. Wireshark merupakan *open source software* berbasis *GNU General Public License* (GPL). Wireshark dapat dengan bebas didistribusikan secara gratis tanpa menggunakan *license key*. Selain itu, juga disediakan *source code* berbasis GPL untuk menunjang pengembangan lebih lanjut bagi *developer*.

Salah satu fitur yang disediakan oleh Wireshark yaitu analisis terhadap paket data berdasarkan protokol *transport* yang digunakan yaitu TCP atau UDP. Pada fitur ini akan ditampilkan hasil *capture* terhadap trafik data yang keluar dan masuk pada suatu *interface network*. Hasil yang ditampilkan berupa informasi detail dari aliran paket data yang berjalan pada jaringan melalui suatu *network interface* yang terdiri dari *source IP address*, *source port*, *destination IP address*, *destination port*, jumlah paket yang keluar, jumlah paket data yang masuk, *download rate*, *upload rate*, waktu transfer dan lain sebagainya. Dengan menggunakan fitur tersebut, pengguna dapat dengan mudah melakukan analisis terhadap kondisi trafik yang berjalan pada suatu *network interface* ketika melakukan proses transmisi dengan *interface* lainnya di dalam jaringan komputer.

BAB 3 METODOLOGI PENELITIAN

Dalam bab ini dijelaskan mengenai prosedur yang digunakan untuk melakukan analisis perbandingan performansi terhadap metode *file sharing* berbasis *multicast* dengan *Peer-to-Peer* dalam proses distribusi konten data. Prosedur yang digunakan sesuai dengan alur diagram pada **Gambar 3.1**. Untuk dapat mengetahui



Gambar 3.1 Diagram alur metode penelitian

performansi dari masing-masing metode *file sharing* dalam proses distribusi konten data, langkah pertama yaitu dengan menentukan contoh aplikasi *file sharing* yang menerapkan metode *multicast* dan *Peer-to-Peer*. Penulis menggunakan aplikasi UFTP sebagai contoh metode *file sharing* berbasis *multicast*

dan BitTorrent Sync sebagai contoh metode *file sharing* berbasis *Peer-to-Peer*. Langkah berikutnya, yaitu melakukan perancangan lingkungan pengujian sistem untuk menunjang proses distribusi konten serta perbandingan terhadap performansi pada masing-masing contoh aplikasi metode *file sharing*. Untuk dapat mengimplementasikan rancangan lingkungan pengujian sistem distribusi konten data, diperlukan analisis kebutuhan terkait apa saja yang dibutuhkan dalam membangun lingkungan pengujian sistem tersebut. Setelah lingkungan pengujian sistem distribusi konten data berhasil diimplementasikan, langkah selanjutnya yaitu melakukan inisialisasi dari parameter-parameter kondisi lingkungan pengujian sistem. Hal ini dilakukan agar data performansi yang diperoleh ketika proses pengumpulan data akurat. Selain itu, langkah ini juga digunakan untuk memastikan bahwa kondisi awal lingkungan *(default)* tidak mempengaruhi performansi dari aplikasi *file sharing* sebeum dilakukan mekanisme emulasi. Langkah berikutnya yaitu masing-masing contoh aplikasi metode *file sharing* dijalankan sesuai dengan skenario pengujian yang digunakan untuk dapat memperoleh hasil perbandingan performansi. Dari hasil perbandingan performansi tersebut dilakukan analisis untuk mengetahui pengaruh dari skenario pengujian terhadap performansi dari masing-masing aplikasi *file sharing*. Setelah hasil analisis perbandingan performansi diperoleh, dapat ditarik kesimpulan terkait dengan metode *file sharing* mana yang lebih sesuai dalam melakukan proses distribusi konten data pada lingkungan pengujian sistem jaringan komputer lokal.

3.1 Studi Literatur

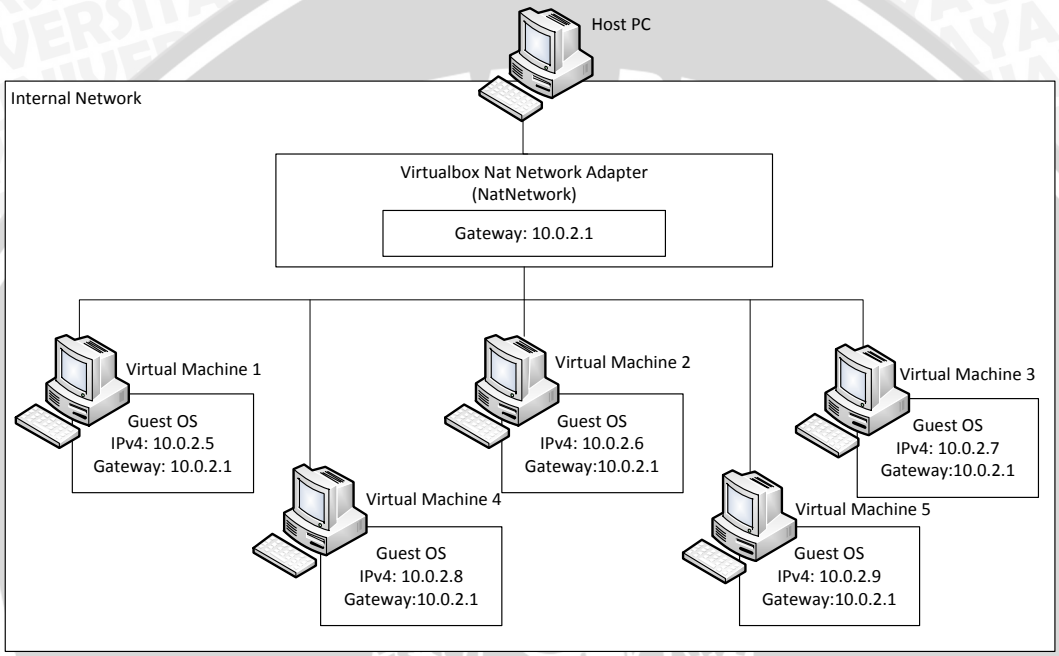
Studi literatur membahas mengenai dasar teori yang digunakan dalam penelitan dan penulisan tugas akhir ini. Dasar teori diperoleh dari penelitian relevan yang dilakukan sebelumnya dan terkait dengan penelitian yang dilakukan oleh penulis, seperti dari jurnal, *e-book*, dan situs *web* resmi yang dapat dipertanggungjawabkan. Dasar teori yang digunakan untuk menunjang penelitian ini antara lain *file sharing*, *multicast*, *Peer-to-Peer*, UFTP, *TCP-Friendly Multicast Congestion Control* (TFMCC), *Distributed Hash Table* (DHT), BitTorrent Sync dan *Wide Area Network emulator* (WANem).

3.2 Rancangan Lingkungan Pengujian Sistem

Dalam tahapan ini dijelaskan mengenai rancangan dari lingkungan pengujian sistem yang digunakan dalam penelitian untuk membantu proses perbandingan performansi dari metode *multicast* dengan *Peer-to-Peer* yang diuji berdasarkan contoh aplikasi UFTP dan BitTorrent Sync. Perancangan dari lingkungan pengujian sistem melibatkan sistem kerja dari aplikasi Virtualbox, *Wide Area Network emulator* (WANem) dan Wireshark. Rancangan lingkungan pengujian sistem yang digunakan yaitu model simulasi dari proses distribusi konten data pada jaringan komputer lokal yang divirtualisasikan.

Untuk merancang suatu sistem lingkungan jaringan komputer lokal secara virtual, maka digunakan aplikasi Virtualbox dengan memanfaatkan model jaringan

NAT Network. Model jaringan *NAT Network* merupakan fitur pemodelan jaringan terbaru yang terdapat pada Virtualbox, dimana semua *virtual machine* dalam *host* yang sama dapat saling berkomunikasi antara satu dengan lainnya, sekaligus dapat terhubung dengan jaringan internet dari luar melalui *host* yang berperan sebagai *home router* dari jaringan internal (Oracle Corporation, 2015). Berdasarkan fungsi dari model *NAT Network* tersebut, maka setiap *virtual machine* mampu memiliki *IP address* berbeda dalam satu *subnet* jaringan yang sama. Setiap *virtual machine* saling berkomunikasi melalui *NAT Network service* yang berperan sebagai *home router* sekaligus sebagai *switch*. Deskripsi dari sistem kerja model *NAT Network* pada Virtualbox dijelaskan pada **Gambar 3.2** dibawah ini.



Gambar 3.2 Sistem kerja model *NAT Network* pada Virtualbox

Berdasarkan **Gambar 3.2** tersebut, antar *virtual machine* dapat saling berkomunikasi dengan atau tanpa terhubung dengan internet dari jaringan eksternal karena terdapat *NatNetwork*. Oleh karena itu virtualisasi jaringan komputer melalui mode *NAT Network* mampu menggambarkan arsitektur komunikasi pada jaringan komputer lokal.

Setelah arsitektur lingkungan pengujian jaringan komputer lokal telah berhasil diimplementasikan secara virtual, selanjutnya dibutuhkan suatu mekanisme emulasi terhadap kondisi lingkungan jaringan komputer tersebut. Salah satu faktor yang mempengaruhi kondisi dari jaringan komputer lokal yaitu tergantung pada perangkat yang digunakan untuk menghubungkan antara *host* dalam *subnet* yang sama dengan jaringan internet (Pingman Tools, 2015). Perangkat-perangkat tersebut antara lain seperti *cable modem*, *DSL modem*, *Dial-up modem* dan *cellular link*. Masing-masing perangkat memiliki rentang nilai *Round Trip Time* (RTT) *delay* atau latensi dan batas toleransi berbeda-beda untuk mendukung proses komunikasi antar *host* dalam satu *subnet* jaringan. **Tabel 3.1** menunjukan rentang

nilai *RTT delay* dari masing-masing perangkat komunikasi yang digunakan dalam jaringan komputer lokal (Pingman Tools, 2015).

Tabel 3.1 Perbandingan nilai *RTT delay* pada perangkat jaringan komputer lokal

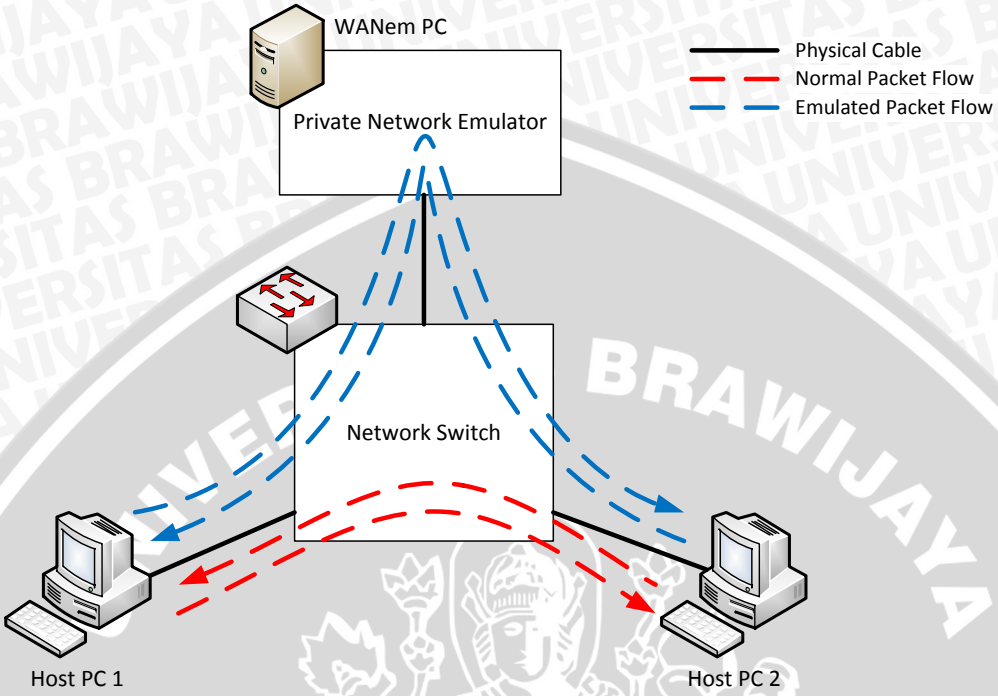
No	Tipe Perangkat	*RTT delay* atau Latensi (ms)
1	*Cable Modem*	5 - 40
2	DSL *Modem*	10 - 70
3	*Dial-up Modem*	100 - 220
4	*Cellular Link*	200 - 600

Dalam penelitian ini, nilai *RTT delay* dari setiap perangkat komunikasi digunakan sebagai variabel bebas untuk menentukan performansi dari aplikasi UFTP dan BitTorrent Sync dalam melakukan proses distribusi konten data. Dalam penelitian ini nilai *RTT delay* yang digunakan dari setiap perangkat diambil dari nilai tengah pada masing-masing rentang nilai toleransi *RTT delay* yang ada.

Dalam suatu mekanisme komunikasi khusunya pada jaringan komputer, kemungkingan terjadi *packet loss* yang muncul ketika proses tranmisi berlangsung hampir tidak dapat dihindari. Termasuk dalam proses distribusi konten data yang memanfaatkan metode *file sharing*. *Packet loss* adalah hilangnya suatu paket data ketika dilakukan transmisi yang disebabkan berbagai macam faktor yang terdapat pada jaringan yang dilalui (Kurose dan Ross, 2000). Hampir seluruh protokol internet memungkinkan terjadi *packet loss* ketika digunakan dalam melakukan proses komunikasi (Pingman Tools, 2015). Meskipun begitu, dalam sistem jaringan komputer nilai *packet loss* yang muncul diharapkan sekecil mungkin atau tidak terjadi sama sekali. Pada umumnya pengaruh dari *packet loss* akan terlihat ketika mencapai nilai 5% dari total keseluruhan paket yang melalui trafik jaringan. Apabila nilai dari *packet loss* diatas dari batas toleransi tersebut, maka akan mempengaruhi kinerja dari proses komunikasi yang berlangsung dan menyebabkan beban trafik yang berjalan semakin padat karena terjadi proses retransmisi terhadap paket yang hilang (Pingman Tools, 2015). Oleh karena itu, dalam penelitian ini, *packet loss* digunakan sebagai variabel bebas pada pengujian performansi aplikasi UFTP dan BitTorrent Sync dengan nilai *packet loss* masing-masing 0% dan 5%.

Untuk dapat menggambarkan kondisi dari variabel bebas yang digunakan pada rancangan lingkungan jaringan komputer lokal seperti *RTT delay* dan *packet loss*, maka digunakan aplikasi *Wide Area Network emulator* (WANem). WANem merupakan emulator jaringan yang didistribusikan melalui sistem operasi Linux Knoppix (Nambiar, 2007). WANem mampu mengemulasikan kondisi lingkungan jaringan komputer sesuai dengan konfigurasi yang dapat disesuaikan secara manual oleh pengguna. Dalam virtualisasi jaringan komputer lokal menggunakan Virtualbox, WANem diimplementasikan pada salah satu *virtual machine* yang ada pada jaringan internal. Agar kondisi jaringan lokal dapat diemulasikan menggunakan aplikasi WANem, maka seluruh alur trafik komunikasi dari setiap

virtual machine yang terkait dengan pengujian distribusi konten data dilewatkan pada *virtual machine* WANem. Pada **Gambar 3.3** dijelaskan mengenai sistem kerja dari WANem.



Gambar 3.3 Sistem kerja WANem

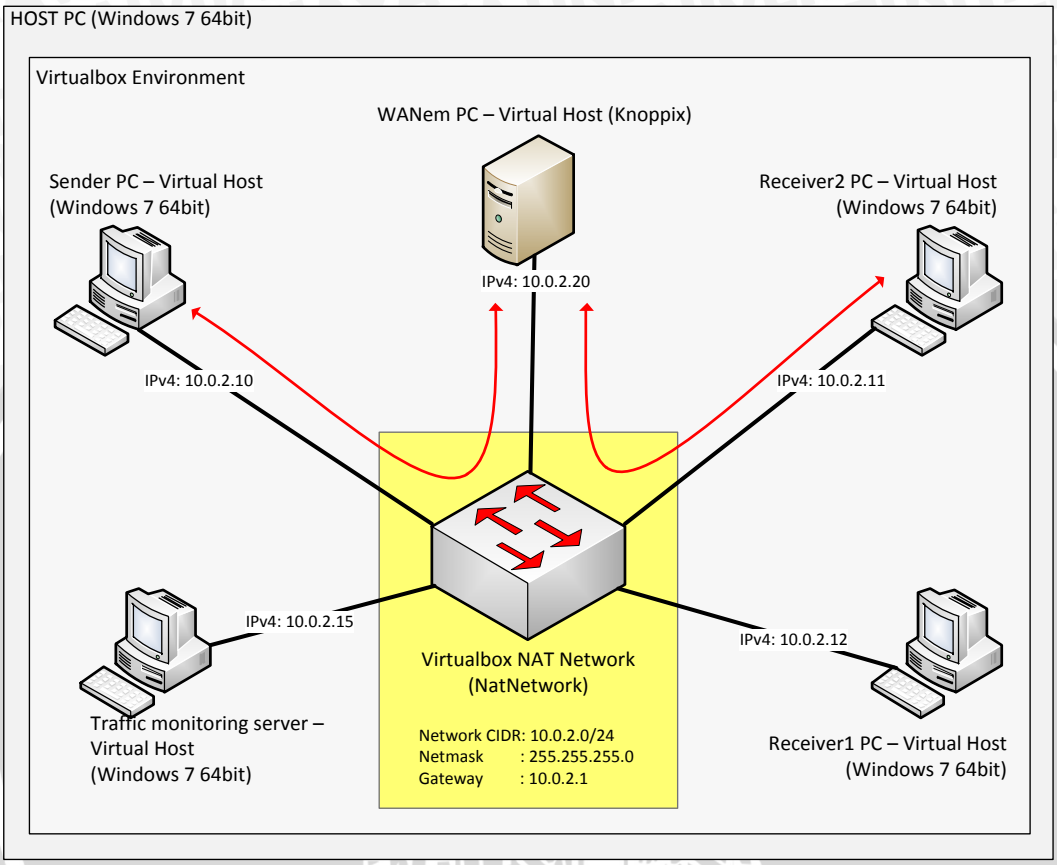
Sumber: Nambiar (2007)

Dari penjelasan pada **Gambar 3.3** dapat diketahui bahwa, alur trafik komunikasi yang berlangsung diubah dari komunikasi langsung antar masing masing *host* diarahkan terlebih dahulu melalui WANem sebelum diteruskan ke *host* tujuan mengikuti garis berwarna biru pada gambar. Hal ini bertujuan agar proses emulasi dapat diterapkan pada lingkungan jaringan komputer yang digunakan.

Dalam proses pengumpulan data perbandingan performansi pada metode *file sharing* dengan contoh aplikasi UFTP dan BitTorrent Sync dibutuhkan analisis terhadap kebutuhan yang digunakan untuk menunjang implementasi rancangan lingkungan pengujian sistem. Jenis kebutuhan yang digunakan antara lain perangkat lunak atau *software* yang terdiri sebagai berikut:

- *Software* pada *physical machine*, yaitu *software* yang berjalan secara langsung pada komputer *host*. *Software* ini terdiri dari sistem operasi Windows 7 Ultimate 64-bit dan Virtualbox versi 5.0.10.
- *Software* pada *virtual machine*, yaitu software yang berjalan pada lingkungan *virtual*. *Software* yang digunakan dalam kategori ini terdiri *installer ISO image* Windows 7 Ultimate 64-bit, *installer ISO image* WANem versi 3.2 pada Linux Knoopix, Wireshark versi 1.12.8, UFTP versi 4.7, BitTorrent Sync versi 2.2.5 dan *browser* Mozilla Firefox versi 41.0.2.

Pada rancangan lingkungan pengujian sistem secara *virtual*, setiap *virtual machine* saling terhubung menggunakan satu *virtual network interface* dimana masing-masing *virtual machine* dikonfigurasi dengan IPv4 secara statis sesuai *subnet* jaringan yang digunakan. **Gambar 3.4** menunjukkan rancangan lingkungan pengujian sistem yang digunakan.



Gambar 3.4 Rancangan lingkungan pengujian sistem

Pada **Gambar 3.4** dapat diketahui bahwa alur trafik jaringan dikonfigurasikan dengan mengubah tabel *routing* dari *virtual host* yang berperan sebagai *receiver 2* untuk melalui WANem PC agar dapat diemulasikan sesuai kondisi jaringan yang digunakan. Perubahan alur trafik ditunjukkan dengan garis berwarna merah pada gambar tersebut. Sedangkan pada *virtual host* yang berperan sebagai *receiver 1* tidak dilakukan perubahan tabel *routing* karena digunakan sebagai faktor pembanding ketika dilakukan skenario mekanisme transmisi dengan satu *receiver* dan dua *receiver*. Begitu juga dengan *virtual host* yang berperan sebagai *traffic monitoring server* tidak dilakukan perubahan *routing* karena digunakan untuk melakukan monitoring trafik yang berjalan melalui Virtualbox *NAT Network* serta berperan sebagai *remote machine* untuk melakukan konfigurasi pada emulator WANem.

3.3 Implementasi Lingkungan Pengujian Sistem

Implementasi yang dilakukan dalam penelitian ini berdasarakan pada integrasi sistem kerja yang telah dijelaskan pada tahap perancangan sebelumnya yaitu antara sistem *NAT Network* pada Virtualbox dengan emulator WANem. Tahapan implementasi lingkungan pengujian sistem pada penelitian ini antara lain sebagai berikut:

1. Implementasi rancangan dan konfigurasi lingkungan pengujian sistem. Pada tahap ini akan diaplikasikan model virtualisasi jaringan komputer lokal pada Virtualbox menggunakan mode *NAT Network*. Lingkungan pengujian sistem terdiri dari lima *virtual machine*, dengan satu *virtual machine* berperan sebagai *sender*, dua *virtual machine* sebagai *receiver*, satu *virtual machine* sebagai emulator WANem PC dan satu *virtual machine* sebagai *traffic monitoring server*. Sistem operasi yang digunakan pada setiap *virtual machine* yaitu Windows 7 64-bit dan khusus pada WANem PC digunakan sistem operasi Linux Knoppix. Konfigurasi pada lingkungan pengujian sistem digunakan untuk memastikan bahwa skenario pengujian dapat berjalan sesuai dengan kondisi pada jaringan komputer lokal secara nyata sesuai dengan perangkat komunikasi yang digunakan.
2. Implementasi aplikasi *file sharing* yang dibandingkan performansinya melalui lingkungan pengujian sistem. Dalam tahapan ini, diimplementasikan aplikasi UFTP dan BitTorrent Sync pada *sender* dan *receiver*. Selain itu juga diimplementasikan aplikasi pendukung lainnya seperti Wireshark pada *traffic monitoring server* untuk melakukan proses monitoring dan analisis perbandingan terhadap trafik jaringan serta performansi dari aplikasi UFTP dan BitTorrent Sync yang keluar masuk dari sisi *sender*. Aplikasi pendukung lain yang digunakan yaitu *browser* Mozilla Firefox pada *sender*, *receiver* dan *traffic monitoring server* yang digunakan untuk menunjang proses *start-up* GUI pada aplikasi BitTorrent Sync dan menampilkan *GUI controller* pada emulator WANem melalui *remote machine*.

3.4 Simulasi dan Pengumpulan Data

Tahapan ini terdiri dari proses simulasi pengujian aplikasi dan pengumpulan data hasil pengujian performansi dari aplikasi yang diuji. Simulasi pengujian yang digunakan mengikuti beberapa aspek antara lain:

1. Jenis variabel. Variabel yang digunakan dalam penelitian ini terdiri sebagai berikut:
   - Variabel tetap: spesifkasi *hardware* dari masing-masing *host*, lingkup pengujian sistem dan sistem operasi yang digunakan.
   - Variabel uji: *throughput* atau *transfer rate*.
   - Variabel pengganti: *RTT delay* atau latensi, ukuran konten data dan *packet loss*.
2. Nilai variabel pengganti. Nilai variabel pengganti yang digunakan dalam penelitian ini adalah sebagai berikut:

- *RTT delay* atau latensi. Menggambarkan nilai *RTT delay* pada jaringan komputer lokal dalam kondisi normal berdasarkan jenis perangkat yang digunakan yaitu 22 ms, 40 ms, 160 ms dan 400 ms.
- Ukuran konten data. Ukuran konten data yang didistribusikan menggunakan contoh aplikasi *file sharing*, menggunakan pola kelipatan tertentu untuk memudahkan analisis perbandingan performansi yang dihasilkan, yaitu 1 MB, 100 MB dan 1 GB.
- *Packet loss*. Menggambarkan kondisi nilai *packet loss* dari kondisi lingkungan jaringan komputer lokal yang dilewati, yaitu 0% dan 5%.

Tabel 3.2 Skenario pengujian performansi aplikasi *file sharing* UFTP dan BitTorrent Sync

Pengujian Ke-	Ukuran Konten Data	*Packet Loss*	*RTT delay* (ms)
Skenario 1 (satu *receiver*): 1, 2, 3, 4, 5, 6, 7, 8	1 MB	0% (1,2,3,4) 5% (5,6,7,8)	22 (1,5) 40 (2,6) 160 (3,7) 400 (4,8)
Skenario 2 (satu *receiver*): 9, 10, 11, 12, 13, 14, 15, 16	100 MB	0% (9,10,11,12) 5% (13,14,15,16)	22 (9,13) 40 (10,14) 160 (11,15) 400 (12,16)
Skenario 3 (satu *receiver*): 17, 18, 19, 20, 21, 22, 23, 24	1 GB	0% (17,18,19,20) 5% (21,22,23,24)	22 (17,21) 40 (18,22) 160 (19,23) 400 (20,24)
Skenario 4 (dua *receiver*): 25, 26, 27, 28, 29, 30, 31, 32	1 GB	0% (25,26,27,28) 5% (29,30,31,32)	22 (25,29) 40 (26,30) 160 (27,31) 400 (28,32)

Tahap berikutnya yaitu tahap pengumpulan data. Tahap ini terdiri dari prosedur perbandingan hasil performansi dari masing-masing aplikasi *file sharing* yang mengikuti skenario yang disesuaikan dengan mekanisme distribusi konten data dan variabel-variabel yang digunakan. **Tabel 3.2** menyajikan skenario yang digunakan dalam penelitian ini. Hasil perbandingan performansi dari aplikasi *file sharing* UFTP dan BitTorrent Sync dalam seluruh skenario berupa nilai *throughput* atau kecepatan transfer data yang berjalan ketika proses distribusi konten data berlangsung. Dengan model skenario pengujian seperti pada **Tabel 3.2**, performansi dari contoh aplikasi metode *file sharing* yang dibandingkan dapat diketahui secara detail mengikuti berbagai macam kombinasi lingkungan pengujian yang ada di dalam kondisi nyata. Dimana konsep dasar yang digunakan yaitu semakin tinggi nilai *throughput* maka semakin baik performansi dari aplikasi *file sharing* dan semakin rendah nilai *throughput* maka semakin buruk performansi

aplikasi *file sharing* (Kurose dan Rose, 2000). Hal tersebut terlepas dari kondisi lingkungan jaringan komputer yang dilalui dalam proses transmisi data.

3.5 Analisis Data Hasil

Pada tahap ini dilakukan analisis terhadap hasil perbandingan performansi dari masing-masing aplikasi *file sharing* sesuai skenario yang diujicobakan. Sehingga dapat diperoleh kesimpulan dari penelitian yang dilakukan. Hasil analisis digunakan untuk mengetahui metode mana yang lebih efektif dan efisien dalam melakukan proses distribusi konten data pada lingkungan jaringan komputer lokal berdasarkan contoh kedua aplikasi *file sharing* yang digunakan.

3.6 Pengambilan Kesimpulan

Tahapan ini merupakan tahapan yang digunakan untuk membuat pernyataan kesimpulan yang didasari dari hasil analisis perbandingan performansi metode *file sharing* berbasis *multicast* dan *Peer-to-Peer* dengan contoh aplikasi UFTP dan BitTorrent Sync. Tahap pengambilan kesimpulan diperoleh apabila seluruh tahapan penelitian telah dilakukan.

BAB 4 IMPLEMENTASI DAN PENGUJIAN

Dalam bab ini dijelaskan mengenai prosedur yang terdiri dari proses implementasi rancangan lingkungan pengujian sistem dan pengujian aplikasi *file sharing*. Pada sub bab implementasi dijabarkan mengenai prosedur implementasi rancangan lingkungan pengujian sistem yang terdiri dari integrasi dan konfigurasi pada aplikasi Virtualbox dan WANem serta aplikasi *file sharing* UFTP dan BitTorrent Sync. Sedangkan pada sub bab pengujian dijelaskan mengenai prosedur pengujian dan perbandingan performansi pada aplikasi *file sharing* UFTP dan BitTorrent Sync.

4.1 Implementasi

Langkah implementasi pada penelitian ini terdiri dari implementasi rancangan lingkungan pengujian sistem dan implementasi aplikasi *file sharing*. Lingkungan pengujian sistem terdiri dari aplikasi Virtualbox, WANem dan Wireshark. Sedangkan aplikasi *file sharing* yang diuji yaitu UFTP dan BitTorrent Sync.

4.1.1 Implementasi Rancangan Lingkungan Pengujian Sistem

Rancangan lingkungan pengujian sistem terdiri dari aplikasi yang digunakan untuk mendukung simulasi terhadap pengujian aplikasi *file sharing*. Komponen yang digunakan untuk melakukan simulasi terhadap kondisi lingkungan pengujian sistem yaitu sebagai berikut:

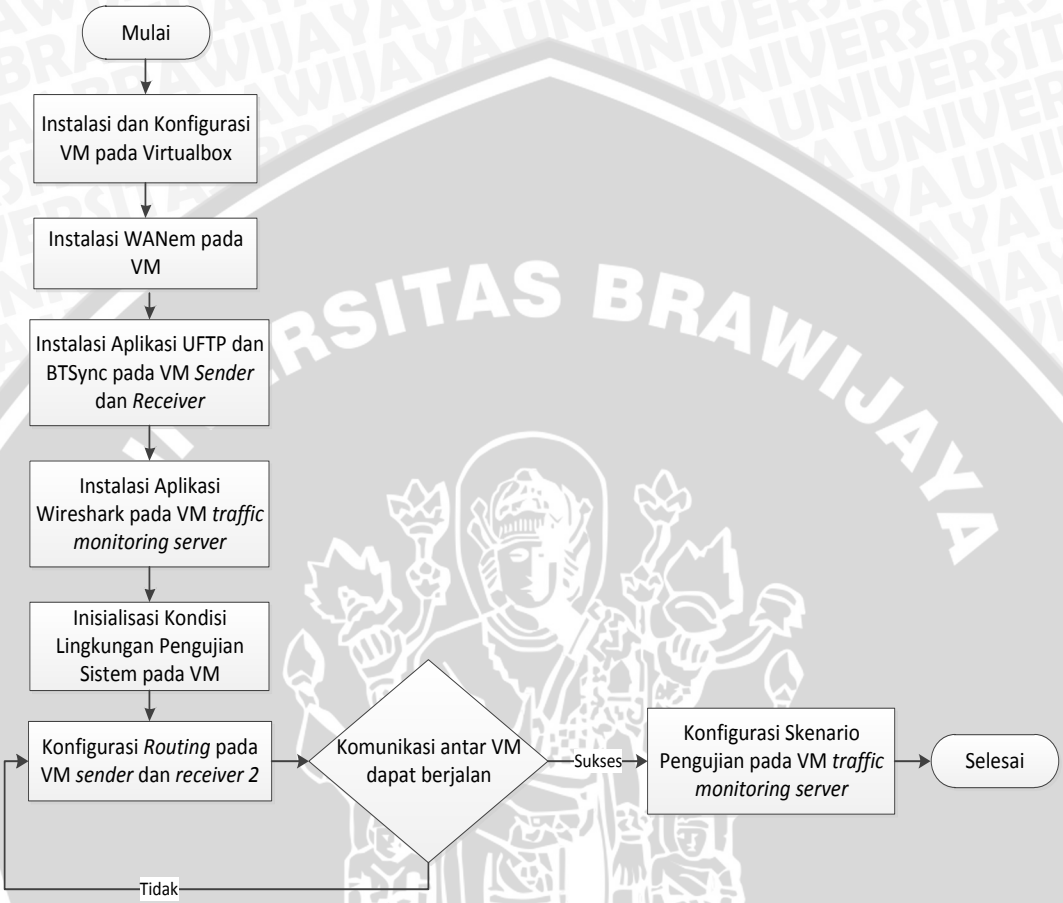
- Virtualbox. Virtualbox adalah aplikasi yang berfungsi untuk melakukan proses virtualisasi terhadap sistem operasi dan jaringan komunikasi antar *virtual machine* yang dibuat. Dalam penelitian ini aplikasi Virtualbox digunakan untuk melakukan simulasi arsitektur jaringan komputer lokal dalam satu *subnet* yang terdiri dari beberapa *virtual machine*. *Virtual machine* yang dibuat pada Virtualbox terdiri dari satu *virtual machine* yang berperan sebagai *sender*, dua *virtual machine* yang berperan sebagai *receiver*, satu *virtual machine* yang berperan sebagai emulator jaringan (WANem) dan satu *virtual machine* yang berperan sebagai *traffic monitoring server* serta untuk melakukan kontrol GUI pada emulator jaringan. *Virtual machine* yang berperan sebagai *sender*, *receiver* dan *traffic monitoring server* menggunakan sistem operasi Windows 7 Ultimate 64-bit. Sedangkan untuk *virtual machine* yang digunakan untuk emulator menggunakan sistem operasi Linux Knoppix. Untuk dapat melakukan simulasi arsitektur jaringan komputer lokal dalam satu *subnet*, mode *network adapter* yang digunakan pada Virtualbox adalah *NAT Network*. Dengan mode *NAT Network* seluruh *virtual machine* dapat saling berkomunikasi satu dengan lainnya seolah-olah berada pada satu *subnet* jaringan yang sama. Mode ini juga memungkinkan komunikasi antar *virtual machine* dapat berjalan dengan atau tanpa adanya koneksi internet dari luar *virtual environment*.

- WANem. WANem adalah aplikasi emulator jaringan berbasis *freeware* yang digunakan untuk mengemulasikan kondisi riil yang terjadi pada suatu jaringan komputer. Dalam penelitian ini, WANem digunakan untuk mengemulasikan berbagai macam kondisi lingkungan jaringan komputer lokal. Pada umumnya, kondisi pada lingkungan jaringan komputer lokal dipengaruhi oleh perangkat yang digunakan untuk menghubungkan seluruh *endnode* pada jaringan tersebut ke internet.
- Wireshark. Wireshark adalah aplikasi berbasis *open-source* yang berfungsi untuk melakukan proses analisis secara detail terhadap paket data yang berjalan melalui jaringan komputer. Dalam penelitian ini, aplikasi Wireshark digunakan untuk melakukan analisis terhadap trafik yang berjalan ketika proses distribusi konten data pada aplikasi *file sharing* sedang berlangsung. Parameter yang digunakan untuk melakukan analisis terhadap performansi aplikasi *file sharing* UFTP dan BitTorrent Sync didasarkan pada protokol *transport* yang digunakan pada masing-masing aplikasi tersebut dalam mendistribusikan konten data yaitu UDP pada *multicast* dan TCP pada *Peer-to-Peer*.

Integrasi dari setiap aplikasi yang dibangun sebagai lingkungan pengujian sistem mengikuti *flowchart* atau diagram alir yang digambarkan pada **Gambar 4.1**. Langkah pertama dalam implementasi lingkungan pengujian sistem diawali dengan instalasi dan konfigurasi empat *virtual machine* (VM) menggunakan sistem operasi Windows 7 Ultimate 64-bit di Virtualbox serta mode jaringan *NAT Network* pada masing-masing *virtual machine*. Kemudian langkah selanjutnya, melakukan proses instalasi dan konfigurasi *virtual machine* yang digunakan sebagai WANem PC menggunakan sistem operasi Linux Knoppix dengan mode jaringan *NAT Network* yang sama dengan *virtual machine* yang telah dibuat sebelumnya. Jika seluruh *virtual machine* yang telah dibuat telah berhasil diimplementasikan, selanjutnya yaitu melakukan instalasi aplikasi UFTP dan BTSync pada *virtual machine* yang berperan sebagai *sender* dan *receiver*.

Untuk melakukan analisis perbandingan terhadap performansi kedua aplikasi *file sharing*, berikutnya dilakukan instalasi aplikasi Wireshark pada *virtual machine* yang berperan sebagai *traffic monitoring server*. Langkah berikutnya yaitu melakukan inisialisasi kondisi lingkungan pengujian sistem untuk memastikan kondisi awal atau *default* dari lingkungan pengujian berjalan dengan normal. Prosedur inisialisasi dilakukan dengan melakukan *ping* antar *virtual machine* untuk mengetahui kondisi *RTT delay* yang dihasilkan dalam lingkungan pengujian sistem ketika dalam kondisi *default*. Kondisi *default* dari lingkungan pengujian sistem diidentifikasikan dengan nilai *RTT delay* kurang dari 1 (<1) pada setiap *virtual machine*. Setelah kondisi awal lingkungan pengujian sistem berjalan dengan normal, langkah berikutnya yaitu melakukan konfigurasi tabel *routing* pada *virtual machine sender* dan salah satu *virtual machine receiver (receiver 2)* agar komunikasi antar *virtual machine* tersebut terlebih dahulu melalui *virtual machine* WANem PC. Untuk mengetahui apakah konfigurasi tabel *routing* berhasil dilakukan, tahap berikutnya yaitu melakukan *testing* menggunakan *tracerout* pada

sender dan *receiver 2*. Jika konfigurasi yang dilakukan tidak berhasil maka kembali dilakukan konfigurasi ulang terhadap tabel *routing*. Jika konfigurasi telah berhasil, langkah berikutnya yaitu melakukan konfigurasi pada *virtual machine guest* yang berperan sebagai *traffic monitoring server* dengan menyesuaikan kondisi skenario jaringan yang akan diemulasikan.



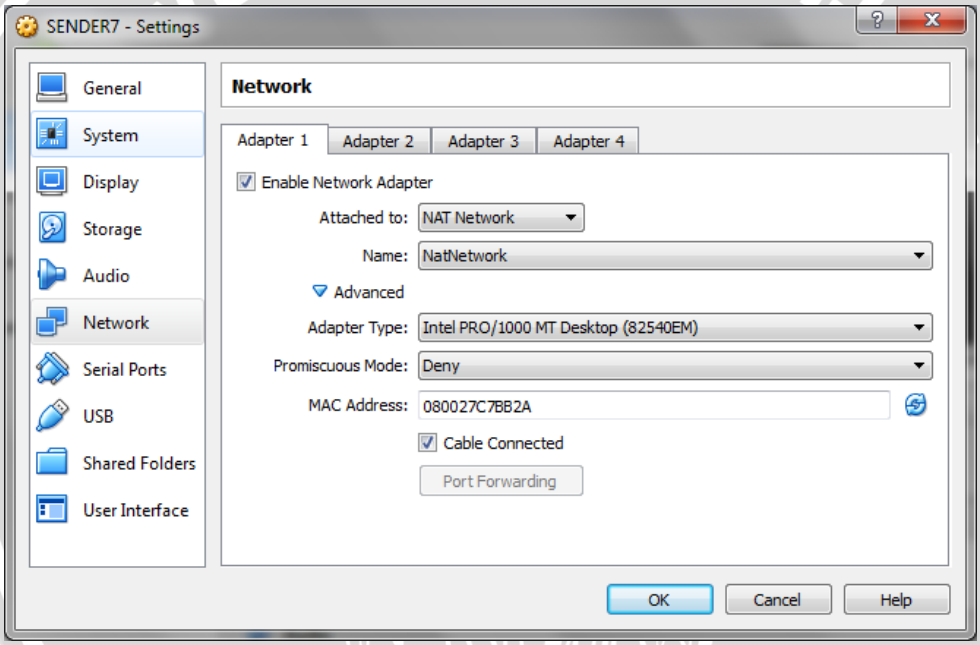
Gambar 4.1 *Flowchart* implementasi rancangan lingkungan pengujian sistem

A. Implementasi *Virtualbox*

Virtualbox digunakan untuk mengimplementasikan simulasi lingkungan jaringan komputer lokal untuk pengujian performansi aplikasi *file sharing*. Virtualbox yang digunakan dalam penelitian ini yaitu Virtualbox versi 5.0.10. Berikut langkah-langkah implementasi dan konfigurasi yang dilakukan pada aplikasi Virtualbox:

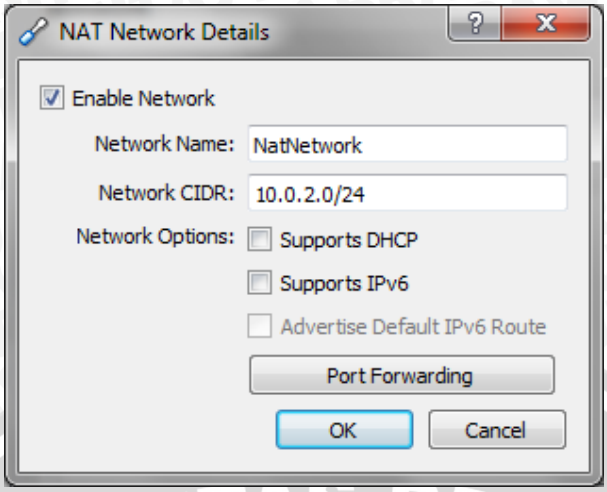
1. Membuat lima *virtual machine* yang terdiri dari empat *virtual machine* dengan sistem operasi Windows 7 Ultimate 64-bit dan satu *virtual machine* dengan sistem operasi Linux Knoppix yang telah terinstal WANem sebelumnya.
2. Setiap *virtual machine* menggunakan alokasi memori sebesar 512 MB dan alokasi *hard disk* sebesar 25 GB untuk menunjang proses yang berjalan pada *virtual machine*.

3. Mode *hard disk* yang digunakan pada setiap *virtual machine* yaitu *Virtual Disk Image* (VDI) dan dialokasikan secara dinamis pada *physical hard disk* pada *host machine*.
4. Pada konfigurasi *network adapter*, masing-masing *virtual machine* menggunakan satu *network interface* dengan mode *NAT Network (NatNetwork)* seperti pada **Gambar 4.2**. Karena *IP address* dari masing-masing *virtual machine* akan dikonfigurasi secara statis maka konfigurasi DHCP pada *NatNetwork* di nonaktifkan dengan menghilangkan *checklist* pada pilihan *Supports DHCP* seperti pada **Gambar 4.3**, agar ketika proses komunikasi berjalan tidak terjadi perubahan *IP address* secara otomatis khususnya pada WANem *virtual machine*. Khusus pada *virtual machine* yang berperan sebagai *traffic monitoring server* pada pilihan *Promiscious Mode* dalam konfigurasi *NAT Network* seperti pada **Gambar 4.2** diubah menjadi *Allow All* agar seluruh trafik jaringan yang berjalan pada lingkungan pengujian sistem dapat dimonitoring melalui *traffic monitoring server*.



Gambar 4.2 Konfigurasi *NAT Network adapter* pada *virtual machine*

5. Melakukan proses instalasi sistem operasi pada masing-masing *virtual machine* sesuai dengan sistem operasi yang telah disebutkan pada langkah pertama.



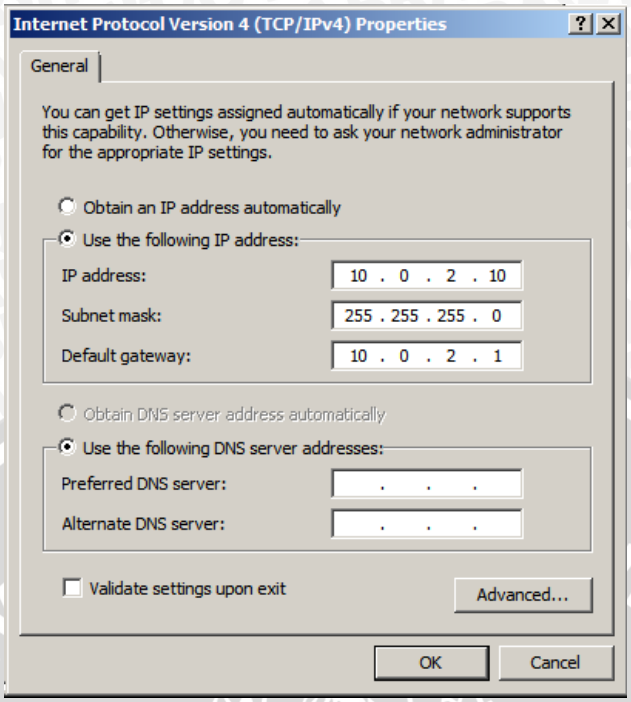
Gambar 4.3 Konfigurasi DHCP *NatNetwork*

6. Pada *virtual machine* yang menggunakan *guest OS* Windows 7 Ultimate 64-bit, dilakukan konfigurasi *IP address* secara statis seperti pada **Gambar 4.4** sesuai pada rancangan lingkungan pengujian sistem. IP statis yang digunakan pada masing-masing *virtual machine* sesuai pada **Tabel 4.1**. Konfigurasi *IP address* masing-masing *virtual machine* menggunakan *subnet mask* 255.255.255.0 dan *default gateway* 10.0.2.1 sesuai dengan konfigurasi pada *NatNetwork adapter* yang digunakan. Khusus pada *virtual machine* yang diimplementasikan emulator WANem, konfigurasi *IP address* statis dilakukan menggunakan prosedur yang berbeda serta dijelaskan pada langkah implementasi selanjutnya.

Tabel 4.1 *IP address* statis pada *virtual machine*

No	*Virtual Machine*	Sistem Operasi	*IP Address* Statis
1	*Sender*	Windows 7 64-bit	10.0.2.10
2	*Receiver* 1	Windows 7 64-bit	10.0.2.11
3	*Receiver* 2	Windows 7 64-bit	10.0.2.12
4	WANem	Linux Knoppix	10.0.2.20
5	*Traffic monitoring server*	Windows 7 64-bit	10.0.2.15

7. Menonaktifkan *service Windows Firewall* untuk memastikan proses komunikasi yang berjalan antar *virtual machine* tidak terganggu oleh sistem keamanan pada masing-masing *guest OS*.



Gambar 4.4 Konfigurasi *IP address* statis pada Windows 7

8. Melakukan testing dan inisialisasi hubungan komunikasi antar *virtual machine* dengan melakukan *ping* terhadap *IP address* setiap *virtual machine* lainnya.

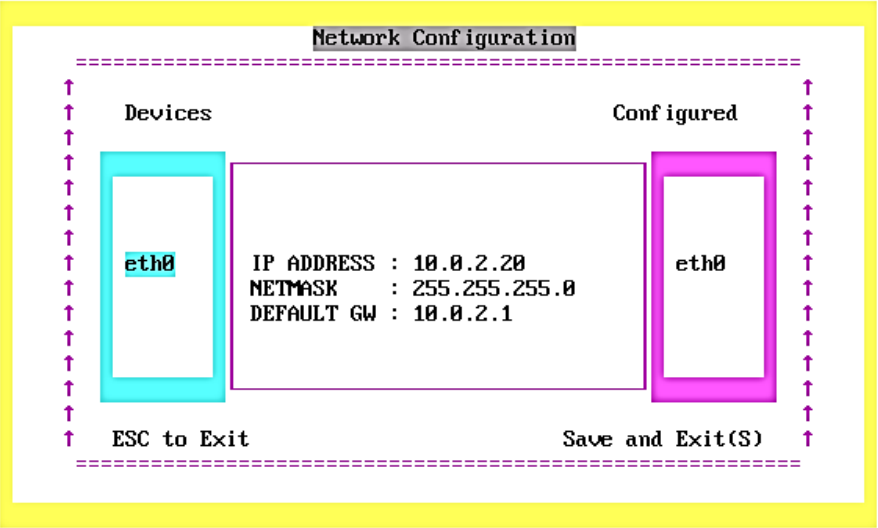
Jika *ping* berhasil dilakukan pada masing-masing *virtual machine* ke *virtual machine* lainnya dan nilai *RTT delay* yang dihasilkan oleh masing-masing *virtual machine* kurang dari 1 ms (<1ms), maka prosedur konfigurasi yang dilakukan telah sesuai dan mekanisme komunikasi antar *virtual machine* berjalan dalam kondisi normal serta siap digunakan untuk mendukung emulasi kondisi lingkungan jaringan komputer lokal.

B. Implementasi WANem

WANem digunakan untuk melakukan emulasi kondisi lingkungan pengujian sistem pada jaringan komputer lokal. Kondisi lingkungan jaringan komputer lokal dipengaruhi oleh kemampuan perangkat yang digunakan sebagai penghubung dari seluruh *endnode* pada jaringan lokal ke jaringan internet (Pingman Tools, 2015). Penelitian ini menggunakan WANem versi 3.2. Pada rancangan lingkungan pengujian sistem WANem diimplementasikan pada salah satu dari lima *virtual machine* yang dibuat pada Virtualbox. Langkah instalasi dan konfigurasi pada WANem dilakukan sebagai berikut:

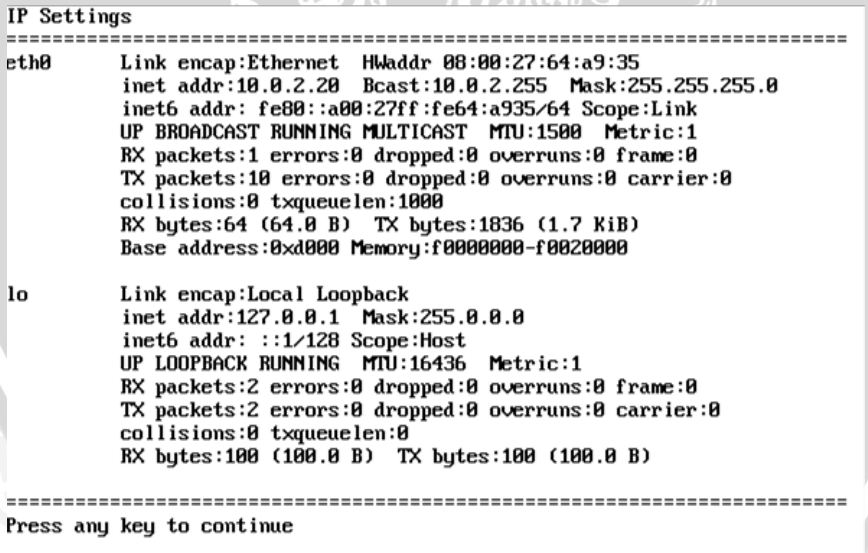
1. Pada *virtual machine* yang telah dibuat sebelumnya, dilakukan proses instalasi dengan menggunakan *file* ISO Linux Knoppix yang sebelumnya telah berisi WANem. Setelah servis DHCP dimatikan, maka dilakukan konfigurasi *IP address* secara statis pada WANem seperti pada **Gambar 4.5**. Kemudian dilakukan konfigurasi *IP address* pada *network interface* WANem secara statis dengan

memasukkan *IP address*, *netmask* atau *subnet mask* dan *default gateway* dari WANem sesuai pada **Tabel 4.1**.



Gambar 4.5 Konfigurasi *IP address* statis pada WANem

2. WANem akan meminta pengguna untuk memasukkan *password* sesuai dengan keinginan untuk memudahkan akses WANem *virtual machine* melalui *remote machine*.
3. Langkah berikutnya yaitu melakukan pengecekan status konfigurasi dengan memasukkan perintah *"status"* pada WANem.



Gambar 4.6 Status konfigurasi *IP address* statis pada WANem

Pengecekkan status akan menampilkan informasi *IP address* dari *network interface* pada WANem *virtual machine* seperti pada **Gambar 4.6**. Konfigurasi berikutnya yaitu melakukan konfigurasi terhadap tabel *routing* pada *virtual machine* yang menggunakan *guest OS* Windows 7 Ultimate 64-bit agar dapat saling berkomunikasi melalui WANem *virtual machine*.

C. Implementasi Wireshark

Wireshark digunakan sebagai aplikasi monitoring terhadap performansi dan aktifitas proses distribusi konten data yang dilakukan oleh masing-masing aplikasi *file sharing*. Wireshark diimplementasikan pada *virtual machine* yang berperan sebagai *traffic monitoring server* untuk melakukan monitoring terhadap trafik yang keluar dan masuk melalui *network interface* yang digunakan oleh setiap *host* dalam melakukan komunikasi dengan *virtual machine* lainnya. Versi Wireshark yang digunakan dalam penelitian ini yaitu versi 1.12.8. Wireshark akan melakukan monitoring terhadap performansi dari setiap aplikasi *file sharing* yang diujikan sesuai dengan protokol *transport* yang digunakan pada masing-masing aplikasi dalam mendistribusikan konten data.

D. Implementasi Tabel *Routing* pada Lingkungan Pengujian Sistem

Agar *virtual machine* dapat diemulasikan melalui WANem, maka dilakukan konfigurasi terhadap tabel *routing* pada *virtual machine* yang berperan sebagai *sender* dan *receiver 2*. Berikut langkah-langkah yang dilakukan untuk melakukan konfigurasi tabel *routing* pada masing-masing *virtual machine*:

1. Konfigurasi tabel *routing* dilakukan pada *virtual machine* yang berperan sebagai *sender* dan *receiver 2* dimana masing-masing menggunakan *guest OS* Windows 7 Ultimate 64-bit.
2. Konfigurasi tabel *routing* pada sistem operasi Windows dilakukan melalui *command prompt* dengan mode *user Administrator* menggunakan perintah "*route add ip_destination mask 255.255.255.255 ip_wanem"* seperti pada **Gambar 4.7**. Pada *virtual machine* yang berperan sebagai *receiver 2* juga dilakukan perubahan tabel *routing* menggunakan perintah yang sama dengan mengubah nilai *ip_destination* yang digunakan.

```
C:\Windows\system32>route add 10.0.2.12 mask 255.255.255.255 10.0.2.20
 OK!
```

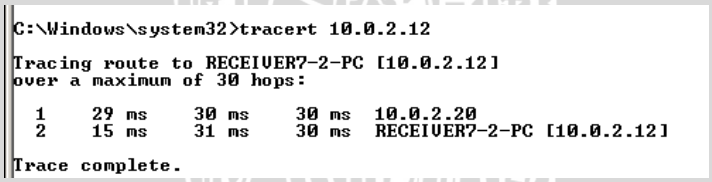
Gambar 4.7 Konfigurasi tabel *routing* pada *virtual machine sender*

3. Konfigurasi tabel *routing* dari *virtual machine* sesuai dengan rancangan lingkungan pengujian sistem yang dijelaskan pada **Tabel 4.2**. *Virtual machine* terlebih dahulu melewati WANem PC sebelum berkomunikasi dengan *virtual machine* lainnya.
4. V*irtual machine* selain *sender* dan *receiver 2* tidak dilakukan konfigurasi tabel *routing* karena *virtual machine*. Sehingga tabel *routing* pada *virtual machine* lainnya tetap dalam kondisi *default*.

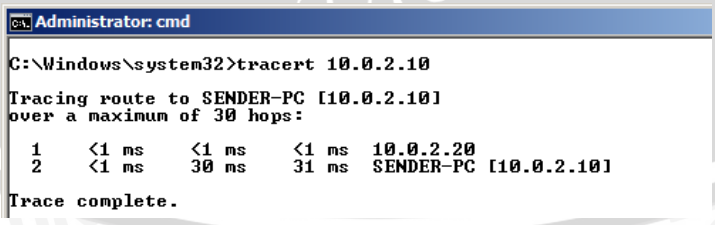
Tabel 4.2 Perubahan tabel *routing* pada lingkungan pengujian sistem

Virtual Machine	*IP Source*	IP WANem	*IP Destination*
Sender	10.0.2.10	10.0.2.20 (*mask* 255.255.255.255)	10.0.2.12
Receiver 2	10.0.2.12	10.0.2.20 (*mask* 255.255.255.255)	10.0.2.10

Setelah seluruh aplikasi yang digunakan untuk membangun lingkungan pengujian sistem telah berhasil diinstal dan konfigurasi tabel *routing* telah berhasil diimplementasikan, maka dilakukan pengujian terhadap alur komunikasi yang berjalan pada lingkungan pengujian sistem untuk mengetahui integrasi dan perubahan tabel *routing* dari masing-masing *virtual machine* menggunakan mekanisme *traceroute*. *Traceroute* dijalankan dengan perintah "*tracert destination_ip_address*" melalui *command prompt* Windows pada setiap *virtual machine* yang berperan sebagai *sender* dan *receiver*. Hasil *traceroute* dari masing-masing *virtual machine* ditunjukkan pada **Gambar 4.8**. Apabila hasil *traceroute* yang ditampilkan telah sesuai dengan konfigurasi tabel *routing* seperti pada **Tabel 4.2**, dimana seluruh komunikasi yang berjalan terlebih dahulu melalui *IP address* 10.0.2.20 pada WANem *virtual machine*, maka konfigurasi lingkungan pengujian sistem berhasil dilakukan dan siap digunakan untuk melakukan pengujian sesuai dengan skenario yang ada. Jika hasil *traceroute* menunjukkan masih terdapat satu atau lebih *virtual machine* yang menghasilkan tabel *routing* yang tidak sesuai dengan rancangan pengujian, maka dilakukan konfigurasi ulang terhadap tabel *routing virtual machine* hingga sesuai dengan rancangan lingkungan pengujian yang digunakan.



(a)



(b)

Gambar 4.8 (a) *Traceroute* pada *sender*; (b) *Traceroute* pada *receiver 2*

4.1.2 Implementasi Aplikasi *File Sharing*

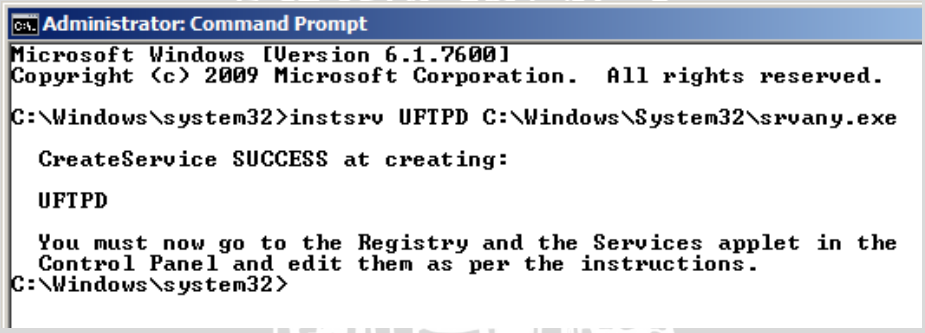
Langkah ini menjelaskan mengenai implementasi aplikasi *file sharing* yang akan dibandingkan dan dianalisis performansinya terhadap kondisi lingkungan

pengujian sistem jaringan komputer lokal yang telah diimplementasikan pada langkah sebelumnya. Aplikasi *file sharing* yang dibandingkan dalam penelitian ini yaitu UFTP dan BitTorrent Sync.

A. Implementasi UFTP

UFTP adalah contoh aplikasi *file sharing* yang digunakan untuk mendistribusikan konten data menggunakan metode *multicast*. UFTP bekerja melalui perintah *command prompt* dan tidak menyediakan tampilan *Graphical User Interface* (GUI) bagi pengguna. UFTP yang digunakan dalam penelitian ini yaitu UFTP versi 4.7 untuk sistem operasi yang berbasis Windows. Untuk melakukan distribusi konten data, UFTP memanfaatkan protokol UDP pada *layer transport*. Langkah-langkah instalasi pada aplikasi UFTP pada sistem operasi Windows 7 Ultimate 64-bit yaitu sebagai berikut:

1. Setelah aplikasi UFTP berhasil dari di*download* melalui website http://uftp-multicast.sourceforge.net dan telah dilakukan ekstraksi terhadap format *file* zip, dilakukan proses *copy* terhadap *file* instsrv.exe dan srvany.exe ke dalam direktori C:\Windows\System32.
2. Melakukan instalasi *service* UFTP melalui perintah pada *command prompt* dengan mode *user Administrator* seperti pada **Gambar 4.9**.



Gambar 4.9 Instalasi *service* UFTP pada *command prompt* Windows

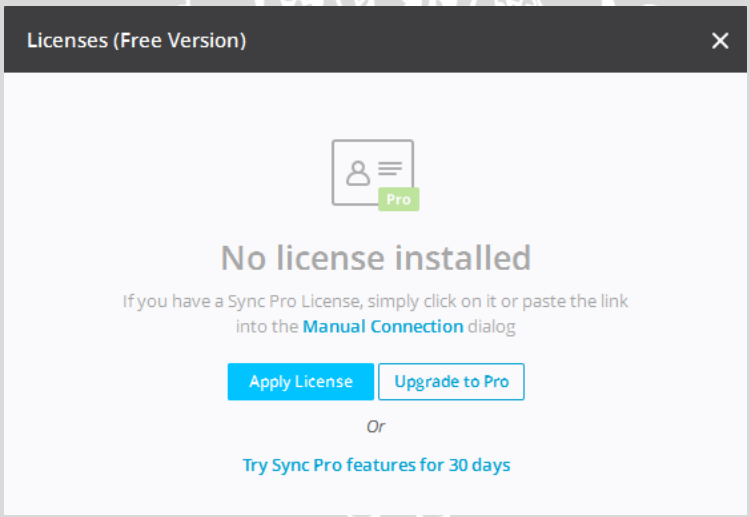
3. Setelah berhasil dilakukan instalasi, dilakukan konfigurasi *regedit* untuk mengarahkan *service* dari aplikasi UFTP agar dapat berjalan pada *command prompt* Windows.
4. Pada *guest OS* yang berperan sebagai *receiver* setelah dilakukan proses instalasi UFTP, langkah berikutnya yaitu membuat *folder* baru secara manual pada direktori C:\ dengan nama *folder "temp"* yang digunakan sebagai direktori tujuan ketika proses distribusi konten data menggunakan aplikasi UFTP berlangsung.

UFTP dibagi menjadi dua macam *service* ketika digunakan dalam proses distribusi konten data pada penelitian ini. *Service* pertama yaitu UFTP yang berperan sebagai *daemon server* dan melakukan distribusi konten data. Sedangkan *service* kedua yaitu UFTPD yang berperan sebagai *daemon client* yang melakukan proses penerimaan terhadap distribusi konten data yang dikirimkan oleh *sender*.

B. Implementasi BitTorrent Sync

BitTorrent Sync adalah aplikasi *file sharing* yang memanfaatkan metode *Peer-to-Peer* dan protokol BitTorrent pada *layer* aplikasi dalam melakukan distribusi konten data. UFTP bekerja menggunakan mekanisme *client application* yang digunakan untuk melakukan proses distribusi konten data. Karena BitTorrent Sync bukan merupakan aplikasi yang berbasis *freeware* dan *open-source*, maka yang digunakan dalam penelitian ini adalah BitTorrent Sync Pro *Trial Version* dengan rentang waktu *trial* selama 30 hari. Langkah instalasi pada BitTorrent Sync adalah sebagai berikut:

1. Setelah aplikasi BitTorrent Sync versi Windows telah di*download* dari situs https://getsync.com, file BitTorrent-Sync.exe dijalankan pada *guest OS*.
2. Konfigurasi aplikasi BitTorrent Sync pada sistem operasi Windows dilakukan melalui GUI yang terintegrasi pada *browser* Internet Explorer versi 9. Karena sistem operasi Windows 7 masih menggunakan Internet Explorer versi 8, konfigurasi BitTorrent Sync dilakukan dengan bantuan *browser* lain yaitu Mozilla Firefox versi 41.0.2 yang digunakan sebagai *default browser*.
3. Melakukan aktivasi *trial* pada BitTorrent Sync Pro *free version* seperti pada **Gambar 4.10** untuk dapat memanfaatkan fitur *link device* yang digunakan untuk mengintegrasikan *client application* pada setiap *virtual machine*.



Gambar 4.10 Fitur *trial* pada BitTorrent Sync *free version*

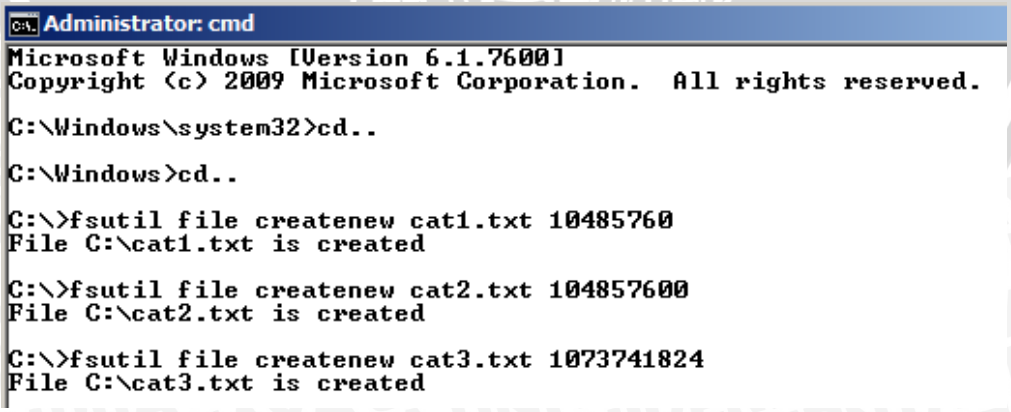
4. Untuk mengintegrasikan *client application* BitTorrent Sync, dilakukan konfigurasi *link device* secara manual menggunakan *copy key* yang di*generate* oleh BitTorrent Sync *client application* pada sisi *sender*. Kemudian *key* tersebut di*input*kan pada masing-masing BitTorrent *client application* yang berada pada sisi *receiver*.
5. Agar proses distribusi konten data dapat berjalan dengan maksimal sesuai dengan kondisi lingkungan pengujian yang digunakan, dilakukan konfigurasi pada beberapa parameter yang terdapat pada aplikasi BitTorrent Sync melalui Menu Preferences, kemudian Advanced. Parameter-parameter yang diubah dari konfigurasi *default* antara lain melakukan di*sable* pada parameter *Use*

relay, *Use tracker*, *Delete to Trash*, *Encryption on LAN, Disk operations are low priority* dan *Enable Debug Profiller*.

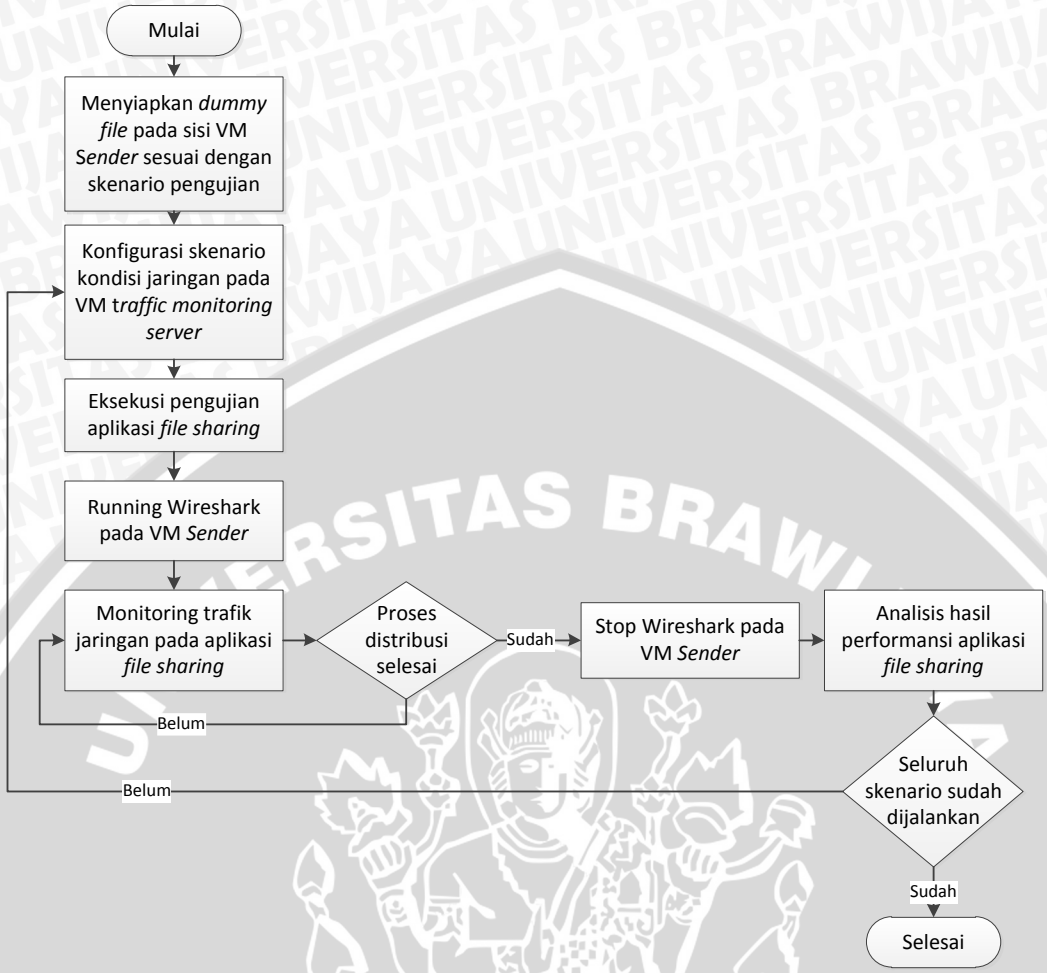
Setelah seluruh *client application* BitTorrent Sync pada masing-masing *virtual machine* telah terintegrasi dan tersinkronisasi satu dengan lainnya, proses distribusi konten data dapat dijalankan melalui aplikasi BiTorrent Sync. *File* yang didistribusikan akan disimpan pada direktori C:\Users\Username\Bit Torrent Sync pada setiap *peer* yang menerima transmisi distribusi konten data.

4.2 Pengujian

Prosedur pengujian dilakukan untuk mengetahui perbandingan performansi dari masing-masing aplikasi *file sharing* dalam melakukan proses distribusi konten data pada lingkungan pengujian sistem jaringan komputer lokal yang digunakan. Pengujian yang dilakukan dalam penelitian ini mengikuti skenario yang telah dideskripsikan dalam Bab III pada masing-masing aplikasi *file sharing*. Pengujian dilakukan sebanyak delapan kali pada setiap aplikasi *file sharing* pada setiap skenario. Dimana terdapat empat skenario pengujian yang digunakan. Performansi masing-masing aplikasi *file sharing* terdiri dari *throughput* yang dianalisis menggunakan bantuan aplikasi Wireshark. Prosedur pengujian terdiri dari konfigurasi pada *traffic monitoring server*, aplikasi *file sharing* dan Wireshark sesuai dengan *flowchart* atau diagram alir pada **Gambar 4.12**. Konfigurasi dilakukan berulang-berulang hingga seluruh skenario telah diujicobakan pada setiap aplikasi *file sharing*. Sebelum dilakukan konfigurasi terhadap aplikasi *file sharing* yang digunakan dalam pengujian, terlebih dahulu dilakukan proses *generate dummy file* dengan ukuran 1 MB, 100 MB dan 1 GB pada *sender virtual machine*. *Dummy file* selanjutnya akan digunakan sebagai sampel ukuran konten data yang didistribusikan melalui aplikasi *file sharing*. *Dummy file* di*generate* menggunakan perintah melalui *command prompt* Windows dengan mode *Administrator* seperti pada **Gambar 4.11**. *Dummy file* yang digunakan sebagai sampel ukuran konten data menggunakan format *file text*.



Gambar 4.11 *Generate dummy file* pada *command prompt* Windows

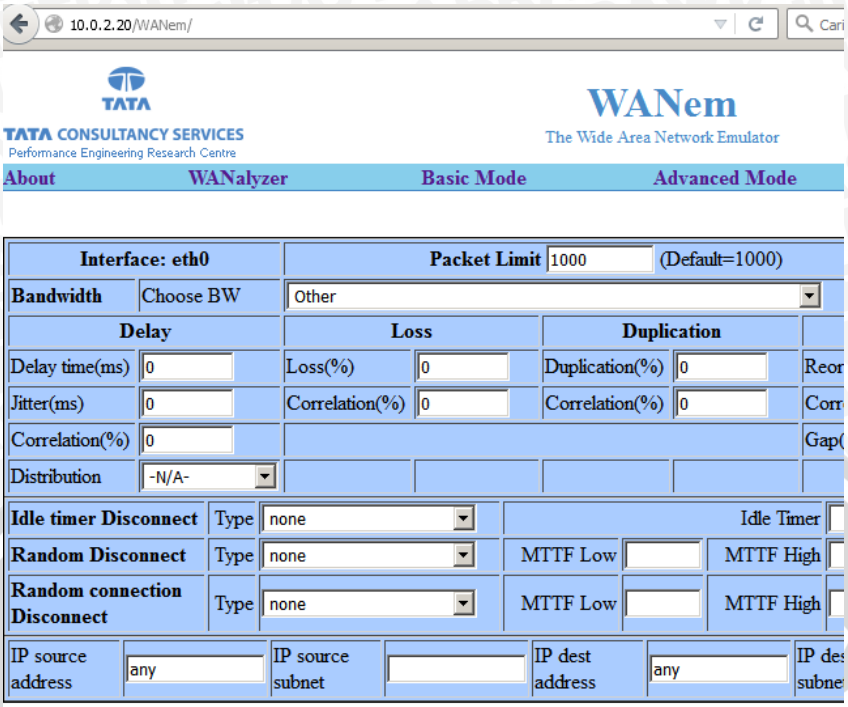


Gambar 4.12 *Flowchart* pengujian aplikasi *file sharing*

4.2.1 Konfigurasi *Traffic Monitoring Server*

Traffic monitoring server digunakan untuk melakukan monitoring terhadap kondisi dan trafik yang berjalan pada lingkungan pengujian sistem. Selain itu juga digunakan untuk melalukan konfigurasi terhadap WANem PC untuk mengemulasikan kondisi jaringan komputer sesuai dengan skenario pengujian yang digunakan melalui tampilan GUI. Langkah yang digunakan dalam konfigurasi pada *traffic monitoring server* untuk melakukan konfigurasi terhadap WANem PC yaitu sebagai berikut:

1. Membuka *browser* pada *traffic monitoring server* kemudian mengakses URL http://10.0.2.20/WANem yang merupakan *IP address* dari WANem PC.
2. Kemudian akan muncul halaman utama *web GUI* dari WANem. Dalam pengujian ini, parameter kondisi lingkungan pengujian yang digunakan yaitu nilai *RTT delay* atau latensi dan *packet loss* yang tersedia pada fitur *Advance mode* seperti pada **Gambar 4.13**.



Gambar 4.13 Konfigurasi parameter *RTT delay* dan *packet loss* pada WANem *Advance Mode*

3. Menyesuaikan *input* pada parameter *delay* dan *packet loss* sesuai dengan skenario pengujian yang digunakan kemudian dilakukan *running* terhadap konfigurasi emulasi WANem.
4. Kondisi lingkungan yang berjalan saat ini telah sesuai dengan skenario pengujian yang digunakan dan telah siap untuk dilakukan pengujian terhadap aplikasi *file sharing*.

4.2.2 Konfigurasi Aplikasi *File Sharing*

Konfigurasi pada masing-masing aplikasi *file sharing* dilakukan secara berulang pada seluruh skenario pengujian yang ada. Pada langkah ini, akan dijabarkan mengenai konfigurasi dari aplikasi UFTP dan BitTorrent Sync dalam proses melakukan distribusi konten data pada lingkungan pengujian sistem. Masing-masing aplikasi memiliki prosedur pengoperasian yang berbeda ketika dilakukan proses distribusi konten data. Konfigurasi dilakukan pada setiap *virtual machine* yang berperan sebagai *sender* dan *receiver*.

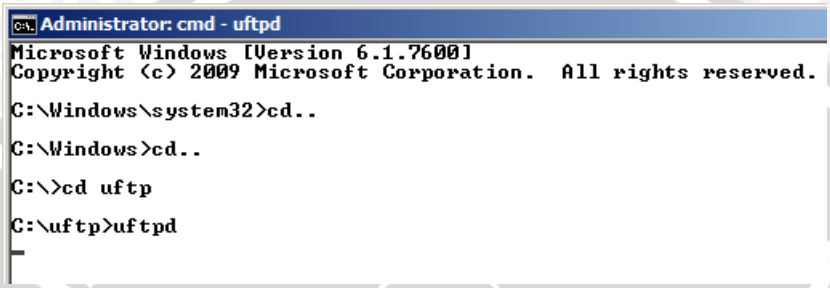
A. Konfigurasi UFTP

Pada penelitian ini, prosedur pengoperasian aplikasi UFTP dalam proses distribusi konten data mengikuti langkah-langkah berikut:

1. Aplikasi UFTP bekerja melalui perintah *command prompt* Windows dengan menggunakan mode *user Administrator*.
2. Sedangkan mode transfer data yang digunakan pada UFTP yaitu mode transfer TFMCC dimana kecepatan *transfer rate* berjalan secara dinamis menyesuaikan kemampuan dari masing-masing *receiver* dalam menerima paket data yang

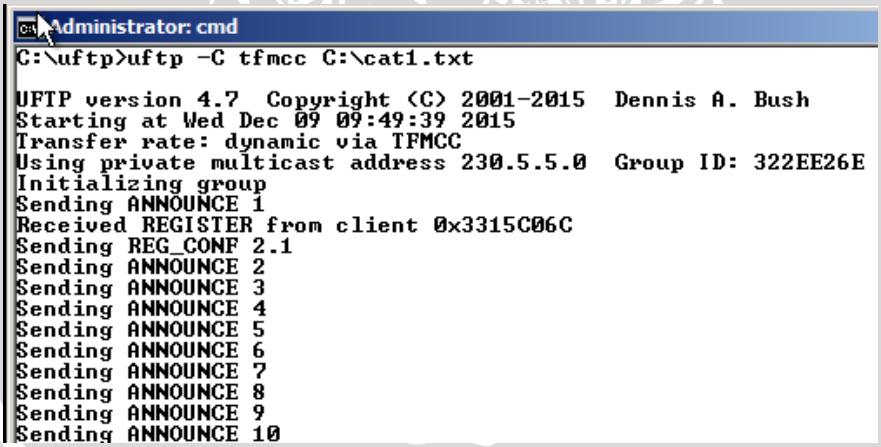
didistribusikan. Secara *default*, kecepatan transfer data pada aplikasi UFTP bersifat statis dan dapat diatur sesuai dengan keinginan pengguna serta kapasitas *link* yang digunakan dalam proses distribusi konten data. Penggunaan mode transfer TFMCC pada penelitian ini karena tidak digunakan parameter kapasitas *link* yang dilalui dalam proses distribusi konten data, sehingga kecepatan *transfer rate* yang digunakan bersifat dinamis.

3. Pada sisi *receiver* dijalankan perintah "*uftpd*" melalui *command prompt* Windows seperti pada **Gambar 4.14** yang digunakan *receiver* untuk menunggu dan memantau jika terdapat aktivitas distribusi konten data oleh *sender*.



Gambar 4.14 Perintah *uftpd* pada *command prompt receiver*

4. Pada sisi *sender*, untuk melakukan proses distribusi konten data menggunakan mode TFMCC, perintah yang di*input*kan melalui *command prompt* yaitu "*uftp –C tfmcc*" seperti pada **Gambar 4.15**.



Gambar 4.15 Perintah *uftp* pada *command prompt sender*

5. UFTP pada sisi *sender* akan melaporkan informasi mengenai proses distribusi konten data yang dilakukan secara *real time*.

B. Konfigurasi BitTorrent Sync

Konfigurasi pada BitTorrent Sync untuk melakukan proses distribusi konten data pada penilitian ini mengikuti langkah-langkah sebagai berikut:

1. Menjalankan BitTorrent *client application* pada sisi *sender* dan *receiver* melalui GUI yang terintegrasi dengan *browser* Mozilla Firefox.
2. Pada sisi *sender*, dilakukan penentuan *file* atau konten data mana yang akan didistribusikan pada *receiver*. *Folder* atau konten data yang

didistribusikan menggunakan tipe *Standar Mode* dimana mekanisme distribusi berdasarkan pada *device-based permissions*.

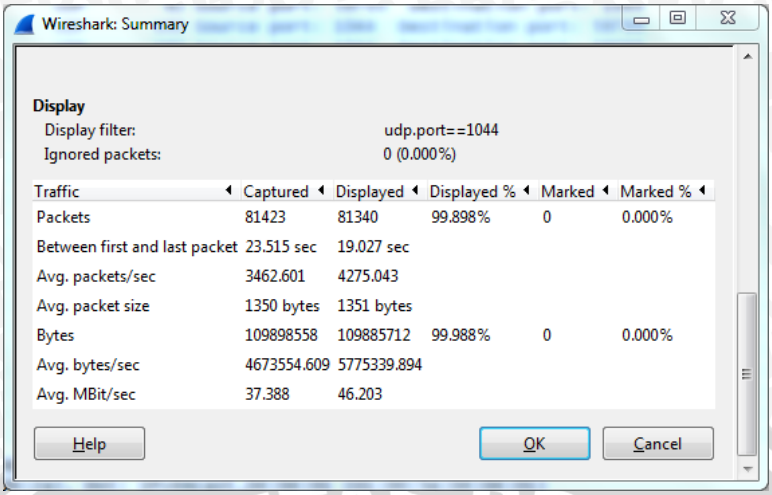
3. Pada pilihan *permission link*, digunakan *read & write permission*, dimana setiap *peer* yang terkait dengan mekanisme distribusi data dapat mengubah isi dari konten data yang didistribusikan.

Proses distribusi konten data selesai jika pada tabel status di *client application sender* maupun *receiver* tidak terdapat aktifitas *sending* atau *receiving*.

4.2.3 Konfigurasi Wireshark

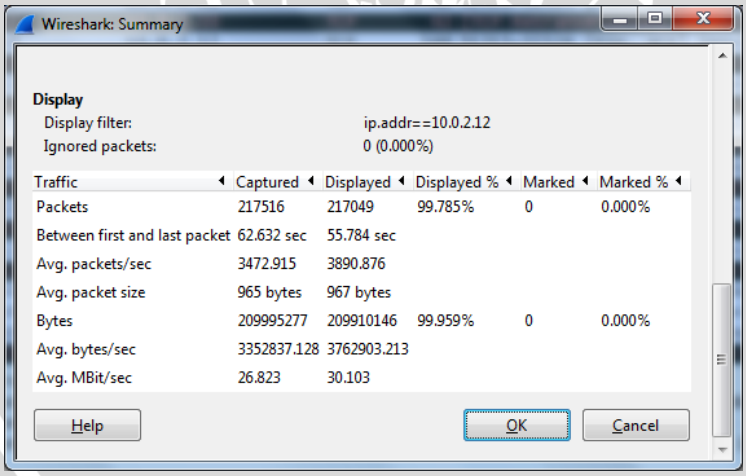
Wireshark digunakan untuk melakukan monitoring terhadap proses distribusi konten data yang berlangsung antara *sender* dan *receiver*. Selain itu Wireshark juga digunakan untuk melakukan analisis perbandingan terhadap performansi dari masing-masing aplikasi *file sharing* berdasarkan hasil monitoring yang dilakukan. Berikut prosedur pengoperasian dari Wireshark yang digunakan dalam penelitian ini:

1. Menjalankan aplikasi Wireshark yang terdapat pada sisi *traffic monitoring server* dan melakukan *capturing* terhadap *network interface* yang digunakan dalam proses distribusi konten data segera setelah aplikasi *file sharing* dijalankan.
2. Melakukan monitoring terhadap trafik data yang berjalan pada lingkungan pengujian sistem, jika proses distribusi konten data selesai, dilakukan penghentian proses *capturing* pada Wireshark.
3. Kemudian dilakukan analisis terhadap hasil *capture* trafik jaringan yang diperoleh untuk mengetahui performansi dari masing-masing aplikasi *file sharing* pada skenario pengujian yang digunakan. Analisis setiap aplikasi *file sharing* berdasarkan protokol *transport layer* yang digunakan untuk melakukan distribusi konten data. UFTP menggunakan protokol *transport layer* UDP sedangkan BitTorrent Sync menggunakan protokol TCP maupun UDP.
4. Untuk melakukan analisis terhadap masing-masing aplikasi *file sharing*, dipilih menu *Statistic*, kemudian *Summary*.
5. Wireshark akan menampilkan hasil monitoring trafik secara detail dari lingkungan pengujian sistem. Dalam penelitian ini parameter yang diambil dari hasil monitoring yaitu *throughput* atau *transfer rate* antara *sender* dan *receiver* dalam satuan MBps.
6. Pada aplikasi UFTP, analisis performansi distribusi konten data diidentifikasikan dengan melakukan *filter* pada hasil *capture* Wireshark dengan *rule "udp.port==1044"*. Port 1044 merupakan *port* UDP *default* yang digunakan pada aplikasi UFTP untuk melakukan transmisi data pada *multicast group*. **Gambar 4.16** menunjukkan hasil *filter* trafik UFTP yang berjalan pada lingkungan pengujian sistem.



Gambar 4.16 Hasil *filter* trafik UFTP pada Wireshark

7. Sedangkan pada aplikasi BitTorrent Sync, performansi distribusi konten data diidentifikasikan dengan melakukan *filter* dengan *rule "ip.addr==10.0.2.12"*. Dalam melakukan distribusi konten data, BitTorrent Sync menggunakan lebih dari satu *port*. Selain itu BitTorrent Sync juga berkomunikasi menggunakan protokol *transport layer* TCP maupun UDP. Oleh karena itu, digunakan parameter *filter IP address* dari *receiver 2* sebagai *rule* untuk mengidentifikasi trafik dari BitTorrent Sync pada lingkungan pengujian sistem. *IP address* 10.0.2.12 merupakan *IP address* dari *receiver 2*. **Gambar 4.17** menunjukkan hasil *filter* trafik BitTorrent Sync yang berjalan pada lingkungan pengujian sistem.



Gambar 4.17 Hasil *filter* trafik BitTorrent Sync pada Wireshark

Jika seluruh skenario yang digunakan telah dilakukan untuk menguji performansi masing-masing aplikasi *file sharing*, maka proses pengujian telah selesai. Apabila masih terdapat skenario pengujian yang belum dilakukan, maka dilakukan pengujian ulang pada skenario tersebut hingga seluruh skenario berhasil dijalankan.

BAB 5 HASIL DAN ANALISIS

Bab ini mendeskripsikan hasil perbandingan performansi dari contoh aplikasi metode *file sharing* pada lingkungan pengujian sistem yang dilakukan pada bab sebelumnya serta analisis terhadap hasil pengujian yang diperoleh. Hasil perbandingan performansi terdiri *throughput* atau *transfer rate* yang dihasilkan oleh masing-masing aplikasi *file sharing*. Sedangkan parameter yang digunakan untuk melakukan perbandingan performansi terdiri dari kombinasi variabel ukuran data, nilai *packet loss* dan *RTT delay*.

5.1 Hasil Perbandingan

Hasil perbandingan performansi dari aplikasi *file sharing* UFTP dan BitTorrent Sync berdasarkan pada parameter *throughput* yang dihasilkan ketika melalui berbagai macam kondisi lingkungan pengujian sistem sesuai dengan skenario pengujian yang digunakan. Kondisi lingkungan pengujian sistem sendiri dipengaruhi oleh skenario pengujian yang terdiri dari kombinasi variabel ukuran konten data, *packet loss* dan *RTT delay*. Sebagai dasar perbandingan performansi, digunakan nilai *RTT delay* sebagai acuan perbandingan variabel lainnya. Penggunaan nilai *RTT delay* sebagai dasar perbandingan performansi karena disesuaikan dengan kondisi *RTT delay* nyata yang dihasilkan oleh perangkat komunikasi pada jaringan komputer lokal.

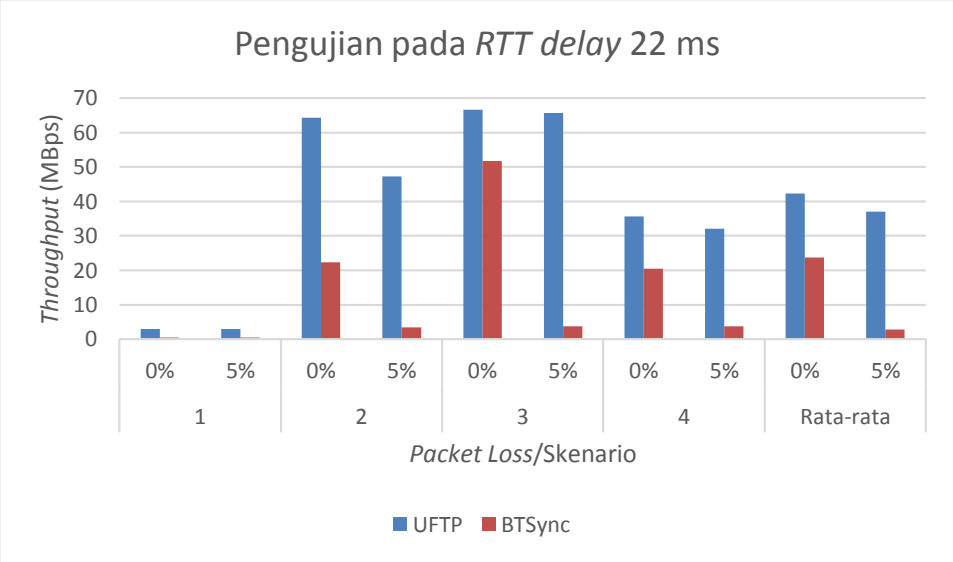
5.1.1 Hasil Perbandingan pada 22 ms *RTT Delay*

Hasil perbandingan performansi dari aplikasi *file sharing* UFTP dan BitTorrent Sync pada seluruh skenario pengujian yang digunakan ketika kondisi *RTT delay* pada lingkungan pengujian sistem sebesar 22 ms ditampilkan pada **Tabel 5.1**. Nilai perbandingan yang dihasilkan dari setiap aplikasi *file sharing* berupa *throughput* yang dihasilkan ketika digunakan melakukan distribusi konten data.

Tabel 5.1 Hasil perbandingan *throughput* UFTP dan BitTorrent Sync pada *RTT delay* 22 ms

Skenario	*Packet Loss*	*Throughput* (ms)	
		UFTP	BTSync
1	0%	2.97	0.52
	5%	2.89	0.45
2	0%	64.32	22.36
	5%	47.23	3.37
3	0%	66.52	51.70
	5%	65.59	3.68
4	0%	35.57	20.43
	5%	32.11	3.79
Rata-rata	0%	42.34	23.75
	5%	36.96	2.82

Pada pengujian dengan kondisi nilai *RTT delay* 22 ms dilakukan percobaan sebanyak delapan kali pada setiap aplikasi *file sharing* sesuai skenario pengujian yang digunakan. Data hasil perbandingan kedua aplikasi *file sharing* pengujian dengan kondisi nilai *RTT delay* 22 ms disajikan dalam grafik yang terdapat di **Gambar 5.1**. Dari grafik yang disajikan pada **Gambar 5.1**, dapat diketahui bahwa performansi *throughput* yang dihasilkan dari setiap aplikasi *file sharing* ketika digunakan untuk mendistribusikan konten data dalam kondisi *RTT delay* 22 ms dipengaruhi oleh kondisi *packet loss*, ukuran konten data yang didistribusikan dan jumlah *receiver*. Performansi *throughput* yang dihasilkan masing-masing aplikasi *file sharing* meningkat secara drastis ketika ukuran konten data yang didistribusikan meningkat dari 1 MB pada skenario ke-1 menuju ke skenario ke-2 dengan ukuran konten data 100 MB. Selanjutnya *throughput* terus meningkat pada skenario ke-3 dengan ukuran konten data 1 GB. Kemudian pada skenario ke-4 dimana dilakukan distribusi konten data dalam ukuran 1 GB pada dua *receiver*, *throughput* yang dihasilkan oleh setiap aplikasi mengalami penurunan. Perubahan kondisi *packet loss* berpengaruh lebih signifikan pada aplikasi BitTorrent Sync dibandingkan pada aplikasi UFTP. Hal ini dibuktikan dengan penurunan nilai *throughput* secara signifikan pada aplikasi BitTorrent Sync ketika nilai *packet loss* meningkat dari 0% ke 5%. Nilai rata-rata *throughput* yang dihasilkan pada kondisi *RTT delay* 22 ms menunjukkan bahwa aplikasi UFTP lebih unggul dibandingkan dengan BitTorrent Sync baik dalam kondisi packet loss 0% dan 5%.



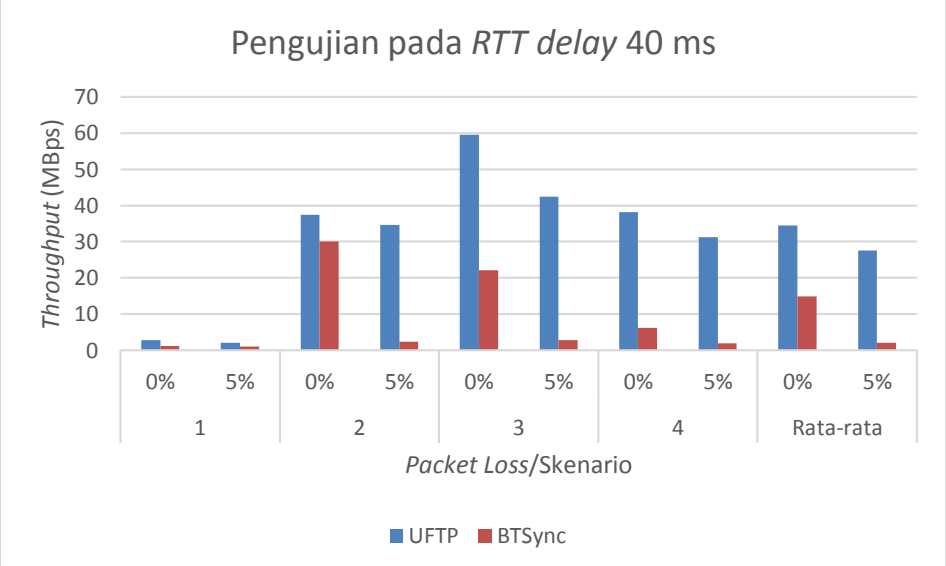
Gambar 5.1 Grafik perbandingan *throughput* UFTP dan BitTorrent Sync pada *RTT delay* 22 ms

Berikutnya pada pengujian ketika kondisi *RTT delay* 40 ms, hasil perbandingan performansi aplikasi *file sharing* UFTP dan BitTorrent Sync terdapat pada **Tabel 5.2**. Pengujian pada kondisi *RTT delay* sebesar 40 ms terdiri dari *throughput* yang dihasilkan oleh setiap aplikasi ketika digunakan untuk mendistribusikan konten data dalam seluruh skenario pengujian yang digunakan. Setiap aplikasi dilakukan pengujian sebanyak delapan kali mengikuti kombinasi ukuran konten data, jumlah *receiver* dan *packet loss* yang digunakan.

Tabel 5.2 Hasil perbandingan *throughput* UFTP dan BitTorrent Sync pada *RTT delay* 40 ms

Skenario	*Packet Loss*	*Throughput* (ms)	
		UFTP	BTSync
1	0%	2.83	1.09
	5%	2.04	1.08
2	0%	37.38	30.10
	5%	34.63	2.4
3	0%	59.56	22.08
	5%	42.5	2.73
4	0%	38.13	6.16
	5%	31.25	1.95
Rata-rata	0%	34.47	14.86
	5%	27.60	2.04

Grafik hasil perbandingan performansi aplikasi UFTP dan BitTorrent Sync pada kondisi nilai *RTT delay* 40 ms disajikan pada **Gambar 5.2**. Dari grafik tersebut jika dibandingkan dengan hasil yang diperoleh ketika nilai *RTT delay* 22 ms mengalami penurunan nilai *throughput* yang dihasilkan oleh masing-masing aplikasi. Meskipun nilai penurunan yang terjadi tidak berpengaruh secara signifikan. Selain itu, dapat diketahui bahwa performansi dari aplikasi UFTP cenderung lebih baik dibandingkan dengan BitTorrent Sync pada kondisi skenario *RTT delay* 40 ms. Hal ini dibuktikan dengan nilai rata-rata *throughput* yang dihasilkan oleh aplikasi UFTP lebih tinggi ketika digunakan untuk mendistribusikan konten data dalam kondisi *packet loss* dan ukuran konten data yang digunakan pada skenario pengujian.



Gambar 5.2 Grafik perbandingan *throughput* UFTP dan BitTorrent Sync pada *RTT delay* 40 ms

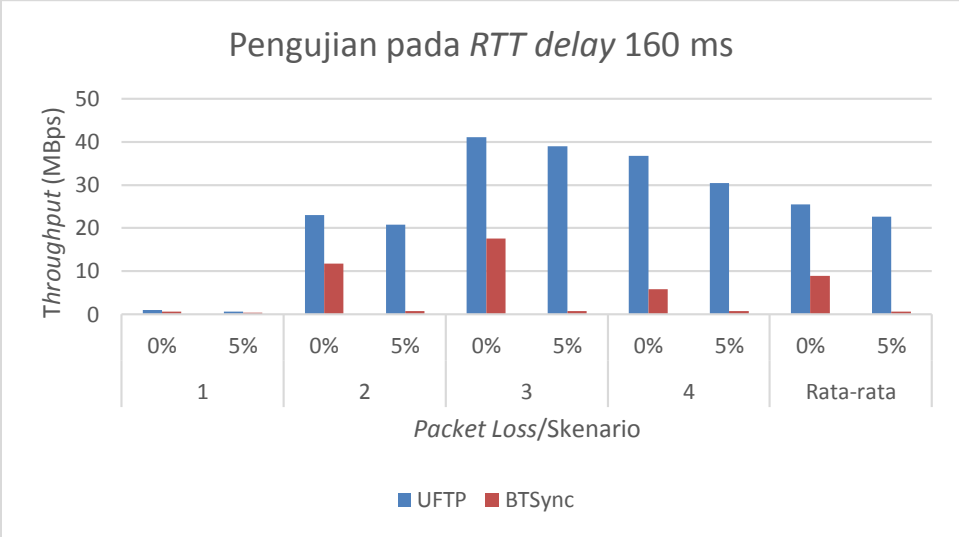
Berikutnya pada pengujian ketika kondisi *RTT delay* 160 ms, hasil perbandingan performansi aplikasi *file sharing* UFTP dan BitTorrent Sync terdapat pada **Tabel**

5.3. Pengujian pada kondisi ini terdiri dari *throughput* yang dihasilkan oleh setiap aplikasi ketika digunakan untuk mendistribusikan konten data dalam seluruh skenario pengujian ketika kondisi *RTT delay* yang berjalan 140 ms. Setiap aplikasi dilakukan pengujian sebanyak delapan kali percobaan.

Tabel 5.3 Hasil perbandingan *throughput* UFTP dan BitTorrent Sync pada *RTT delay* 160 ms

Skenario	*Packet Loss*	*Throughput* (ms)	
		UFTP	BTSync
1	0%	0.97	0.62
	5%	0.54	0.41
2	0%	22.97	11.73
	5%	20.83	0.70
3	0%	41.15	17.54
	5%	39	0.71
4	0%	36.76	5.81
	5%	30.50	0.69
Rata-rata	0%	25.46	8.92
	5%	22.72	0.63

Grafik hasil perbandingan performansi aplikasi UFTP dan BitTorrent Sync pada kondisi *RTT delay* sebesar *160 ms* disajikan pada **Gambar 5.3**. Dari grafik tersebut jika dibandingkan kondisi pengujian *RTT delay* pada pengujian sebelumnya, nilai *throughput* yang dihasilkan mengalami penurunan lebih signifikan. Selain itu, dapat diketahui bahwa performansi dari aplikasi UFTP cenderung lebih baik dibandingkan dengan BitTorrent Sync pada kondisi *RTT delay* 160 ms. Hal ini dibuktikan dengan nilai *throughput* rata-rata pada aplikasi UFTP lebih tinggi dalam kondisi *packet loss* 0% maupun 5% dibandingkan dengan BitTorrent Sync.

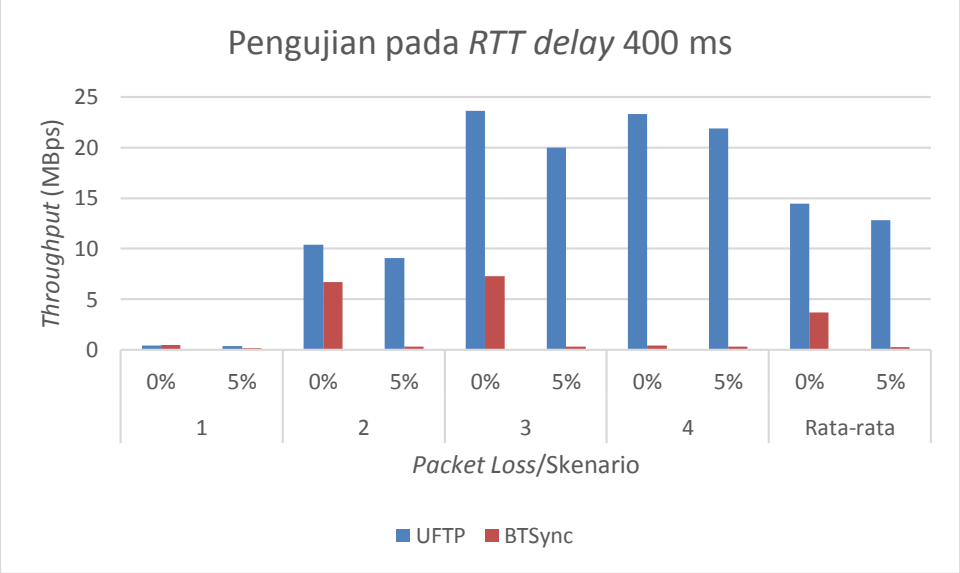


Gambar 5.3 Grafik perbandingan *throughput* UFTP dan BitTorrent Sync pada *RTT delay* 160 ms

Tabel 5.4 Hasil perbandingan *throughput* UFTP dan BitTorrent Sync pada *RTT delay* 400 ms

Skenario	*Packet Loss*	*Throughput* (ms)	
		UFTP	BTSync
1	0%	0.45	0.51
	5%	0.39	0.18
2	0%	10.37	6.71
	5%	9.08	0.32
3	0%	23.61	7.26
	5%	19.96	0.32
4	0%	23.30	0.41
	5%	21.87	0.31
Rata-rata	0%	14.43	3.73
	5%	12.82	0.28

Berikutnya pada pengujian pada *RTT delay* 400 ms, hasil perbandingan performansi aplikasi *file sharing* UFTP dan BitTorrent Sync terdapat pada **Tabel 5.4**. Grafik hasil perbandingan performansi aplikasi UFTP dan BitTorrent Sync pada skenario kondisi *RTT delay* 400 ms disajikan pada **Gambar 5.4**. Dari grafik tersebut dapat diketahui bahwa performansi dari aplikasi UFTP dan BitTorrent Sync semakin menurun dibandingkan kondisi *RTT delay* yang diujikan dari ketiga skenario pengujian sebelumnya. Selain itu performansi aplikasi UFTP jauh lebih baik dibandingkan dengan BitTorrent Sync pada kondisi *RTT delay* 400 ms. Hal ini dibuktikan dengan nilai *throughput* yang dihasilkan oleh aplikasi UFTP lebih tinggi dibandingkan dengan BitTorrent Sync.

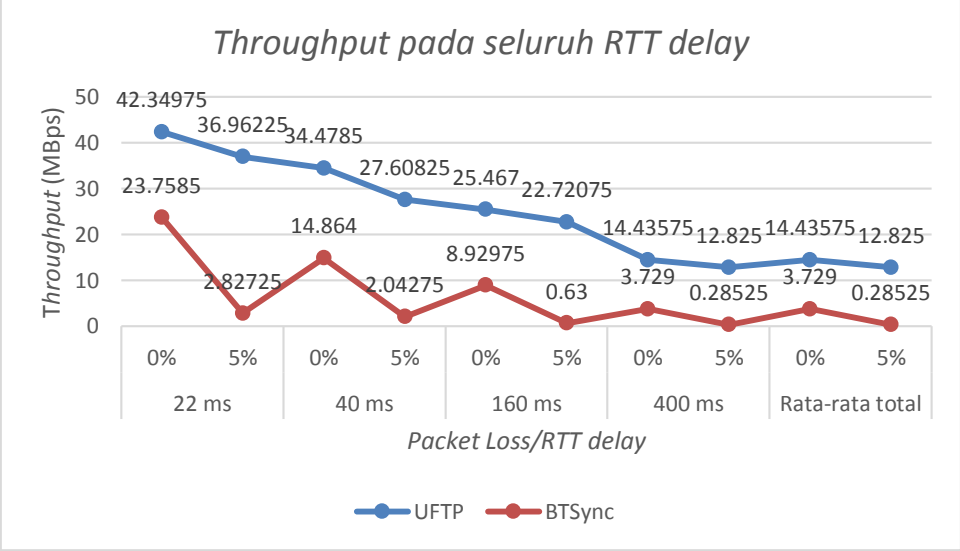


Gambar 5.4 Grafik perbandingan *throughput* UFTP dan BitTorrent Sync pada *RTT delay* 400 ms

Dari hasil perbandingan aplikasi *file sharing* pada seluruh kondisi *RTT delay* pengujian yang digunakan, dapat diketahui bahwa nilai performansi *throughput* yang dihasilkan oleh aplikasi *file sharing* UFTP dengan BitTorrent Sync dipengaruhi oleh nilai parameter *packet loss*, ukuran konten data yang didistribusikan dan jumlah *receiver*. Secara garis besar, semakin tinggi nilai *RTT delay* maka *throughput* yang dihasilkan cenderung semakin rendah. Meskipun pada beberapa kondisi tertentu pada masing-masing aplikasi hal tersebut tidak berlaku. Selain itu, juga dapat dibuktikan bahwa performansi aplikasi *file sharing* UFTP cenderung lebih baik dibandingkan dengan BitTorrent Sync dengan menghasilkan *throughput* yang lebih tinggi ketika digunakan dalam melakukan distribusi konten data dalam berbagai macam ukuran dan kondisi lingkungan pengujian.

5.2 Analisis

Berdasarkan hasil perbandingan yang diperoleh dari seluruh skenario pengujian yang dilakukan pada lingkungan pengujian sistem, masing-masing contoh aplikasi metode *file sharing* dapat melakukan distribusi konten data dengan menghasilkan performansi yang berbeda. Parameter penentuan kualitas performansi yang baik berdasarkan pada nilai *throughput* yang tinggi pada berbagai kondisi lingkungan pengujian yang ada. Pada **Gambar 5.5** dijelaskan mengenai grafik hasil perbandingan performansi *throughput* rata-rata pada pengujian yang didasarkan pada kondisi nilai *RTT delay* pada jaringan komputer lokal. Dari hasil tersebut dapat diketahui bahwa performansi metode *file sharing* berbasis *multicast* dengan contoh aplikasi UFTP menghasilkan performansi *throughput* yang lebih tinggi dibandingkan dengan metode *Peer-to-Peer* pada aplikasi *file sharing* BitTorrent Sync. Hal tersebut dibuktikan dengan nilai rata-rata total *throughput* dari aplikasi UFTP lebih tinggi dibandingkan dengan aplikasi BitTorrent Sync baik dalam kondisi *packet loss* 0% atau 5%.



Gambar 5.5 Grafik perbandingan rata-rata *throughput* total pada UFTP dan BitTorrent Sync

Secara umum, peningkatan nilai *RTT delay* dan *packet loss* pada lingkungan pengujian sistem berpengaruh pada penurunan *throughput* yang dihasilkan pada masing-masing aplikasi. Hasil analisis dari keseluruhan pengujian pada skenario yang digunakan, menunjukkan aplikasi *file sharing* berbasis *multicast* dengan contoh aplikasi UFTP menghasilkan performansi *throughput* yang lebih baik dibandingkan dengan metode *Peer-to-Peer* pada BitTorrent Sync. Setelah dianalisis lebih lanjut, hal tersebut disebabkan karena ketika dilakukan proses distribusi konten data menggunakan aplikasi BitTorrent Sync trafik yang berjalan pada lingkungan pengujian sistem cenderung lebih padat dibandingkan ketika dilakukan distribusi konten data menggunakan aplikasi UFTP. Hal tersebut dibuktikan dengan trafik yang di*capture* oleh aplikasi Wireshark ketika aplikasi BitTorrent Sync berjalan, lebih besar dibandingkan ketika aplikasi UFTP berjalan. Selain itu BitTorrent Sync membutuhkan alokasi memori yang cukup besar dalam menjalankan proses, karena diperlukan integrasi dengan *browser* agar dapat menjalankan tampilan GUI untuk melakukan aktifitas distribusi konten data. Sehingga mengurangi kemampuan dari setiap *host* dalam mengirim dan menerima paket data yang ditransmisikan terutama ketika spesifikasi memori yang digunakan oleh setiap *host* rendah. Sedangkan pada UFTP memori yang digunakan ketika aplikasi UFTP berjalan tidak terlalu besar, karena UFTP menggunakan tampilan *interface* berbasis *Command Prompt*.

BAB 6 PENUTUP

Bab ini mendeskripsikan mengenai kesimpulan dan saran yang diperoleh dari hasil penelitian. Kesimpulan diambil berdasarkan analisis hasil perbandingan performansi yang dilakukan terhadap metode *file sharing* dengan contoh aplikasi UFTP dan BitTorrent Sync. Sedangkan bagian saran, berisi penyataan mengenai permasalahan pengembangan penelitian lebih lanjut.

6.1 Kesimpulan

Dari hasil implementasi lingkungan pengujian sistem, prosedur pengumpulan data dan analisis perbandingan performansi metode *file sharing* berbasis *multicast* dengan *Peer-to-Peer* menggunakan contoh aplikasi UFTP dan BitTorrent Sync, dapat disimpulkan bahwa:

1. Implementasi lingkungan pengujian sistem jaringan komputer lokal yang digunakan dalam proses distribusi konten data melalui metode *file sharing* dapat diimplementasikan dengan memanfaatkan integrasi aplikasi Virtualbox dan WANem. Virtualbox berfungsi untuk melakukan virtualisasi arsitektur sistem jaringan komputer lokal dalam satu *subnet* melalui mode jaringan *NAT Network*. Sedangkan WANem digunakan untuk melakukan emulasi berbagai macam kondisi jaringan komputer lokal ketika dilakukan proses distribusi konten data melalui aplikasi *file sharing* UFTP dan BitTorrent Sync.
2. Dalam menjalankan prosedur pengumpulan data perbandingan performansi ketika dilakukan proses distribusi konten data, setiap aplikasi *file sharing* diimplementasikan ke dalam tiga *host* yang terdiri dari satu *host* yang berperan sebagai *sender* dan dua *host* yang berperan sebagai *receiver* dengan mengikuti skenario pengujian pada satu *sender* dan dua *receiver*. *Sender* mendistribusikan konten data dalam berbagai ukuran pada jaringan komputer lokal pada berbagai macam skenario kondisi jaringan yang disusun berdasarkan kombinasi nilai *RTT delay* dan *packet loss*.
3. Analisis perbandingan performansi pada aplikasi *file sharing* dilakukan menggunakan bantuan aplikasi Wireshark yang diimplementasikan di sisi *traffic monitoring server*. Wireshark berfungsi untuk melakukan monitoring terhadap trafik yang berjalan pada jaringan komputer berupa *throughput* atau kecepatan transfer data yang dihasilkan oleh masing-masing aplikasi *file sharing* ketika melakukan proses distribusi konten data.
4. Hasil analisis perbandingan menujukkan bahwa metode *file sharing* berbasis *multicast* dengan contoh aplikasi UFTP ketika menggunakan mekanisme TFMCC menghasilkan performansi *throughput* yang lebih baik ketika digunakan untuk melakukan proses distribusi konten data pada seluruh skenario lingkungan pengujian jaringan komputer lokal, jika dibandingkan dengan metode *file sharing* berbasis *Peer-to-Peer* dengan contoh aplikasi BitTorrent Sync yang menerapkan mekanisme DHT dengan selisih rata-rata *throughput* yang dihasilkan yaitu 10,70 MBps pada *packet loss* 0% dan 12,54 MBps pada kondisi *packet loss* 5%.

6.2 Saran

Saran yang diperoleh dari penelitian yang dilakukan penulis terkait dengan pengembangan penelitian berikutnya adalah:

1. Diperlukan penelitian lebih lanjut terkait dengan lingkungan pengujian sistem yang digunakan. Penelitian ini menggunakan lingkungan pengujian sistem pada jaringan komputer lokal dalam satu *subnet*. Penelitian lebih lanjut dapat mempertimbangkan pengaruh performansi dari metode *file sharing* dengan contoh aplikasi yang UFTP dan BitTorrent Sync terhadap lingkungan pengujian sistem yang terdiri lebih dari satu *subnet* jaringan.
2. Diperlukan penelitian lebih lanjut mengenai metode *file sharing* lainnya beserta contoh aplikasi yang berbeda dalam melakukan proses distribusi konten data.
3. Diperlukan penelitian lebih lanjut mengenai pengaruh performansi aplikasi *file sharing* terhadap jumlah *host* yang berpartisipasi dalam proses distribusi konten data pada jaringan komputer.

DAFTAR PUSTAKA

Bakharev, A. dan Siemens, E., 2013. *Evaluation of Reliable Multicast Implementations with Proposed Adaption for Delivery of Big Data in Wide Area Networks*. International Conference on Networking and Services (9). Lisbon: International Academy Research and Industry Association.

Camarillo, G., ed., 2009. *Peer-to-Peer (P2P) Architecture: Definition, Taxonomies, Examples, and Applicability*. RFC 5694. California: Internet Engineering Task Force.

Costa-Montenegro, E., Lopez-Bravo, C., Rodriguez-Hernandez, P.S. & Barragans-Martinez, A.B., 2012. *Analysis of the BitTorrent Protocol Modified with Multicast*. New Jersey: Internet Electrical and Electronics Engineers.

Deering, S., 1989. *Host Extensions for IP Multicasting*. RFC 1112. California: Internet Engineering Task Force.

Farina, J., Scanlon M. dan Kechadi, M.T., 2014. *BitTorrent Sync: First Impressions and Digital Forensic Implications*. Dublin: University Collage Dublin.

IXIA, 2005. Multicast: Conformance and Performance Testing. Calabasas: IXIA.

Jacobsen, O.J., eds., 1998. *The Internet Protocol Journal, 1(2)*. San Jose: Cisco System.

Kalita, H.K. dan Rane, S., 2008. WANem 2.0 Wide Area Network Emulator Performance Engineering Research Centre. Mumbai: TATA Consultancy Services.

Klumpp, T., 2013. *File Sharing, Network Architecture, and Copyright Enforcement: An Overview*. Edmonton: University of Alberta.

Kurose, J.F. dan Ross, W.K., 2000. *Computer Networking: A Top-Down Approach Featuring the Internet*. Boston: Addison-Wesley Longman.

Lamping, U. dan Warnicke, E., 2014. *Wireshark User's Guide: For Wireshark 2.1*.

Nambiar. M.K., 2007. *Wide Area Network Emulator Setup Guide (WANem)*. Mumbai: TATA Consultancy Service.

Oracle Corporation, 2015. *Oracle VM Virtualbox User Manual (5.0.10)*. California: Oracle Corporation.

Osterman Research, Inc. 2014. *Best Practices for File Sharing*. Washington: Osterman Research, Inc.

Pingman Tools, LLC. *What's "normal" for latency and packet loss?*. [online] Tersedia di: <https://www.pingman.com/kb/> [Diakses 13 Desember 2015]

Quiin, B. dan Almeroth, K., 2001. *IP Multicast Application Challenges and Solutions*. RFC 3170. California: Internet Engineering Task Force.

Truksans, L., Znots, L. dan Barzdins, G., 2011. *File Transfer Protocol Performance Study for EUMETSAT Meteorologival Data Distribution*. Riga: University of Latvia.

Whitney, J., 1969. *Five key criteria when selecting a business-class file sharing solution*. [online] Tersedia di: <http://www.jeffwhitney.utilizer.com/node/2580753/> [Diakses 7 Desember 2015]

Widmer, J. dan Handley, M., 2006. TCP-Friendly Multicast Congestion Control (TFMCC): Protocol Spesification. RFC 4654. California: Internet Engineering Task Force.

Zhang J. dan McLeod, R.D., 2003. *A UDP-Based File Transfer Protocol with Flow Control Using Fuzzy Logic Approach*. Manitoba: University of Manitoba.

Zhang J., 2005. *Improved Large-Set Data Transport over the Internet using Computational Intelligence Techniques*. Manitoba: University of Manitoba.