TUGAS AKHIR – IF184802

ANALISIS KINERJA MODEL PROPAGASI *TWORAYGROUND* PADA *AD HOC ON DEMAND MULTIPATH DISTANCE VECTOR* (AOMDV) *ROUTING* PADA MANET

A. CHANIF EKA W.
NRP. 05111240007001

Dosen Pembimbing:
Dr. Eng. Radityo Anggoro, S.Kom., M.Sc.

Departement Informatika
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2019

TUGAS AKHIR – IF184802

ANALISIS KINERJA MODEL PROPAGASI *TWORAYGROUND* PADA *AD HOC ON DEMAND MULTIPATH DISTANCE VECTOR* (AOMDV) *ROUTING* PADA MANET

A. CHANIF EKA W.
NRP. 05111240007001

Dosen Pembimbing:
Dr. Eng. Radityo Anggoro, S.Kom., M.Sc.

Departement Informatika
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2019

[*Halaman ini sengaja dikosongkan*]

Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR – IF184802

PERFORMANCE ANALYSIS OF TWORAYGROUND PROPAGATION MODEL ON AD HOC ON DEMAND MULTIPATH DISTANCE VECTOR (AOMDV) ROUTING ON MANET

A. CHANIF EKA W.
NRP. 05111240007001

Advisor:
Dr. Eng. Radityo Anggoro, S.KOM., M.Sc.

Departement of Informatics
Faculty of Information Technology and Communication
Sepuluh Nopember Institut of Technology
Surabaya 2019

[*Halaman ini sengaja dikosongkan*]

LEMBAR PENGESAHAN

ANALISIS KINERJA MODEL PROPAGASI *TWORAYGROUND* PADA *AD HOC DEMAND MULTIPATH DISTANCE VECTOR* (AOMDV) *ROUTING* PADA MANET

TUGAS AKHIR

Diajukan untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Arsitektur Jaringan Komputer
Program Studi S-1 Departemen Informatika
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember Surabaya

Oleh

AHMAD CHANIF EKA WILDANA
NRP. 05111241000701

Disetujui oleh Pembimbing Tugas Akhir:

Dr. Eng. Radityo Anggoro, S.Kom., M.Sc. .....
NIP : 198410162008121002 (Pembimbing)

DEPARTEMEN TEKNIK INFORMATIKA

SURABAYA
JANUARI, 2019

[*Halaman ini sengaja dikosongkan*]

ANALISIS KINERJA MODEL PROPAGASI *TWORAYGROUND* PADA *AD HOC ON DEMAND MULTIPATH DISTANCE VECTOR* (AOMDV) *ROUTING* PADA MANET

Nama Mahasiswa	**:**	**A. Chanif Eka W.**
NRP	**:**	**05111240007001**
Jurusan	**:**	**Teknik Informatika FTIK – ITS**
Dosen Pembimbing	**:**	**Dr. Eng. Radityo Anggoro, S.Kom., M.Sc.**

ABSTRAK

Kehadiran teknologi jaringan *nirkabel* serta berkembangnya perangkat komunikasi *mobile* yang dapat bergerak secara dinamis menjadikan semakin mudahnya setiap perangkat komunikasi *mobile* untuk saling terhubung dan bertukar informasi tanpa adanya infrastruktur tetap yang terpusat. Memanfaatkan kemajuan seperti ini, dimungkinkan untuk dikembangkan suatu lingkungan jaringan yang memungkinkan perangkat komunikasi *mobile* berkomunikasi dengan menggunakan jaringan komunikasi yang bersifat sementara.

Lingkungan infrastruktur jaringan seperti ini disebut dengan *Mobile Ad Hoc Network* (MANET), di mana setiap *node* yang terhubung berupa perangkat komunikasi *mobile* yang bergerak secara dinamis sehingga topologi jaringan yang digunakan harus bisa menyesuaikan pergerakan tersebut dengan menggunakan protokol *routing* yang memadai. Namun implementasi dari MANET pada dunia nyata masih mengalami kesulitan, sehingga banyak penelitian dilakukan dengan

mamanfaatkan aplikasi simulasi seperti *Network Simulator 2* untuk membuat simulasi teknologi jaringan MANET.

Pada Tugas Akhir ini dilakukan penelitian dan analisis kinerja skema lingkungan jaringan MANET dengan menggunakan protokol *routing* AOMDV dan model propagasi *TwoRayGround* pada aplikasi simulasi *Network Simulator 2*. Analisis dilakukan pada dua model skenario uji coba dengan hasil untuk skenario *Model Mobility Node* memiliki nilai *Packet Delivery Ratio (*PDR) mencapai 94.831 %, *End-to-End Delay* (E2D) sebesar 16.101 *milisecond*, serta *Routing Overhead* (RO) sebanyak 69.0. Hasil untuk skenario *Model Traffic Load* memiliki nilai *Packet Delivery Ratio (*PDR) mencapai 93.111 %, *End-to-End Delay* (E2D) sebesar 22.852 *milisecond*, serta *Routing Overhead* (RO) sebanyak 555.8.

Kata Kunci : ***AOMDV, MANET, Network Simulator 2, NS-2, TwoRayGround.***

PERFORMANCE ANALYSIS OF TWORAYGROUND PROPAGATION MODEL ON AD HOC ON DEMAND MULTIPATH DISTANCE VECTOR (AOMDV) ROUTING ON MANET

Name	**:**	**A. Chanif Eka W.**
NRP	**:**	**05111240007001**
Departement	**:**	**Teknik Informatika FTIK – ITS**
Advisior	**:**	**Dr. Eng. Radityo Anggoro, S.Kom., M.Sc.**

ABSTRAK

The presence of wireless network technology and the development of mobile communication devices that can move dynamically make it easier for each mobile communication device to connect and exchange information without a fixed and centralized infrastructure. Making use of this improvement, it is possible to develop a network environment that allows mobile communication devices to communicate using a temporary communication network.

This network infrastructure environment is called Mobile Ad Hoc Network (MANET), where each connected node in the form of a mobile communication device that moves dynamically so, the network topology used must be able to adjust the movement using adequate routing protocols. However, the implementation of MANET in the real life is still experiencing difficulties, much research has been carried out using simulation applications such as Network Simulator 2 to make simulations of MANET network technology.

In this Final Project, research and analysis of the performance of the MANET network environment using AOMDV routing protocol and TwoRayGround propagation model in the Network Simulator 2 simulation application. Analysis was carried out on two pilot scenario models with results for the Mobility Node Model scenario having a Packet Delivery Ratio (PDR) reaches 93.406%, End-to-End Delay (E2D) through 16.101 miliseconds, and Overhead (RO) Routing as much as 69.0. The results for the Traffic Load Model scenario have a Packet Delivery Ratio (PDR) value of 93.111%, End-to-End Delay (E2D) through 22.852 miliseconds, and Overhead Routing (RO) as much as 555.8.

Keywords : ***AOMDV, MANET, Network Simulator 2, NS-2, TwoRayGround.***

KATA PENGANTAR

Bismillahirrohmanirohim.

Alhamdulilahirabil'alamin, segala puji bagi Allah SWT. Atas segala rahmat dan karunia-Nya yang tak terhingga sehingga penulis dapat menyelesaikan Tugas Akhir dengan judul:

"ANALISIS KINERJA MODEL PROPAGASI *TWORAYGROUND* PADA *AD HOC ON DEMAND MULTIPATH DISTANCE VECTOR* (AOMDV) *ROUTING* PADA MANET".

Terselesaikannya Tugas Akhir ini tidak lepas dari bantuan banyak pihak. Oleh karena itu melalui lembar ini penulis ingin mengucakan terima kasih dan penghormatan sebesar-besarnya kepada pihak-pihak sebagai berikut:

1. Allah SWT, karena limpahan rahmat dan karunia-Nya penulis dapat menyelesaikan Tugas Akhir dan juga perkuliahan di Departemen Informatika ITS.

2. Ayah dan Ibu Penulis, adik, beserta keluarga dekat yang tiada henti memberikan semangat, do'a, serta dukungan penuh kepada penulis selama ini sehingga dapat menyelesaikan Tugas Akhir dan perkuliahan di Departemen Informatika ITS Surabaya.

3. Bapak Dr. Eng. Raditio Anggoro S.Kom, M.Sc selaku dosen pembimbing dan dosen wali dari penulis yang telah memberikan bimbingan, dukungan, masukan, nasihat, dan banyak arahan kepada penulis dalam menyelesaikan Tugas Akhir ini.

4. Bapak Dr. Darlis Herumurti, S.Kom., M.Kom., selaku ketua Jurusan Teknik Informatika ITS.

5. Bapak Ibu dosen yang tidak dapat disebutkan satu-persatu yang telah mendidik dan mengajar dengan baik dan penuh kesabaran.

6. Sahabat dan teman-teman dari penulis terutama sahabat seperjuangan selama perkuliahan keluarga D'12 yang telah sama-sama berjuang dalam masa perkuliahan dari awal sampai telah berhasil menyelesaikan perkuliahan di ITS Surabaya. Semoga ilmu yang didapatkan senantiasa memberikan manfaat.

7. Teman-teman TC angkatan 2012, kakak angkatan, dan juga adik angkatan yang telah ramah dan berbaik hati membantu penulis selama masa perkuliahan.

8. Pihak-pihak yang tidak dapat penulis sebutkan satu per satu, yang telah membantu penulis dalam menyelesaikan Tugas Akhir ini.

Penulis menyadari bahwa laporan Tugas Akhir ini masih jauh dari kata sempurna. Oleh karena itu, dengan segala kerendahan hati penulis mengharapkan kritik dan saran pembaca untuk perbaikan kedepannya. Penulis berharap laporan Tugas Akhir ini dapat memberikan manfaat dan berguna bagi pembaca secara umum. Semoga Allah SWT, selalu memberikan rahmat dan karunia-Nya serta memberikan balasan terbaik atas segala kebaikan yang telah dilakukan.

Surabaya, 28 Nopember 2018

A. Chanif Eka W.

DAFTAR ISI

[*Halaman ini sengaja dikosongkan*]

DAFTAR GAMBAR

DAFTAR TABEL

[*Halaman ini sengaja dikosongkan*]

DAFTAR LAMPIRAN

[*Halaman ini sengaja dikosongkan*]

BAB I
PENDAHULUAN

Bab ini memaparkan mengenai garis besar Tugas Akhir yang terdiri dari latar belakang, tujuan, rumusan dan batasan permasalahan, metodologi pembuatan Tugas Akhir, serta sistematika penulisan.

1.1 LatarBelakang

Pada saat ini, teknologi bidang komunikasi telah berkembang dengan pesatnya. Kehadiran teknologi jaringan *nirkabel* serta berkembangnya perangkat komunikasi *mobile* yang dapat bergerak secara dinamis menjadikan semakin mudahnya setiap perangkat komunikasi untuk saling terhubung dan bertukar informasi tanpa adanya infrastruktur tetap yang terpusat.

Memanfaatkan pergerakan perangkat komunikasi *mobile* yang dapat bergerak secara dinamis, salah satu teknologi yang dapat dikembangkan dengan sebuah infrastruktur jaringan yang dinamis adalah teknologi *Mobie Ad-Hoc Network* atau yang lebih dikenal dengan MANET. MANET merupakan suatu lingkungan jaringan yang terdiri dari sekumpulan perangkat komunikasi *mobile* seperti *smartphone*, laptop, tablet, dan sejenisnya yang saling terhubung dan bertukar informasi satu sama lainnya [9].

Adanya pergerakan dinamis dari perangkat komunikasi *mobile* pada jaringan MANET menyebabkan terjadinya perubahan pada topologi yang digunakan pada lingkungan jaringan MANET tersebut. Sehingga diperlukan suatu protokol *routing* agar rute *routing* yang diberikan pada lingkungan jaringan MANET tetap optimal.

Salah satu protokol *routing* yang bisa digunakan pada lingkungan jaringan MANET adalah *Ad Hoc on Demand Multipah Distance Vector (*AOMDV). *Routing* AOMDV merupakan suatu *multipath routing protocol* yang memiliki

kelebihan berupa toleransi kesalahan, peningkatan *bandwidth*, dan penigkatan keamanan [6].

Implementasi lingkungan jaringan MANET dapat dilakukan simulasi dengan baik menggunakan aplikasi *Network Simulator 2* (NS-2), sehingga dapat dilakukan analisis kinerja model propagasi *TwoRayGround* pada protokol *routing Ad Hoc on Demand Multipah Distance Vector* (AOMDV) di lingkungan jaringan MANET dengan menggunakan *Network Simulator 2.*

1.2 RumusanMasalah

Rumusan masalah yang diangkat dalam Tugas Akhir ini dapat dipaparkan sebagaimana berikut:

1. Bagaimana melakukan implementasi model propagasi *TwoRayGround* dengan protokol *routing* AOMDV di lingkungan jaringan MANET pada *Network Simulator 2*?
2. Bagaimana bentuk simulasi dan ujicoba model propagasi *TwoRayGround* dengan protokol *routing* AOMDV pada lingkungan MANET dengan menggunakan *Network Simulator 2*?
3. Bagaimana kinerja Model propagasi *TwoRayGround* dengan protokol *rouiting* AOMDV di lingkungan jaringan MANET ?

1.3 TujuanPenelitian

Tujuan pembuatan Tugas Akhir ini adalah untuk mendapatkan hasil analisis kinerja model propagasi *TwoRayGround* pada lingkungan MANET dengan protokol *routing Ad Hoc on Demand Multipath Distance Vector* (AOMDV) menggunakan aplikasi *Network Simulator 2* (NS-2).

1.4 BatasanMasalah/RuangLingkupMasalah

Permasalahan yang dibahas dalam Tugas Akhir ini memiliki beberapa batasan antara lain:

1. Protokol *routing* yang digunakan adalah *Ad Hoc on Demand Multipath Distance Vector* (AOMDV).
2. Model propagasi yang digunakan adalah *TwoRayGround*.
3. Lingkungan jaringan yang digunakan untuk melakukan ujicoba adalah *Mobile Ad Hoc Network* (MANET).
4. Aplikasi untuk melakukan ujicoba dan simulasi adalah *Network Simulator* 2 (NS-2).

1.5 Metodologi Penelitian

a. Penyusunan proposal tugas akhir

Proposal Tugas Akhir ini berisi tentang deskripsi pendahuluan dari Tugas Akhir yang dibuat. Pendahuluan ini terdiri atas hal yang menjadi latar belakang diajukannya usulan Tugas Akhir, rumusan masalah yang diangkat, batasan masalah untuk Tugas Akhir, tujuan dari pembuatan Tugas Akhir, dan manfaat dari hasil pembuatan Tugas Akhir. Selain itu dijabarkan pula tinjauan pustaka yang digunakan sebagai referensi pendukung pembuatan Tugas Akhir. Subbab metodologi berisi penjelasan mengenai tahapan penyusunan Tugas Akhir mulai dari penyusunan proposal hingga penyusunan buku Tugas Akhir. Terdapat pula subbab jadwal kegiatan yang menjelaskan jadwal pengerjaan Tugas Akhir.

b. Studi literatur

Pada studi literatur akan dipelajari sejumlah refrensi yang diperlukan dalam melakukan analisis kinerja model propagasi *TwoRayGround* pada lingkungan MANET dengan protokol *routing Ad Hoc on Demand Multipath Distance Vector* (AOMDV) menggunakan aplikasi *Network Simulator 2* (NS-2) sebagaimana berikut:

- Refrensi mengenai model propagasi *TwoRayGround*.
- Refrensi tentang bentuk *routing* AOMDV.
- Refrensi mengenai lingkungan jaringan MANET.
- Refrensi mengenai cara kerja *Network Simulator 2* (NS-2).

c. Analisis dan desain perangkat lunak

Pada tahap ini dilakukan analisis dan desain awal terhadap simulasi *routing* MANET.

d. Implementasi

Implementasi dilakukan dengan membuat *prototype* dan pembuatan desain simulasi *routing*.

e. Pengujian dan evaluasi

Pengujian dilakukan degan mejalankan *prototype* dan desain simulasi yang sudah dibuat untuk mengetahui kesesuaian dengan perancangan yang sudah dibuat. Selain dari pada itu, dilakukan evaluasi kinerja jika ditemukan kesalahan.

f. Penyusunan Buku Tugas Akhir

Pada tahap ini dilakukan penyusunan laporan yang menjelaskan dasar teori dan metode yang digunakan dalam Tugas Akhir serta hasil dari implementasi aplikasi perangkat lunak yang telah dibuat. Sistematika penulisan buku tugas akhir secara garis besar antara lain:

1. Bab I. Pendahuluan
   a. Latar Belakang
   b. Rumusan Masalah
   c. Batasan Tugas Akhir
   d. Tujuan
   e. Metodologi
   f. Sistematika Penulisan

2. Bab II. Tinjauan Pustaka

   Bab ini membahas beberapa teori penunjang yang berhubungan dengan pokok pembahasan dan mendasari pembuatan Tugas Akhir ini.

3. Bab III. Perancangan

   Bab ini berisi desain dan perancagan dari metode yang akan diimplementasikan dan dilakukan pengujian dari analisi yang akan dilakukan.

4 Bab IV. Implementasi

   Bab ini membahas implementasi dari desain dan perancangan yang telah dilakukan pada Bab

sebelumnya. Implementasi *routing protocol* AOMDV pada lingkungan MANET.

5. Bab V. Pengujian dan Evaluasi

   Bab ini menjelaskan mengenai pengujian dan evaluasi model propagasi *TwoRayGround* dengan *routing protocol* AOMDV pada lingkungan jaringan MANET untuk mendapatkan hasil analisis performa.

6. Bab VI. Penutup

   Bab ini berisi tentang kesimpulan dan hasil performa dari model propagasi *TwoRayGround* dengan *routing protocol* AOMDV pada lingkungan jaringan MANET serta rekomendasi saran untuk mendapatkan hasil performa yang lebih baik.

BAB II
TINJAUAN PUSTAKA

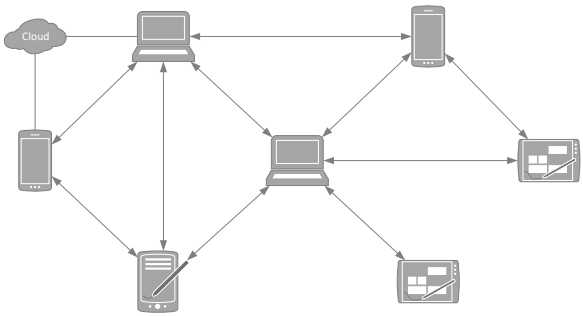
Bab ini berisi pembahasan mengenai dasar teori dan penjelasan dari metode serta *tool* yang digunakan dalam Tugas Akhir. Penjelasan bertujuan untuk memberikan gambaran secara umum serta penunjang mengenai pengembangan riset yang memiliki kaitan.

2.1 *Mobile Ad Hoc Network* (MANET)

Mobile Ad-Hoc Network pada mulanya merupakan suatu sistem radio paket yang dibiayai oleh DAPRA *(Defense Advanced Research Project Agency*) untuk kebutuhan perang pada awal tahun 1970-an [16].

Secara umum jaringan *Mobie Ad-Hoc Network* atau yang lebih dikenal dengan sebutan MANET merupakan suatu jaringan *Autonomus Terminal* di mana setiap *node* dapat berperan sebagai *router* ataupun berperan sebagai perangakat komunikasi *mobile* biasa. *Node* pada lingkungan jaringan MANET berupa perangkat komunikasi *mobile* seperti *smartphone,* laptop, tablet, dan sejenisnya yang bergerak secara dinamis dan bertukar informasi satu sama lainnya secara *Distributed Operation* tanpa adanya infrastruktur terpusat [14].

Sebagaimana ditunjukkan pada Gambar 2.1, setiap *mobile node* dalam lingkungan jaringan MANET dapat meneruskan paket dari satu *mobile node* menuju *mobile node* selanjutnya secara *nirkabel* tanpa harus terhubung dengan perangkat *router* ataupun *access poin* [12].



Gambar 2.1 Skema *Mobile Ad Hoc Network*

Di dalam penerapannya, MANET memiliki beberapa keuntungan seperti pengaturan sistem jaringan *nirkabel* yang mudah dan cepat sehingga dapat mengurangi kebutuhan dalam penggunaan kabel, jaringan dapat diperluas di tempat-tempat yang tidak dapat dijangkau kabel, jaringan *nirkabel* memberikan lebih banyak fleksibilitas dan kemudahan dalam beradaptasi dengan perubahan konfigurasi dan topologi jaringan [9].

Jaringan MANET cocok digunakan dalam situasi dengan infrastruktur yang memadai dengan biaya yang efektif seperti dalam bidang Bisnis, Militer, Kondisi operasional darurat, *Vehicular ad hoc network* (VANET), *Wirelees sensor network*, dan *wireless mesh network.*

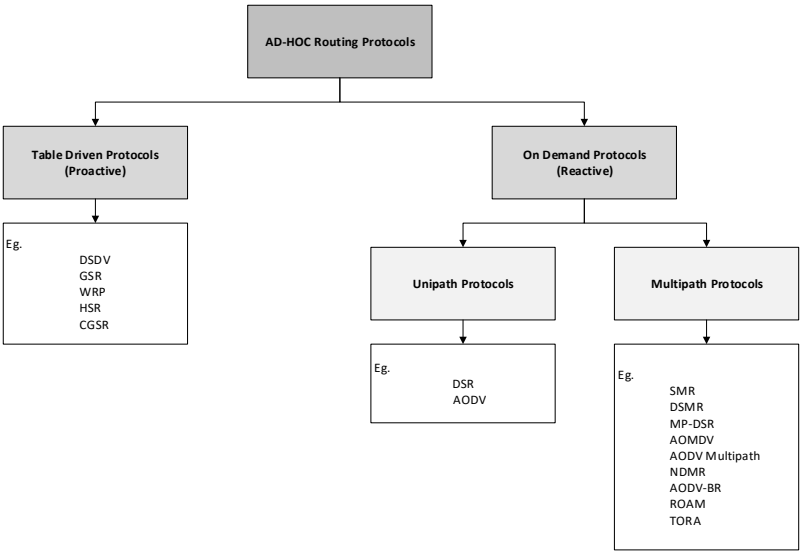
Beberapa hal yang perlu diperhatikan dalam penerapan jaringan MANET antara lain adalah tidak terprediksinya kondisi lingkungan di mana jaringan MANET diterapkan, kurang bisa diandalkannya jaringan *nirkabel* yang ada, *node* yang terhubung tidak memiliki sumber daya yang memenuhi untuk suatu jaringan MANET yang optimal, dinamisnya topologi jaringan MANET

Beberapa hal di atas dapat menyebabkan adanya *node failur*, *link failur*, *route breakages* (rute kadaluarsa), dan

congestion node atau *link* yang menyebabakan penundaan atau kehilangan paket sehingga jaringan MANET tidak bisa digunakan secara optimal.

Pergerakan yang dinamis dari setiap *mobile node* pada lingkungan jaringan MANET menyebabkan pula perubahan pada topologi dan konektifitas dari setiap *mobile node* yang saling terhubung, sehingga topologi jaringan pada MANET bersifat *Dynamic Network Topology*. Oleh karena itu diperlukan adanya protokol *routing* yang dapat memenuhi kebutuhan jaringan untuk memberikan jalur *routing* yang optimal [5].

Protokol *routing* adalah suatu aturan yang digunakan untuk menentukan bagaimana *node* saling berkomunikasi satu sama lainnya dan melakukan pemilihan rute untuk menyebarkan informasi.



Gambar 2.2 Klasifikasi *Routing Protocol*

Ada 2 jenis protokol *routing* pada *wireless network* yaitu protokol *routing reaktif* dan protokol *routing proaktif* [7].

1. Protokol *Routing Proactive*

   Protokol *routing proactive* merupakan suatu protokol yang melakukan *routing* berdasarkan tabel *routing* yang diperbaharui terus-menerus secara reguler. Protokol *routing proactive* melakukan pengelolaan tujuan pengiriman paket data dan rute yang dilewati dengan cara mendistribusikan tabel *routing* ke seluruh jaringan, proses ini dilakukan secara berulang dari awal jika terjadi rekonstruksi *routing* sehingga dapat memperlambat pengiriman paket data.

   Beberapa protokol *routing* yang masuk ke dalam kategori *proactive routing* adalah sebagai berikut: DSDV (*Destination Sequences Distance Vector*), GSR (*Geographic Source Routing*), WRP (*Wireless Routing Protocol*), HSR (*Hierarchically Segmented Routing)*, dan CGSR (*Clusterhead Gateway Switch Routing*).

2. Protokol *Routing Reactive*

   Protokol *routing reaktif* adalah protokol yang bekerja berdasarkan permintaan untuk membuat rute baru atau perubahan rute. Protokol *routing reaktif* bekerja sesuai permintaan untuk melakukan pembuatan rute baru atau pembaharuan rute.

   Pencarian rute dilakukan dengan mengirimkan peket *router request* ke seluruh jaringan hingga ditemukan rute paling efektif untuk mengirimkan paket sampai ke tujuan.

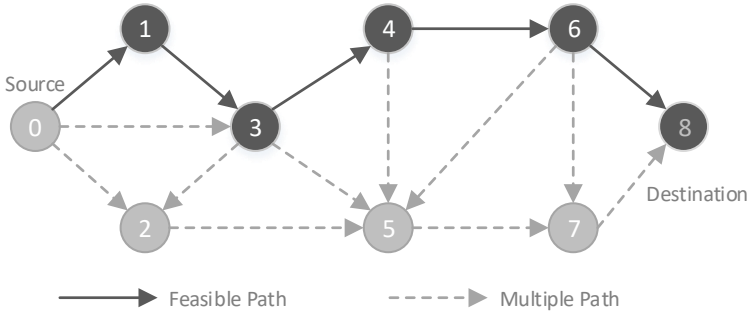
Protokol *routing* yang masuk ke dalam kategori *reactive routing* terkelompokkan lagi menjadi dua sub-kategori yaitu *Protocol Unipath* seperti protokol *routing* DSR (*Dynamic Source Routing*) dan AODV (*Ad Hoc On-Demand Distance Vector*), serta *Protocol Multipath* seperti protokol SMR (*Split Multipath Routing*), DSMR (*Dynamic Source Multi-path Routing*), MP DSR (*Multi-path Dynamic Source Routing*), AOMDV (*Ad-hoc On-Demand Multipath Distance Vector)*, AODV *Multipath*, NDMR (*Node-Disjoint Multipath Routing*), AODV BR (*Ad Hoc On-Demand Distance Vector Backup Routing*), ROAM (*Routing On-demand Acyclic Multipath*), TORA (*Temporally Ordered Routing Algorithm*).

Tujuan utama dari strategi *routing* adalah untuk memberikan data secara efisien mengenai rute *routing* dari sumber ke tujuan pengiriman paket. Meskipun semua protokol *routing* memiliki tujuan yang sama yaitu untuk mendapatkan rute yang efektif, namun setiap protokol *routing* mengadopsi pendekatan yang berbeda dalam mencapainya.

Pencarian rute dalam MANET tergantung pada banyak faktor termasuk topologi, pemilihan *router*, spesifik implementasi, dan lingkungan tempat implementasi MANET. Strategi *routing* memiliki dampak yang signifikan terhadap kinerja dari *Jaringan Mobile Ad hoc Network* seperti efisiensi energi, waktu penundaan pengiriman paket (*delay*), besaran paket, dan lainnya.

2.2 Ad Hoc on Demand Multipath Distance Vector (AOMDV)

Ad Hoc on Demand Multipah Distance Vector (AOMDV) merupakan suatu *multipath routing protocol* perluasan dari r*outing Ad Hoc on Demand Distance Vector* (AODV) yang memiliki dua prinsip utama berupa penemuan dan pemeliharaan rute [6].



Gambar 2.3 Proses *Routing* pada AOMDV

Proses *routing* pada AOMDV memiliki perbedaan dengan proses *routing* AODV, di mana pada proses *routing* AODV semua duplikat RREQ (*Route Request*) dari *route discovery* dihapuskan. Sedangkan pada proses *routing* AOMDV beberapa duplikat RREQ tetap digunakan untuk melakukan pencarian rute alternatif. Namun hal penting yang perlu diperhatikan dalam pencarian rute alternatif pada AOMDV adalah rute harus tetap bersifat *loop-free* dan saling lepas (*disjoint*) [10].

Kelebihan dalam penggunaan *routing* AOMDV adalah berupa toleransi kesalahan, peningkatan *bandwidth*, dan penigkatan keamanan. Di sisi lain, terjadinya *overlapping, infinity*

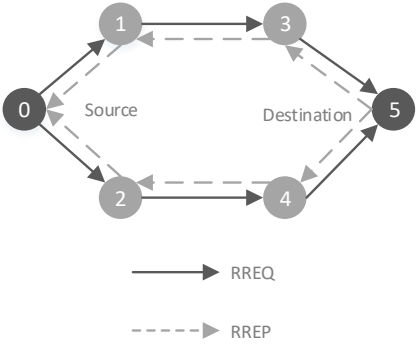
loop, dan *node disjointed* menjadi kendala dalam penerapan *routing* AOMDV [8].

AOMDV melibatkan penemuan rute dan fase pemeliharaan rute yang mirip dengan AODV. Fitur penting dari protokol AOMDV adalah penggunaan informasi *routing* yang sudah tersedia dalam protokol AODV dengan didasari sebanyak mungkin penemuan rute *routing*. Jadi sedikit tambahan biaya diperlukan untuk perhitungan beberapa jalur. Protokol AOMDV memiliki dua komponen utama:

1. Pencarian Rute Baru

   Proses pencarian rute dimulai ketika sebuah rute diperlukan oleh *node* sumber untuk mengirim paket ke *node* tujuan, sedangkan tidak tersedia informasi rute pada tabel *routing.*

   Proses pencarian rute baru memiliki dua fase utama yaitu fase *Route Request* (RREQ) dan fase *Route Replay* (RREP). Proses pengiriman RREQ dan RREP dalam pencarian suatu rute dapat dilihat pada Gambar 2.2.



Gambar 2.4 Proses Pengiriman RREQ dan RREP

Pertama, *node* sumber men-*generate* sebuah pesan *Route Request* (RREQ) untuk disebarkan ke *node* tetangga yang berada dalam jangkauan transmisi *node* sumber. Ketika sebuah *node* tetangga menerima pesan RREQ, *node* tersebut akan melakukan proses pengiriman pesan balasan *Route Reply* (RREP) jika *node* tersebut adalah *node* tujuan, atau menyebarkan ulang pesan RREQ ke *node* tetangga jika *node* tersebut bukanlah *node* tujuan pengiriman paket. *Node* selanjutnya akan menyebarkan ulang pesan RREQ yang diterimanya sampai *node* tujuan menerima pesan RREQ Jika *node* tujuan telah menerima pesan RREQ, maka *node* tujuan akan mengirim *Route Reply* (RREP) ke *node* asal melalui *node* tetangganya.

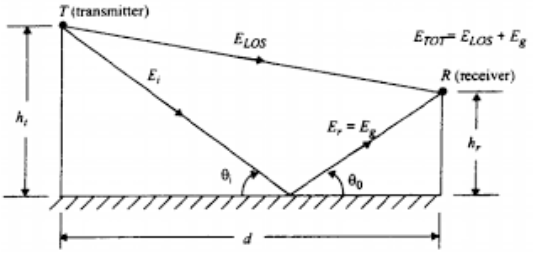
2. Pemeliharaan Rute

   Kegagalan rute pada MANET dapat disebabkan oleh adanya *mobilitas node*, kemacetan, tabrakan paket, kegagalan *node* dan lain sebagainya. Pada protokol AOMDV, terdapat umpan balik lapisan link dari *IEEE 802.11* yang digunakan untuk mendeteksi kegagalan rute. Ketika sebuah *node* mendeteksi adanya *link* yang rusak, maka *node* tersebut akan menyebarkan paket *Route Error* (RRER) ke tetangganya. *Node* tetangga kemudian menyebarkan ulang paket RRER sampai *node* sumber menerima RRER. Jika *node* sumber menerima RRER, *node* ini akan menghapus setiap entri pada tabel *routing* yang menggunakan rute rusak. Berbeda dengan protokol *routing* satu jalur, paket rute yang rusak berisi informasi tentang jalur rute yang rusak dan rute cadangan yang

rusak. Ketika *node* sumber menerima RRER, *node* sumber akan menghapus semua entri rute yang rusak dan menggunakan rute cadangan terpendek. *Node* sumber akan memulai pencarian rute jika semua rute cadangan rusak.

2.3 Model Propagasi *TwoRayGround*

Model propagasi *TwoRayGround* merupakan model propagasi optik geometris yang memodelkan perambatan sinyal pada media *nirkabel* tidak hanya sebagai suatu saluran langsung *Line of Sight* (LOS) antara pemancar sinyal/*transmitter* dan penerima/*reciver*, namun juga memperhatikan saluran pantulan permukaan (*ground reflection*) dalam perambatan sinyal dari *transmitter* ke penerima [1]. Sehingga kuat sinyal yang diterima merupakan suatu transmisi saluran langsung dan juga refleksi di permukaan tanah antara pengirim dan penerima.



Gambar 2.5 Cara Kerja Model Propagasi *TwoRayGround*

Mekanisme kerja dari propagasi *TwoRayGround* ditunjukkan sebagaimana Gambar 2.5. Di mana h_t merupakan tinggi antena *transmitter*, h_r merupakan tinggi antena penerima, dan *d* merupakan jarak antara antena pemancar dan antena penerima.

Model propagasi *TwoRayGround* merupakan salah satu model propagasi yang tidak sulit untuk dilakukan pengemabangan dan sangat baik digunakan pada konfigurasi jaringan dengan skala yang luas dengan menggunakan menara yang tinggi [17].

Kuat sinyal yang diterima oleh antena penerima pada model propagasi *TwoRayGround* diperhitungkan dengan menggunakan Persamaam (1):

$$P_r(d) = \frac{P_t G_t G_r h_t^2 h_r^2}{d^4 L} \tag{1}$$

di mana:

$P_r(d)$ adalah kekuatan sinyal yang dipancarkan,

P_t adalah kekuatan sinyal yang ditrasmisikan,

G_t adalah tegangan pada antena *transmitter*,

G_r adalah tegangan pada antena penerima,

d adalah jarak antara antena *transmitter* dan antena penerima,

h_t adalah tinggi antena *transmitter*,

h_r adalah tinggi antena penerima.

Dari Persamaan (1), meskipun kekuatan sinyal akan menurun seiring dengan bertambahnya jarak antara antena *transmitter* dan antena penerima namun model propagasi *TwoRayGround* tidak bisa memberikan hasil kekuatan sinyal yang maksimal untuk jarak dekat disebabkan oleh konstruksi dan kombinasi dari dua bias sinar langsung dan sinar pantul dari model propagasi *TwoRayGraound* [15].

2.4 *Network Simulator 2* (NS-2)

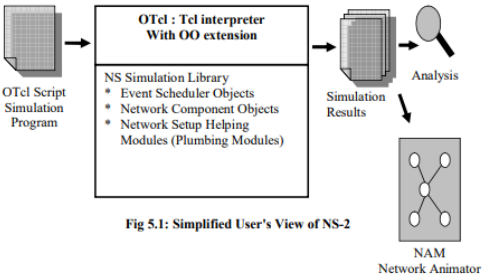
Network Simulator 2 (NS-2) merupakan suatu aplikasi yang bersifat *open source* di bawah *Gnu Public Licences* (GPL) yang dapat dipergunakan untuk mempelajari struktur dinamis dari suatu jaringan komunikasi. *Network Simulator 2* yang muncul sejak tahun 1989 merupakan sebuah *event-driven simulator* yang didesain secara spesifik untuk penelitian dalam bidang jaringan komunikasi komputer [11].

Tahapan simulasi dimulai dengan pembuatan *object* dan *event scheduler* menggunakan *Tool Command Language* (TCL), Kemudian dilanjutkan dengan pembuatan topologi jaringan serta memberikan definisi untuk pola trafik dan *trace file*. Dilakukan pengaturan skenario jalannya simulali terlebih dahulu sebelum simulasi dijalankan.

Pada software NS-2, *user* hanya melakukan pembuatan topologi dan skenario simulasi yang sesuai dengan riset. Pemodelan media, protokol dan *network* komponen lengkap dengan perilaku trafiknya sudah tersedia pada *library* NS-2. NS-2 bersifat *open source* di bawah GPL (Gnu Public License), sehingga NS-2 dapat diunduh melalui *website* NS-2 http://www.isi.edu/nsnam/dist.

Konsep dasar *network simulator* adalah untuk menampilkan secara simulasi proses komunikasi yang berlangsung baik komunikasi dengan menggunakan kabel maupun komunikasi *nirkabel*. Komunikasi yang berlangsung dapat dilakukan oleh *node* tidak bergerak dan juga *node* bergerak sesuai dengan kebutuhan simulasi, namun tentunya tidak sama persis dengan keadaan yang sebenarnya.

Network Simulator 2 menggunakan dua buah bahasa pemrograman standar, yaitu simulasi berorientasi obyek yang ditulis dalam C++ dan OTcl (*object oriented extension of Tcl*) untuk eksekusi terhadap perintah-perintah yang ditulis sebagai skrip dari NS-2.



Gambar 2.6 Alur Simulasi pada *Network Simulator 2*

Alur simulasi pada *Network simulator 2* dilakukan dengan pembuatan skenario simulasi dan skrip simulasi dalam format skrip tcl (*.tcl) yang nantinya akan dieksekusi oleh *Network Simulator 2* dari awal simulasi hinga akhir simulasi yang menghasilkan *file* berbentuk *trace file* yang bisa dilakukan analisis serta hasil animasi yang bisa dijalankan menggunakan program animasi *Networ Animator* atau NAM.

2.5 Generator *File Node-Movement (Mobility Generator)*

Pergerakan random dari *node* dalam jaringan diambil dari *Generator File Node-Movement* hasil dari *tool 'setdest'* yang dikembangkan oleh CMU (*Carnegie Mellon University*). Pergerakan *node* dapat dihasilkan dengan kecapatan tertentu menuju lokasi acak atau menuju suatu lokasi spesifik dalam kawasan yang telah ditentukan. Saat *node* tiba pada satu lokasi pergerakan, *node* dapat diberikan jeda berhenti sementara waktu sebelum melanjutkan ke lokasi pergerakan selanjutnya.

Proses pembangkitan skenario pergerakan *node* pada *Network Simulator 2* menggunkan modul yang telah disediakan oleh *Network Simulator 2* yang terdapat pada *derektori "~ns-allinone-2.35-RC7\ns-2.35\indep-utils\cmu-scen-gen\setdest",*

yang terdiri dari *file setdest.h* dan *setdest.cc*. Cara menjalankan dan mengubah parameter yang harus dimasukkan adalah sebagai berikut:

```
setdest [-v version] [-n number of nodes]
[-p pausetime] [-M maxspeed] [-t time] [-x maxX]
[-y maxY] > [outputfile/file-movement]
```

Gambar 2.7 Formad *Command Line 'setdest'*

Penjelasan parameter dari format *'setdest'* di atas adalah sebagai berikut:

Tabel 2.1 Keterangan Parameter pada *Command Line 'setdest'*

Parameter	Keterangan
-v version	Versi dari '*setdest' network simulator* yang digunakan
-n number of nodes	Jumlah *node* dalam skenario simulasi
-p pausetime	Durasi waktu berhenti sebuah *node* ketika telah sampai pada suatu lokasi pergerakan
-M maxspeed	Kecepatan maksimal dari pergerakan sebuah *node*. *Node* akan bergerak dalam rentang kecepatan (0, maxspeed)
-t time	Waktu berlangsungnya simulasi
-x maxX	Panjang 'X' maksimal area simulasi
-y maxY	Lebar 'Y' maksimal area simulasi
Output file	Nama *file*/tempat *file-movement*

Hasil *output File Node-Movement* dari *command line'setdest'* berisi sejumlah *node* beserta *mobilitas* dan lokasi tujuan pergerkan dari *node* tersebut. Selain itu, *file output* juga berisi sejumlah statistik lain tentang perubahan *link* dan rute [12]

Berikut satu contoh *commant line* untuk pembuatan sebuah *file node-movement:*

```
setdest  -v 1  -n 50   -p 10.0  -M 15.0  -t 200.0
-x 510  -y 510  >  scen-15-test
```

Gambar 2.8 Contoh *Command Line 'setdest'*

Hasil eksekusi *command line* di atas akan menghasilkan sebuah *file node-movement* dengan jumlah *node* simulasi 50 *node*, bergerak dengan kecepatan maksimal 15 m/s, jeda waktu pergerakan 10 detik, lama waktu simulasi 200 detik, luas area simulasi 510 x 510 meter^2 , dan disimpan dengan nama *file node-movement "scen-15-test":*

2.6 *File Traffic-Connection Pattern (Traffic Generation)*

Pola trafik di-*generate* secara acak *(traffic-connection pattern generator)* oleh *file 'cbrgen.tcl'*. *File* tersebut telah disediakan oleh NS-2 dan berada pada direktori *"~ns-allinone-2.35RC7\ns-2.35\indep-utils\cmu-scen-gen"*. Berikut adalah parameter-parameter yang harus ditentukan untuk membuat sebuah *file traffic connection pattern* yaitu:

```
ns cbrgen.tcl [-type cbr|tcp] [-nn nodes] [-seed
seed]   [-mc   connections]   [-rate   rate]   >
[outputfile/traffic-connection]
```

Gambar 2.9 Format *Command Line 'cbrgen.tcl'*

Penjelasan parameter dari format *command line* di atas adalah sebagai berikut:

Tabel 2.2 Keterangan Parameter *Command Line 'cbrgen.tcl'*

Parameter	Keterangan
-type cbr|tcp	Jenis *traffic-connection* yang digunakan pada simulasi (CBR/TCP)
-nn nodes	Jumlah *node*
-seed seed	*Random seed*
-mc connections	Jumlah koneksi maksimal
-rate rate	Jumlah paket per detik
outputfile	Nama file/tempat *traffic-connection*

Berikut adalah satu contoh *command line* untuk membuat suatu *file traffic-connection pattern* tipe CBR dengan jumlah *node* 2, jumlah konesksi maksimal 1, nilai *seed* 1.0, jumlah pengiriman paket per detik adalah 4.0, dan disimpan dengan nama *file "cbr-20-test"* :

```
ns  cbrgen.tcl  -type cbr  -nn 2  -seed 1.0  -mc 1
-rate 4.0  >  cbr-20-test
```

Gambar 2.10 Contoh *Command Line 'cbrgen.tcl'*

Hasil eksekusi *command line* di atas memuat semua paremeter untuk menghubungkan antara *node* 1 dan *node* 2 pada waktu (2.5568388786897245). Output lengkap dari *file traffic-connection pattern* sebagai berikut:

```
#
# nodes: 2, max conn: 1, send rate: 4.0, seed: 1.0
#
#
# 1 connecting to 2 at time 2.5568388786897245
#
set udp_(0) [new Agent/UDP]
$ns_ attach-agent $node_(1) $udp_(0)
set null_(0) [new Agent/Null]
$ns_ attach-agent $node_(2) $null_(0)
set cbr_(0) [new Application/Traffic/CBR]
$cbr_(0) set packetSize_ 512
$cbr_(0) set interval_ 4.0
$cbr_(0) set random_ 1
$cbr_(0) set maxpkts_ 10000
$cbr_(0) attach-agent $udp_(0)
$ns_ connect $udp_(0) $null_(0)
$ns_ at 2.5568388786897245 "$cbr_(0) start"
#
#Total sources/connections: 1/1
#
```

Gambar 2.11 Contoh *File Traffic-Connection Pattern*

Agen yang digunakan dalam koneksi adalah UDP yang akan menghubungkan antara *node* 1 dan *node* 2 sebagaimana yang tertera pada Gambar 2.11.

2.7 AWK (*Aho, Weinberger, Kernighan*)

AWK yang merupakan singkatan dari nama akhir ketiga penemunya *Alfred **A**ho, Peter **W**einberger dan Brian **K**erningham* adalah suatu bahasa pemrograman *text-processing* yang dapat digunakan untuk melakukan ekstrasi data, membuat laporan, mengelola *database* sederhana, memvalidasi data, membuat indeks dokumen.

AWK bersifat *data-driven* yang terdiri dari sekumpulan perintah yang dapat dijalankan pada data tekstual baik secara langsung pada *file* ataupun digunakan sebagai bagian dari *pipeline*. Seperti bahasa pemrograman lainnya, AKW terdiri dari variable, kondisi, dan *looping*. AWK juga dapat digunakan untuk melakukan perhitungan aritmatika.

Fungsi dasar AWK adalah untuk melakukan pencarin per baris pada suatu *file* dengan pola tertentu hingga ke akhir baris. AWK memiliki standart tertentu dalam bentuk *file input* yang digunakan hanya berupa *file* berbentuk teks, AWK tidak bisa digunakan untuk melakukan pemprosesan pada *file* yang tidak mengandung teks [4].

Dalam penulisannya, AWK memiliki 2 *pattern* penting sebagai kata kunci yaitu dimulai dengan *BEGIN* dan diakhiri dengan *END*. Berikut format penggunaan *BEGIN* dan *END* dalam sintaks AWK:

```
BEGIN{Action}

{ACTION}      #Action untuk setiap baris dalam file

END{Action}
```

Gambar 2.12 *Pattern* Penulisan AWK

Berikut adalah prinsip kerja awk dalam memproses data pada suatu *file*:

a. Awk akan membaca *file input* pada tiap baris.
b. Pada tiap baris yang dibaca, jika ditemukan data sesuai *pattern* yang ada maka dilakukan proses sesuai dengan *action* yang ada.
c. Jika data tidak ditemukan sesuai *pattern* yang ada, maka tidak *action* yang akan diproses.

d. Jika tidak terdapat pencarian *pattern*, maka awk akan memproses *action* pada tiap baris *input*.
e. Jika tidak terdapat proses *action* diberikan, maka data pada baris yang cocok dengan *pattern* akan ditampilkan pada layar sebagai *default action.*

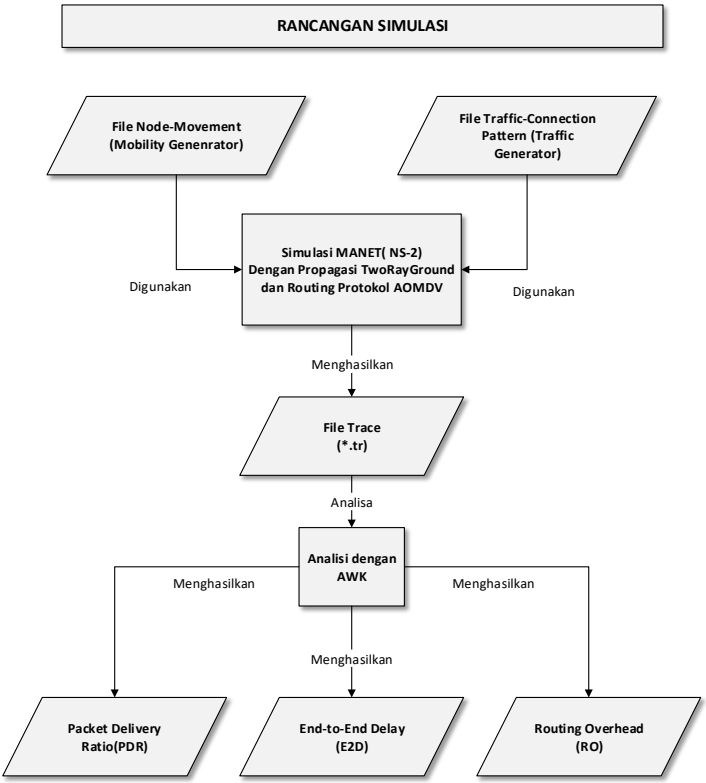
BAB III
PERANCANGAN

Pada Bab perancangan dijelaskan mengenai dasar perancangan dari perangkat lunak yang akan dibangun dalam Tugas Akhir. Secara khusus akan dibahas mengenai deskripsi umum sistem, peranangan skenario, serta gambaran implementasi sistem yang akan diterapkan pada *Network Simulator 2*.

3.1 Deskripsi Umum

Dalam Tugas Akhir ini dilakukan studi kinerja model propagasi *TwoRayGround* pada lingkungan jaringan MANET dengan protokol *routing Ad Hoc on Demand Multipah Distance Vector* (AOMDV) *routing*. Skenario simulasi MANET menggunkan *Mobility Generator* yang bersifat *Random Way Point* yang terdapat pada modul *Network Simulator 2* dengan cara melakukan *generate file node-movement (mobility generation)* dan membuat koneksi antar *node* menggunakan *file traffic-connection pattern* sebagaimana rancangan simulasi pada Gambar 3.1.

Skenario simulasi dilakukan dengan dua model skenario simulasi. *Model Mobility Node (*Pergerakan *Node)* dengan variasi pergerakan *node* yang memilliki kecepatan maksimal pergerakan sebesar 5, 10, dan 15 m/s, dan *Model Traffic Load (*Beban Trafik) dengan variasi jumlah beban trafik sebesar 10, 20, dan 30 koneksi. Kedua model skenario dilakukan simulasi sebanyak 10 kali pada *Netwok Simulator 2* sehingga dihasilkan *trace file* yang dapat dilakukan analisis kinerja dari model propagasi *TwoRayGround* pada MANET menggunakan protokol *routing* AOMDV.



Gambar 3.1 Tahapan Rancangan Simulasi

Analisis dilakukan pada hasil simulasi dari *trace file Network Simulator 2* (NS-2). Hasil analisis didapat dari pengolahan *trace file* simulasi dengan menggunakan skrip kode AWK sehingga didapat nilai metrik hasil analisis berupa *Paket Delivery Ratio* (PDR), *End-to-End Delay* (E2E), dan *Routing Overhead* (RO). Selain dari pada itu, dilakukan perbandingan hasil kinerja dengan model propagasi *Nakagami* untuk mendapatkan suatu perbandingan hasil kinerja.

3.2 Perancangan Skenario

Skenario simulasi pada Tugas Akhir ini dibuat dalam dua model skenario untuk mengetahui kinerja model propagasi *TwoRayGround* pada *protocol routing* AOMDV di lingkungan jaringan *Mobile Ad hoc Network* pada beberapa kondisi berbeda. Model skenario yang digunakan adalah sebagai berikut:

1. Skenario *Model Mobility Node* (Pergerakan *Node*) digunakan untuk mengetahui kinerja MANET dengan beberapa kondisi *mobilitas node* yang berbeda.
2. Skenario *Model Traffic Load* (Beban Trafik), untuk mengetahui kinerja MANET pada kondisi beban trafik yang berbeda.

Parameters simulasi *Network Simulator 2* yang akan digunakan pada Tugas Akhir ini adalah sebagaimana Tabel 3.1.

Tabel 3.1 Parameter Simulasi *Network Simulator 2*

No	Parameter	Spesifikasi
1	Jumlah *Node*	50 *Node*
2	Waktu Simulasi	200 detik
3	Area Simulasi	510 m x 510 m
4	Kecepatan Maksimal	- 5 m/s - 10 m/s - 15 m/s
5	Waktu Jeda	10 detik
6	Ukuran Paket	512 *bytes*

NO	Parameter	Spesifikasi
7	Tipe Koneksi	CBR
8	*Random seed*	1.0
9	Jumlah Maksimal Koneksi	- 10 Koneksi - 20 Koneksi - 30 Koneksi
10	*Packet Rate*	0.25
11	Agen	UDP
12	Model Mobilitas	*Random Way Point*

Perancangan skenario simulasi diawali dengan pembuatan skenario *file node-movement* yang bersifat *Random Way Point* dan kemudian dilanjutkan dengan membuat koneksi berupa *file traffic-connection pattern* menggunakan *tool* yang telah disediakan *Network Simulator 2*.

Penjelasan lebih detail mengenai model skenario yang akan digunakan adalah sebagaimana berikut:

3.2.1 Skenario *Model Mobility Node*

Skenario *Model Mobility Node* (Pergerakan *Node*) dibuat berdasarkan hasil *Mobility Generation* yang dilakukan dengan membuat *file node-movement* melalui *tool 'setsdets'* yang telah tersedia pada *Network Simulator 2*.

Skenario *Model Mobility Node* digunakan untuk mengetahui pengaruh pergerakan *node* pada kinerja MANET. *Mobility generation* dilakukan dengan variasi kecepatan maksimal pergerakan *node* yaitu dengan kecepatan maksimal 5

m/s, 10 m/s, dan 15 m/s. *File node-movement* akan dihasilkan dari *Mobility Generator* untuk setiap variasi kecepatan maksimal pergerakan *node*.

Untuk setiap variasi kecepatan maksimal dilakukan *generate file node-movement* sebanyak 10 kali dengan parameter lama waktu simulasi sebesar 200 detik pada area seluas 510 m x 510 m. Parameter lain yang digunakan untuk *Mobility Generator* dengan menggunakan *tool 'setdest'* sebagaimana Tabel 3.2.

Tabel 3.2 Parameter *Mobility Generator* untuk Skenario *Model Mobility Node*

No	Parameter	Spesifikasi
1	Versi	1
2	Jumlah *Node*	50 *Node*
3	Waktu Jeda	10 detik
4	Kecepatan Maksimal	- 5 m/s - 10 m/s - 15 m/s
5	Waktu Simulasi	200 detik
6	Area Simulasi	510 m x 510 m
7	Model Mobilitas	*Random Way Point*

Sebelum menjalankan simulasi pada *Network Simulator 2* selain dibutuhkan *file-node movement*, juga diperlukan *file traffic-connection pattern*. *File traffic-connection pattern* dihasilkan dengan menjalankan skrip *'cbrgen.tcl'* yang telah disediakan pada *Network Simulator 2* untuk digunakan menghubungkan antar *node* pada skenario saat simulasi dijalankan pada *Network Simulator 2*.

Untuk skenario *Model Mobility Node*, dibutuhkan satu bentuk koneksi yang akan digunakan pada setiap variasi kecepatan maksimal pergerakan *node*. Koneksi yang akan dibuat digunakan untuk menghubungkan *node* dengan tipe koneksi CBR.

Parameter koneksi untuk membuat suatu *file traffic-connection pattern* adalah sebagaimana Tabel 3.3.

Tabel 3.3 Parameter *Traffic Generator* untuk Skenario *Model Mobility Node*

No	Parameter	Spesifikasi
1	Tipe Koneksi	CBR
2	Jumlah Node	2
3	*Random seed*	1.0
4	Jumlah Maksimal Koneksi	1
5	*Packet Rate*	0.25

3.2.2 Skenario *Model Traffic Load*

Skenario *Model Traffic Load* (Beban Trafik) dibuat berdasarkan hasil *traffic generation* yang dilakukan dengan membuat *file traffic-connection pattern* menggunakan file *'cbrgen.tcl'* yang telah tersedia pada modul *Network Simulator 2*.

Skenario *Model Traffic Load* digunakan untuk mengetahui pengaruh beban trafik koneksi pada kinerja MANET. *Traffic generation* dilakukan dengan variasi jumlah koneksi maksimal yaitu dengan jumlah koneksi maksimal 10 buah, 20 buah, dan 30 buah. *File traffic-connection pattern* akan dihasilkan dari *Traffic Generator* utnuk setiap variasi jumlah maksimal koneksi.

Untuk setiap variasi jumlah koneksi maksimal dilakukan *generate file node-movement* sebanyak 10 kali dengan parameter lama waktu simulasi sebesar 200 detik pata area seluas 510 m x 510 m. Parameter yang digunakan untuk *Mobility Generator* dengan menggunakan *tool 'setdest'* sebagaimana Tabel 3.4.

Tabel 3.4 Parameter *Mobility Generator* untuk Skenario *Model Traffic Load*

No	Parameter	Spesifikasi
1	Versi	1
2	Jumlah *Node*	50 *Node*
3	Waktu Jeda	10 detik
4	Kecepatan Maksimal	10 m/s
5	Waktu Simulasi	200 detik
6	Area Simulasi	510 m x 510 m
7	Model Mobilitas	*Random Way Point*

Untuk menjalankan simulasi pada *Network Simulator 2* dengan menggunakan skenario *Model Traffic Load* selain dibutuhkan *file-node movement*, juga diperlukan *file traffc-connection pattern*. *File Traffic-Connection Pattern* dihasilkan dengan menjalankan program *'cbrgen.tcl'* yang telah ada pada *Network Simulator 2* untuk digunakan menghubungkan antar *node* pada saat simulasi dijalankan pada *Network Simulator 2.*

Untuk skenario *Model Traffic Load*, dibutuhkan tiga buah bentuk koneksi sesuai dengan variasi jumlah maksimal koneksi yang akan digunakan pada saat menjalankan simulasi. *File traffic-*

connection pattern yang akan dibuat digunakan untuk menghubungkan *node* dengan tipe koneksi CBR.

Dalam pembuatan *file traffic-connection pattern* dengan variasi jumlah maksimal koneksi 10 buah, 20 buah, dan 30 buah digunakan parameter sebagaimana Tabel 3.5.

Tabel 3.5 Parameter *Traffic Generation* untuk Skenario *Model Traffic Load*

No	Parameter	Spesifikasi
1	Tipe Koneksi	CBR
2	Jumlah *Node*	50
3	*Random seed*	1.0
4	Jumlah Maksimal Koneksi	- 5 Koneksi - 10 Koneksi - 15 Koneksi
5	*Packet Rate*	0.25

3.3 Perancangan Simulasi Pada *Network Simulator 2*

Pada perancangan skrip kode simulasi *Network Simulator 2* dilakukan penggabungan antara skenario *file node-movement* dengan *file traffic-connection pattern* beserta parameter-parameter untuk membangun suatu simulasi lingkungan jaringan MANET dengan model propagasi *TwoRayGround* dan *protocol routing* AOMDV pada *Network Simulator 2*. Berikut parameter simulasi perancangan lingkungan jaringan MANET dapat dilihat pada Tabel 3.6.

Tabel 3.6 Parameter Simulasi pada *Network Simulator 2*

No	Parameter	Spesifikasi
1	*Network Simulator*	NS-2.35
2	*Routing Protocol*	AOMDV
3	Waktu Simulasi	200 Detik
4	Waktu Pengiriman Paket Data	200 Detik
5	Area Simulasi	510 m x 510 m
6	Junlah *Node*	5, 25, 50
7	Radius Transmisi	100 m
8	Agen	UDP
9	Tipe Koneksi	*Connection Bit Rate (CBR)*
10	Kecepatan *Generate* Paket	1 Paket pet Detik
11	Ukuran Paket Data	512 *Bytes*
12	Protokol *Mac*	IEEE 802.11
13	Mode Transmisi	*TwoRayGround*
14	Tipe Antena	*OmniAntenna*
15	Tipe *Interface Queue*	*Droptail*
16	Tipe Kanal	*Wireless Channel*

No	Parameter	Spesifikasi
17	Tipe Peta	MANET (*Random Way Point*)
18	*Tipe Trace*	*Old Trace Format*

3.4 Perancangan Metrik Analisis

Metrik yang digunakan untuk analisis pada Tugas Akhir ini adalah *Packet Delivery Ratio* (PDR), *End-to-End Delay* (E2D), dan *Routing Overhead* (RO) dengan penjelasan masing-masing sebagaimana berikut:

3.4.2 *Packet Delivery Ratio* (PDR)

Packet Delivery Ratio (PDR) merupakan rasio antara jumlah total seluruh paket data yang suskes diterima, selanjutnya dibagi dengan jumlah total seluruh paket data yang dikirimkan [4].

Secara matematis *packet delivery ratio* dapat dijabarkan dengan Persamaan (2).

$$\text{PDR} = \frac{RecvNum}{SentNum}\ X\ 100\% \tag{2}$$

Packet Delivery Ratio (PDR) menunjukkan nilai keberhasilan suatu paket dikirimkan sampai ke tujuan. Semakin tinggi nilai dari PDR, maka semakin baik pula tingkat keberhasilan pengiriman suatu paket dilakukan.

3.4.3 *End-to-End Delay* (E2D)

End-to-end delay merupakan rata-rata *delay* atau waktu yang dibutuhkan suatu paket data untuk sampai pada tujuan. Dalam hal ini hanya paket data yang berhasil dikirim ke tujuan lah yang dihitung [13].

$$\text{E2D} = \frac{\sum_{n=1}^{RecvNum}(ReciveTime - SentTime)}{RecvNum} \tag{3}$$

Besar nilai *End-to-End Delay* dapat dihitung melalui Persamaan (3), di mana jumlah waktu *delay* dari setiap paket didapatkan dari rentang waktu antara *node* asal mengirimkan paket ($SentTime$) dan *node* tujuan menerima paket ($ReciveTime$) yang dibagi dengan jumlah paket yang berhasil diterima ($RecvNum$).

3.4.4 *Routing Overhead* (RO)

Routing Overhead merupakan jumlah paket kontrol *routing* yang ditransmisikan dari *node* asal ke *node* tujuan selama simulasi berjalan [13]. *Routing overhead* dihitung berdasarkan keseluruhan jumlah paket kontrol *routing* yang ditransmisikan, baik berupa *Packet Router Request* (RREQ), *Router Reply* (RREP), dan juga *Router Eror* (RERR).

$$\text{RO} = \sum_{n=1}^{SentNum\ and\ FwdNum} (SentNum) \tag{4}$$

Besar nilai *routing overhead* bergantung pada jumlah paket yang dikirimkan ($SentNum$) dan jumlah paket yang diteruskan ($FwdNum$) selama simulasi berjalan.

[*Halaman ini sengaja dikosongkan*]

BAB IV
IMPLEMENTASI

Bab ini membahas mengenai implementasi dari perancangan yang telah dibuat pada Bab sebelumnya yang meliputi lingkungan pembangunan baik perangkat lunak maupun perangkat keras, implementasi skenario simulasi, implmentasi simulasi pada NS-2, dan implementasi metrik analisis hasil simulasi.

4.1 Lingkungan Pembangunan Perangkat Lunak

Pengembangan perangkat lunak dilakukan pada lingkungan pengembangan sebagaimana berikut:

4.1.1 Lingkungan Perangkat Lunak

Perangkat lunak yang digunakan untuk pengembangan sistem adalah sebagai berikut:

- Sistem Operasi *Ubuntu 18.04 LTS 64 bit* untuk lingkungan *Network Simulator 2*.
- *Network Simulator 2 Versi 2.35.*

4.1.2 Lingkungan Perangkat Keras

Spesifikasi perangkat keras yang digunakan untuk implementasi sistem pada Tugas Akhir ini adalah sebagai berikut:

- *Processor Intel(R) Core(TM) i5-5200U CPU @ 2.20GHz 2.19 GHz;*
- Media penyimpanan sebesar 500 GB;
- RAM sebesar 4 GB DDR3.

4.2 Implementasi Skenario

Implementasi skenario simulasi MANET dilakaukan pada kondisi yang berbeda dalam mobilitas atau pergerakan dari *node* dan juga beban trafik-nya. Model skenario yang digunakan untuk mempelajari kinerja dari jaringan MANET adalah *Model Mobility Node* dan *Model Traffic Load*. Di mana, skenario *Model Mobility Node* digunakan untuk mempelajari pengaruh mobilitas dari *node* pada kinerja keseluruhan jaringan MANET. Sedangkan skenario *Model Traffic Load* digunakan untuk mempelajari pengaruh dari beban trafik pada jaringan MANET. Implementasi dari skenario *Model Mobility Node* dan *Model TraffIc Load* digunakan *file node-movement d*an *file traffic-connection pattern* dengan penjelasan sebagaimana berikut:

4.2.1 *File Node-Movement (Mobility Generation)*

Dalam implementasi skenario dari *Model Mobility Node*, digunakan *file node-movement* hasil dari *mobility generation* dengan *tool generate default* yang disediakan oleh *Network Simulator 2* yaitu *tool 'setdest'*. Untuk mempelajari efek mobilitas *node* pada jaringan MANET, maka simulasi dilakukan dengan pola variasi kecepatan maksimal yang berbeda.

File node-movement yang dihasilkan oleh *tool 'setdest'* menggunakan algoritma *Random Way Point*. Untuk melakukan *generate file node-movement*, format *command line* yang digunakan adalah sebagaimana Gambar 4.1.

```
Setdest    [-v  version]    [-n  number  of  nodes]
[-p pausetime] [-M maxspeed] [-t time] [-x maxX]
[-y maxY] > [outputfile/file-movement]
```

Gambar 4.1 Format *Command Line 'setdest'*

Ketentuan parameter uji coba yang digunakan untuk men-*generate file node-movement* berturut-turut adalah versi *'setdest'* simulator yaitu 1, jumlah *node* dalam simulasi ada 50 *node*, waktu jeda pergerakan *node (pause time)* yaitu 10 detik, luas area simulasi yaitu 510 meter x 510 meter, lama simulasi yaitu 200 detik, dan variasi kecepatan maksimal yang digunakan dalam skenario yaitu 5 m/s, 10 m/s, dan 15 m/s/

Command line untuk melakukan *generate file node - movement* dengan 3 buah variasi kecepetan maksimal yang berbeda sebesar 5 m/s, 10 m/s, dan 15 m/s adalah sebagaimana pada Gambar 4.2.

```
setdest  -v 1  -n 50  -p 10.0  -M 5.0  -t 200.0 -x
510  -y 510  >  scen-5-test

setdest  -v 1  -n 50  -p 10.0  -M 10.0 -t 200.0 -x
510  -y 510  >  scen-10-test

setdest  -v 1  -n 50  -p 10.0  -M 15.0 -t 200.0 -x
510  -y 510  >  scen-15-test
```

Gambar 4.2 Implementasi *Command Line Mobility Generation* pada 'setdest'

Untuk setiap variasi kecepatan, dilakukan *generate* 10 buah *file node-movement* di mana *file node-movement* hasil *generate* disimpan pada direktori *"~ ns-allinone-2.35- RC7\ns-2.35\indep-utils\cmu-scen-gen\setdest"* yang nantinya akan dijadikan skenario simulasi pada *Network Simulator 2* untuk skenario *Model Mobility Node.*

4.2.2 *File Traffic-Connection Pattern (Traffic Generation)*

Implementasi *traffic generator* untuk menghasilkan *file traffic-connection pattern* dilakukan menggunakan skrip *traffic generator* yang berada pada direktori *"~ns-allinone-2.35RC7\ns-2.35\indep-utils\cmu-scen-gen"* dengan nama *file 'cbrgen.tcl'.*

Dengan menggunakan skrip *'cbrgen.tcl'*, dapat diatur tipe koneksi yang digunakan apakah menggunakan tipe CBR atau tipe TCP. Selain itu dapat juga dapat dilakukan pengaturan berapa jumlah *node* dan koneksi maksimal yang dihasilkan, *random seed*, serta nilai *transfer rate*. Format *command line* untuk melakukan pengaturan parameter dari *traffic generator* adalah sebagaiman ditunjukkan pada Gambar 4.3.

```
Setdest    [-v  version]    [-n  number  of  nodes]
[-p pausetime] [-M maxspeed] [-t time] [-x maxX]
[-y maxY] > [outputfile/file-movement]
```

Gambar 4.3 Format *Command Line cbrgen.tcl*

Ketentuan parameter uji coba untuk implementasi *file traffic-connection pattern* dengan variasi jumlah koneksi maksimal 10, 20, dan 30 buah koneksi untuk skenario *Model Traffic Load* adalah sebagaimana pada Gambar 4.4.

```
ns  cbrgen.tcl -type cbr  -nn 50  -seed 1.0 -mc 10
-rate 0.25  >  cbr-10-test

ns  cbrgen.tcl -type cbr  -nn 50  -seed 1.0 -mc 20
-rate 0.25  >  cbr-20-test

ns  cbrgen.tcl -type cbr  -nn 50  -seed 1.0 -mc 30
-rate 0.25  >  cbr-30-test
```

Gambar 4.4 Implementasi *Command Line Traffic Generation* pada *cbrgen.tcl*

Eksekusi dari *command line* di atas akan menghasilkan tiga buah *file traffic-connection pattern* yang nantinya akan digunakan dalam skenario *Model Traffic Load* pada saat simulasi *Network simulator 2*.

4.3 Implementasi Simulasi pada *Network Simulator 2*

Implementasi awal dari simuasi pada *Network Simulator 2* adalah dengan melakukan pengaturan *file node-movement* dan *file traffic-connection pattern* yang digunakan dalam simulasi. Selain dari pada itu, dilakukan pengaturan parameter-parameter yang digunkan. Pada Gambar 4.5, menunjukkan skrip konfigurasi awal parameter-parameter yang digunakan untuk menjalankan simulasi MANET pada *Network Simulator 2* yang terdiri dari parameter untuk konfigurasi tipe *channel* berupa *Wireless Channel* pada baris pertama, kemudian pada baris kedua adalah parameter untuk tipe transmisi yang digunakan yaitu model transmisi *TwoRayGround.* Tipe *Mac* yang digunakan adalah *Mac 802.11*.

```
set val(chan) Channel/WirelessChannel
set val(prop) Propagation/TwoRayGround
set val(netif) Phy/WirelessPhy
set val(mac) Mac/802_11
set val(ifq) Queue/DropTail/PriQueue
set val(ll) LL
set val(ant) Antenna/OmniAntenna
set val(x) 510 ;# X dimension of the topography
set val(y) 510 ;# Y dimension of the topography
set val(ifqlen) 50 ;# max packet in ifq
set val(seed) 1.0
set val(adhocRouting) AOMDV
set val(nn) 50 ;# how many nodes are simulated
set val(cp) "cbr-10-test"
set val(sc) "scen-15-test"
set val(stop) 200 ;# simulation time
```

Gambar 4.5 Konfigurasi Awal Parameter pada *Network Simulator 2*

```
# Initialize Global Variables
# create simulator instance
set ns_ [new Simulator]
# setup topography object
set topo [new Topography]
# create trace object for ns and nam
set tracefd [open 5-1.tr w]
set namtrace [open 5-1.nam w]
$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace $val(x)
$val(y)
# define topology
$topo load_flatgrid $val(x) $val(y)
# Create God
set god_ [create-god $val(nn)]
#global node setting
set chan [new $val(chan)]
$ns_ node-config -adhocRouting $val(adhocRouting)\
       -llType $val(ll) \
       -macType $val(mac) \
       -ifqType $val(ifq) \
       -ifqLen $val(ifqlen) \
       -antType $val(ant) \
       -propType $val(prop) \
       -phyType $val(netif) \
       -channel $chan \
       -topoInstance $topo \
       -agentTrace ON \
       -routerTrace ON \
       -macTrace OFF

# Create the specified number of nodes [$val(nn)]
and "attach" them to the channel.
for {set i 0} {$i < $val(nn) } {incr i} {
set node_($i) [$ns_ node]
$node_($i) random-motion 0 ;# disable random
motion}
```

Gambar 4.6 Konfigurasi *Trace File, NAM,* dan *Node-Movement* pada *Network Simulator 2*

Skrip pada Gambar 4.6 merupakan bentuk pengaturan variabel global yang diawali dengan pembuatan simulasi baru dengan kode skrip s*et ns_*. Kode skrip *set tracefd* dan *set nametrace* berfungsi untuk pengaturan *trace file* dengan ekstensi (*.tr) yang akan menghasilkan data hasil simulasi selama simulasi berjalan. Sedangkan *file network animator* akan menghasilkan suatu *file* berekstensi (*.nam) yang berguna untuk menyimpan data hasil simulasi dengan format animasi yang bisa dijalankan dengan program animasi *Network Animator* atau NAM.

Trace file hasil simulasi memiliki dua buah format yaitu *old trace format* dan *new trace format* yang bisa dilakukan pengaturan pada saat inisialisasi variable grobal *set tracefd*. Pada Tugas Akhir ini, format *trace file* yang digunakan adalah *old trace format.*

Konfigurasi objek topografi dilakukan pada skip kode *set topo* berdasarkan pada luas koordinat yang telah diinisialisasi pada parameter awal. *Create-got* dan *node-config* merupakan konfigurasi dari setiap *node* yang akan dibuat. Pada *create-got*, dilakukan implementasi dari parameter *setval(nn)* pada setiap *node* yang dibuat.

Bagian akhir dari skip kode pada Gambar 4.6 digunakan untuk melakukan pergerakan pada *node* secara berulang. Pergerakan *node* diambil dari *file node-movement* hasil dari *mobility generator.*

Kode skrip pada Gambar 4.7 merupakan bagian akhir dari keseluruhan skrip, di mana pada bagian ini dilakukan pembacaan skenario *file node-movement* untuk melakukan konfigurasi pergerakan *node* dan juga pembacaan *file traffic-connection* untuk konfigurasi trafik. Selanjutnya dilakuakn pengiriman paket dari detik ke-0 sampai dengan simulasi berakhir pada detik ke-200.

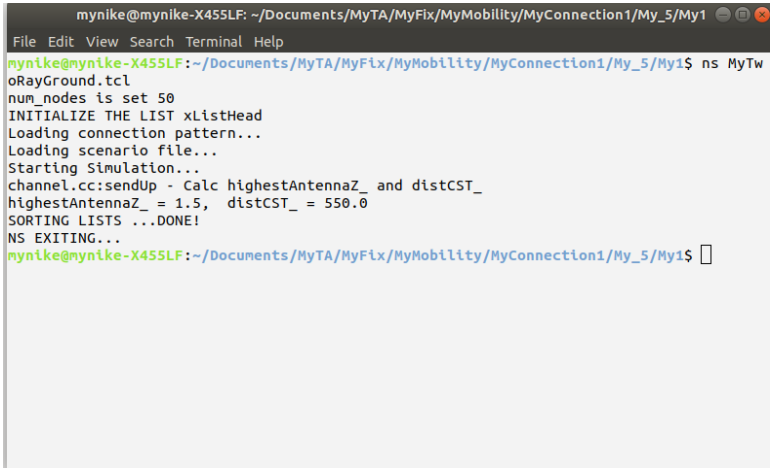
```
# Define node movement model
puts "Loading connection pattern..."
source $val(sc)
# Define traffic model
puts "Loading scenario file..."
source $val(cp)
# Define node initial position in nam
for {set i 0} {$i < $val(nn)} {incr i} {
# 20 defines the node size in nam, must adjust it
according to your scenario
# The function must be called after mobility model
is defined
$ns_ initial_node_pos $node_($i) 50
}
# Tell nodes when the simulation ends
for {set i 0} {$i < $val(nn) } {incr i} {
$ns_ at $val(stop).0 "$node_($i) reset";
}
$ns_ at $val(stop).0002 "puts \"NS EXITING...\" ;
$ns_ halt"
puts $tracefd "M 0.0 nn $val(nn) x $val(x) y
$val(y) rp $val(adhocRouting)"
puts $tracefd "M 0.0 sc $val(sc) cp $val(cp) seed
$val(seed)"
puts $tracefd "M 0.0 prop $val(prop) ant
$val(ant)"
puts "Starting Simulation..."

proc stop {} {
global ns_tracefd
namtrace
$ns_ flush-trace
close $tracefd
close $namtrace
}

$ns_ run
```

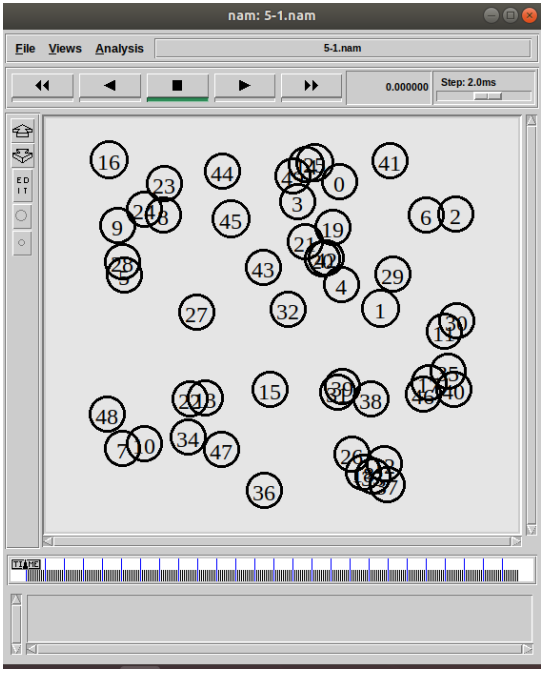
Gambar 4.7 Konfigurasi Pengiriman Paket pada *Network Simulator 2*

Pada Gambar 4.8, merupakan contoh *running*/eksekusi suatu simulasi jaringan MANET dengan menggunakan model propagasi *TwoRayGround* dan *Routing* AOMDV dengan jumlah node sebanyak 50 node yang bersifat acak dengan algoritma *Random Way Point* dengan menggunakan salah satu model skenario yang telah di-*generate* sebelumnya.



Gambar 4.8 *Runnig* Skrip (*.tcl) dengan Model Propagasi *TwoRayGround*

Simulasi *Network Simulator 2* dilakukan dengan running/eksekusi *file* (*.tcl) yang akan menghasilkan dua buah *file* berupa *trace file* (*.tr) yang berisikan data hasil simulasi dan sebuah *file* (*.nam) yang dapat dijalankan dengan menggunakan aplikasi *Network Animator* untuk mendapatkan visualisasi jalannya simulasi.



Gambar 4.9 Visualisasi Hasil Simulasi pada Aplikasi Nam

Gambar 4.9, merupakan hasil visualisasi dari *file* (*.nam) yang dijalankan dengan menggunakan aplikasi *Network Animator*. Di mana pada visualisasi animasi yang dijalankan dengan aplikasi *Network Animator* untuk suatu jaringan MANET akan menampilkan sejumlah *node* sesuai dengan *file-node movement* yang digunakan beserta pergerakannya. Selain dari pada itu ditampilkan animasi koneksi anatra *node* pada saat pengiriman paket data sesuai *file traffic-connection pattern.*

4.4 Implementasi Metrik Analisis

Hasil simulasi MANET dengan *Network Simulator 2* memiliki bentuk *trace file* dengan ekstensi (*.tr). *Trace file* dianalisis dengan menggunakan tiga metrik analisis yaitu *Packet Delivery Ratio* (PDR), *End-to-End Delay* (E2D), dan *Routing Overhead* (RO). Ketiga metrik analisis diimplementasikan dengan menggunakan bahasa pemrograman AWK sebagaimana dijelaskan pada tiap-tiap subbab berikut:

4.4.1 *Packet Delivery Ratio* (PDR)

Proses perhitungan *Packet Delivery Ratio* dilakukan dengan menghitung rasio perbandingan jumlah paket data yang diterima oleh *node* tujuan dan jumlah paket yang dikirimpakn oleh *node* asal.

```
ALGORITMA PDR (trace file)
INPUT  : Trace file hasil simulasi
OUTPUT : Jumlah paket terkirim, paket diterima,
         dan nilai PDR

BEGIN(
SentNum ← 0
RecvNum ← 0
PDR ← 0)
(
if (($1 == "s") and ($4 == "AGT") and ($7 ==
         "cbr") SentNum + 1;
if (($1 == "r") and ($4 == "AGT") and ($7 ==
         "cbr") RecvNum + 1;
)
EDN(
PDR ← ( RecvNum/SentNum ) * 100
Print SentNum
Print RecvNum
Print PDR
)
```

Gambar 4.10 *Pseudecode Packet Delivery Ratio* (PDR)

Sebagaimana *psedeucode* PDR yang ditunjukkan pada Gambar 4.10, penambahan jumlah paket data yang terkirim dilakukan apabila pada baris data *trace file* mengandung kondisi di mana pada kolom pertama berupa huruf "s" yang menandakan *packet sent* (paket terkirim), pada kolom ke-4 dan ke-7 menunjukkan "AGT" dan "cbr" yang menandakan bahwa pengiriman paket yang dilakukan adalah berupa paket data dengan tipe koneksi CBR (*Connection Bit Rate).*

Penambahan jumlah peket data diterima dilakukan apabila pada baris *trace file* ditemukan kondisi bahwa kolom pertama berupa huruf "r" yang menandakan *packet recived* (paket diterima), pada kolom ke-4 dan ke-7 menunjukkan "AGT" dan "cbr" yang menandakan bahwa pengiriman paket yang dilakukan adalah berupa paket data dengan tipe koneksi CBR *(Connection Bit Rate).* Perhitungan jumlah paket diterima dan paket dikirimkan dilakukan sampai dengan akhir baris *trace file*. Nilai *Packet Delivery Ratio* didapat dari persentase hasil bagi antara paket data yang diterima dan paket data yang dikirimkan.

Implementasi *psedeucode* dari *Packet Delivery Ratio* (PDR) dilakukan menggunakan bahasa pemrograman AWK. Di mana contoh perintah untuk menjalankan analisis PDR pada suatu *trace file* adalah sebagaimana ditunjukkan dengan skrip di bawah:

```
Awk –f PDR.awk trace_file.tr
```

Contoh hasil analisis yang didapat dari eksekusi *command line* di atas adalah sebagaimana ditunjukkan pada Gambar 4.11.

```
#Hasil Packet Delivery Ratio

Jumlah Paket Dikirim = 52
Jumlah Paket Diterima = 46
Packet Delivery Ratio (PDR) = 88.462 %
```

Gambar 4.11 Contoh Hasil *Running* Skrip PDR.awk

4.4.2 *End-to-End Delay* (E2D)

Perhitungan *End-to-End Delay* (E2D) dilakukan dengan menghitung selisih antara waktu data terkirim dari *node* asal dan waktu peket data diterima oleh *node* tujuan. Waktu paket terkirimkan dicatat dalam *trace file* pada kolom ke-2 dari setiap baris.

Penambahan jumlah paket data yang terkirim dilakukan apabila pada baris data *trace file* mengandung kondisi di mana pada kolom pertama berupa huruf "s" yang menandakan *packet sent* (paket terkirim), pada kolom ke-4 dan ke-7 menunjukkan "AGT" dan "cbr" yang menandakan bahwa pengiriman paket yang dilakukan adalah berupa paket data dengan tipe koneksi CBR *(Connection Bit Rate).*

Penambahan jumlah peket data diterima dan pencatatan ID paket data yang diterima dilakukan ababila pada baris *trace file* ditemukan kondisi bahwa kolom pertama berupa huruf "r" yang menandakan *packet recived* (paket diterima), pada kolom ke-4 dan ke-7 menunjukkan "AGT" dan "cbr" yang menandakan bahwa pengiriman packet yang dilakukan adalah berupa paket data dengan tipe koneksi CBR *(Connection Bit Rate).*

Pencatatan waktu peket terkirim dilakukan apabila pada baris *trace file* ditemukan kondisi pada kolom pertama berupa huruf "s" yang menandai *packet sent* (paket dikirimkan), pada kolom ke-4 dan ke-7 menunjukkan "AGT" dan "cbr" yang menandakan bahwa pengiriman paket yang dilakukan adalah berupa paket data dengan tipe koneksi CBR *(Connection Bit Rate).*

Pencatatan waktu peket diterima dilakukan apabila pada baris *trace file* ditemukan kondisi pada kolom pertama berupa huruf "r" atau huruf "D" yang menandai *packet recived* (paket diterima), pada kolom ke-4 dan ke-7 menunjukkan "AGT" dan "cbr" yang menandakan bahwa pengiriman paket yang dilakukan adalah berupa paket data dengan tipe koneksi CBR *(Connection Bit Rate).*

```
ALGORITMA PDR (trace file)
INPUT  : Trace file hasil simulasi
OUTPUT : Jumlah paket terkirim, paket diterima,
       dan nilai E2D

BEGIN(
Seqno ← -1 , SentNum ← 0, RcvNum ← 0,
Delay ← 0, Count ← 0, E2D ← 0)
(
if (($1 == "s") and ($4 == "AGT") and ($7 ==
       "cbr") SentNum + 1;

if (($1 == "s") and ($4 == "AGT") and (Seqno < $6)
       Seqno = $6;
else if (($1 == "r") and ($4 == "AGT") and ($7 ==
       "cbr") RcvNum + 1;

if (($1 == "s") and($4 == "AGT") and ($7 ==
       "cbr")) star_time[$6] = $2;
else if (($1 == "r") and ($7 == "cbr"))
       end_time[$6] = $2;
else if (($1 == "D") and ($7 == "cbr"))
       end_time[$6] = -1;
)
END(
for i in Seqno
       if (end_time[i] > 0)
              delay[i] = end_time[i] -
              start_time[i];
              Count + 1;
       else
              delay[i] = -1;
for i in seqno
       if (delay[i] > 0)
              Delay = Delay + delay[i];
E2D ← Delay/RecvNum
Print SentNum
Print RecvNum
Print E2D
)
```

Gambar 4.12 ***Pseudecode End-to-End Delay*** **(E2D)**

Perhitungan jumlah paket diterima, paket dikirimkan, pencatatan ID dari paket yang dikirimkan, pencatatan waktu paket dikirim, dan pencatatan waktu data diterima dilakukan sampai dengan akhir baris *trace file*. Nilai *End-to-End Delay* didapat dari rata-rata *delay* selisih waktu antara paket data dikirimkan dan waktu paket data diterima sebagaimana ditunjukkan pada Gambar 4.12.

Implementasi *psedeucode* dari *End-to-End Delay* dilakukan menggunakan bahasa pemrograman AWK. Di mana contoh perintah untuk menjalankan analisis E2D pada suatu *trace file* adalah sebagaimana ditunjukkan dengan skrip di bawah:

```
Awk –f E2D.awk trace_file.tr
```

Contoh hasil analisis yang didapat dari eksekusi *command line* di atas adalah sebagaimana ditunjukkan pada Gambar 4.13.

```
#Hasil End-to-End Delay

Jumlah Paket Dikirim = 52
Jumlah Paket Diterima = 46
End-to-End Delay = 17.814 milisecond
```

Gambar 4.13 Contoh Hasil *Running* Skrip E2D.awk

4.4.3 *Routing Overhead* (RO)

Implementasi perhitungan metrik *Routing Overhead* sebagaimana ditunjukkan pada Gambar 4.14. Di mana nilai *Routing Overhead* didapat dari jumlah keseluruhan paket yang ditransmisikan pada saat simulasi baik paket data yang dikirimkan (*sent packet*) ataupun paket data yang diteruskan (*forward packet*).

```
ALGORITMA RO (trace file)
INPUT  : Trace file hasil simulasi
OUTPUT : Jumlah Routing Overhead

BEGIN(
RoNum ← 0)
(
if (((($1 == "s") or ($1 == "f")) and ($4 == "RTR")
       and ($7 == "cbr")RoNum + 1;
)
END(
Print RoNum
)
```

Gambar 4.14 Pseudecode *Routig Overhead* (RO)

Penambahan jumlah paket data yang ditransmisikan dilakukan apabila pada baris data *trace file* ditemukan kondisi kolom pertama berupa huruf "s" yang menandakan paket data dikirimkan (*sent packet*) atau huruf "f" yang menandakan paket data diteruskan (*forward packet*), pada kolom ke-4 menunjukkan "RTR" yang menandakan *routing packet*, dan pada kolom ke-7 menunjukkan "cbr" yang menandakan bahwa pengiriman paket yang dilakukan adalah berupa paket data dengan tipe koneksi CBR *(Connection Bit Rate)*. Perhitungan jumlah paket yang dikirimkan dan diteruskan dilakukan sampai dengan akhir baris *trace file.*

Implementasi *psedeucode* dari *Routing Overhead* dilakukan menggunakan bahasa pemrograman AWK. Di mana contoh perintah untuk menjalankan analisis RO pada suatu *trace file* adalah sebagaimana ditunjukkan dengan skrip di bawah:

```
Awk –f RO.awk trace_file.tr
```

Contoh hasil analisis yang didapat dari eksekusi *command line* di atas adalah sebagaimana ditunjukkan pada Gambar 4.15.

```
#Hasil Routing Overhead

Routing Overhead = 90
```

Gambar 4.15 Hasil *Running* Skrip RO.awk

[*Halaman ini sengaja dikosongkan*]

BAB V
PENGUJIAN DAN EVALUASI

Bab ini membahas mengenai pengujian dan evaluasi dari skenario simulasi pada *Network Simulator 2*. Pengujian dilakukan dengan menggunakan beberapa variasi skenario pengujian.

5.1. Lingkungan Pengujian

Pengujian simulasi dilakukan pada laptop dengan dua buah sistem operasi yaitu sistem operasi *Windows* dan *Linux*. Spesifikasi lebih lengkap ditunjukkan pada Tabel 5.1.

Tabel 5.1 Spesifikasi Laptop yang Digunakan

Komponen	Spesifikasi
CPU	*Processor Intel(R) Core(TM) i5-5200U CPU @ 2.20GHz 2.19 GHz;*
Sistem Operasi	*Windows 10 Pro 64 bit, Linux Ubuntu 18.04 LTS 64 bit (NS-2, Mobility Generation, Traffic Connection Pattern, TwoRayGround)*
Memori	4 GB DDR3
Media Penyimpanan	500 GB

Network Simulator untuk simulasi terpasang pada sistem operasi *Linux*. Versi yang digunakan untuk simulasi Tugas Akhir adalah *Network Simulator 2.35*.

5.2. Kriteria Pengujian

Pengujian simulasi MANET pada *Network Simulator 2* dilakukan dengan menggunakan skenario *Model Mobility Node* sesuai dengan *file node-movement* yang dihasilkan oleh *mobility generator* dan skenario *Model Traffic Load* sesuai dengan *file traffic-connection pattern.* Kriteria uji coba simulasi ditunjukkan pada Tabel 5.2.

Tabel 5.2 Kriteria Pengujian

No	Kriteria	Spesifikasi
1	Skenario	MANET*(Model Mobility Node, Model Traffic Load.)*
2	Kecepatan Maksimal Perpindahan *Node*	- 5 m/s - 10 m/s - 15 m/s
3	Jumlah koneksi maksimal	- 10 Koneksi - 20 Koneksi - 30 Koneksi
4	Jumlah Percobaan	10 Kali
5	Posisi & Pergerakan *Node*	Acak
6	Protokol *Ruting*	AOMDV
7	Model Mobilitas	*Random Way Point*

Uji coba simulasi dilakukan sebanyak 10 kali untuk tiap model skenario, baik skenario *Model Mobility Node* dengan variasi kecepatan maksimal pergerakan *node* dan juga skenario *Model Traffic Load* dengan variasi jumlah koneksi maksimal.

Protokol *routing* yang digunakan pada simulasi MANET adalah protokol *routing Ad Hoc on Demand Multipah Distance Vector* (AOMDV) yang merupakan *reactive routing protocol* dengan model propagasi *TwoRayGround.*

5.3. Analisis Hasil Uji Coba

Trace file hasil dari simulasi skenario MANET dengan protokol *routing* AOMDV menggunakan model propagasi *TwoRayGround* pada *Network Simulator 2* selanjutnya dilakukan analisis hasil menggunakan metrik *Packet Delivery Ratio* (PDR), *End-to-End Delay* (E2D), dan *Routing Overhead* (RO) untuk mendapatkan hasil analisis kinerja. Selanjutnya dilakukan perbandingan hasil kinerja dengan model propagasi *Nakagami* utnuk mengetahui perbandingan kinerja dari model propagasi *TwoRayGround*.

Penjelasan lebih lengkap mengenai metrik analisis hasil simulasi dijelaskan pada subbab berikut:

5.2.1 Skenario *Model Mobility Node*

Untuk mendapatkan hasil analisis pengaruh mobilitas *node*, digunakan skenario *Model Molility Node* dengan variasi kecepatan maksimal pergerakan *node* sebesar 5 m/s, 10 m/s, dan 15 m/s. Dengan jumlah *node* sebesar 50 *node* dan jumlah maksimal koneksi adalah satu koneksi. Hasil simulasi dari skenario *Model Mobility Node* dilakukan analisis menggunakan metrik analisis sebagai berikut:

5.2.1.1 Analisis *Packet Delivery Ratio* (PDR)

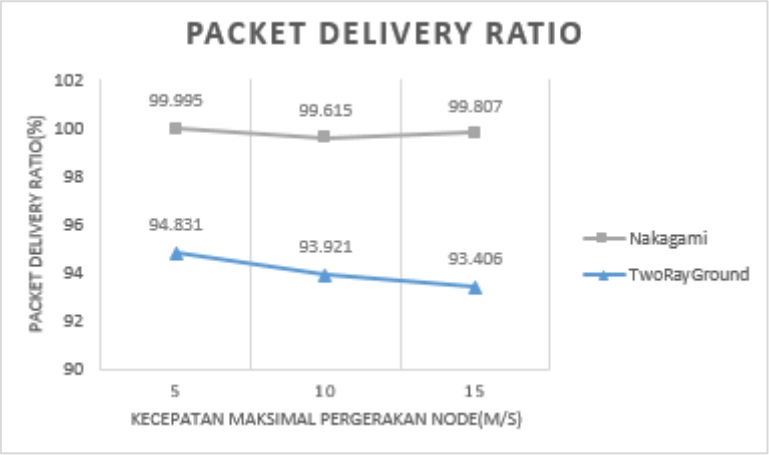
Hasil analisis *Packet Delivery Ratio* dari *trace file* untuk tiap simulasi dari skenario *Model Mobility Node* ditabulasikan dan diambil rata-rata dengan hasil sebagaimana pada Tabel 5.3.

Tabel 5.3 Hasil Nilai *Packet Delivery Ratio*

Kecepatan Maksimal Perpindahan *Node* (m/s)	**PDR (%)**	
	TwoRayGround	**Nakagami**
5	94.831	99.995
10	93.921	99.615
15	93.406	99.807

Tabel 5.3 dan Gambar 5.1 menunjukkan performa *Packet Delivery Ratio* hasil simulasi *Network Simulator 2* dari Model propagasi *TwoRayGround* pada jaringan MANET menggunakan protokol *routing Ad Hoc on Demand Multipah Distance Vector* (AOMDV) dengan skenario *Model Mobility Node*. Terlihat bahwa besar nilai rata-rata dari *Packet Delivery Ratio* (PDR) yang dihasilkan memiliki kecenderungan semakin menurun seiring dengan pertambahan kecepatan maksimal pergerakan *node.*

Packet Delivery Ratio (PDR) pada model propagasi *TwoRayGround* menunjukkan nilai 94.831% pada saat kecepatan maksimal pergerakan *node* 5 m/s, pada kecepatan maksimal pergerakan *node* 10 m/s nilai PDR sebesar 93.921 %, dan pada saat kecepatan maksimal pergerakan *node* 15 m/s nilai PDR adalah sebesar 93.406 %. Hasil nilai *Packet Delivery Ratio* (PDR) mengalami penurunan sebesar 0.96% pada saat kecepatan maksimal 5 m/s ke kecepatan maksimal 10 m/s. Dan kembali mengalami penurunan sebesar 0.54% pada saat kecepatan maksimal 10 m/s ke kecepatan maksimal 15 m/s.



Gambar 5.1 Grafik Nilai *Packet Delivery Ratio* terhadap Kecepatan Maksimal *Perpindahan Node*

Bila dibandingan dengan model propagasi *Nakagami* yang memiliki nilai *Packet Delivery Ratio* relatif stabil, hasil nilai *Packet Delivery Ratio* pada model propagasi *TwoRayGround* memiliki kecenderungan menurun seiring dengan bertambahnya kecepatan maksimal pergerakan *node*. Nilai PDR dari model propagasi *TwoRayGround* memiliki nilai yang lebih rendah dari pada nilai PRD dari model propagasi *Nakagami*, di mana nilai PDR dari model propagasi *Nakagami* mencapai nilai 99.995% pada kecepatan maksimal pergerakan *node* 5 m/s, pada kecepatan maksimal pergerakan *node* 10 m/s nilai PDR mencapai 99.615 %, dan pada kecepatan maksimal pergerakan *node* 15 m/s nilai PDR dari model propagasi *Nakagami* mencapai 99.807 %.

Penurunan performa dari *Packet Delivery Ratio* (PDR) pada model propagasi *TwoRayGround* terjadi seiring dengan banyaknya paket *drop* yang terjadi pada saat pengiriman paket. Hal ini diakibatkan karena terputusnya rute pengiriman paket yang terjadi lebih banyak seiring dengan kecepatan maksimal pergerakan *node* yang semakin besar.

5.2.1.2 Analisis *End-to-End Delay* (E2D)

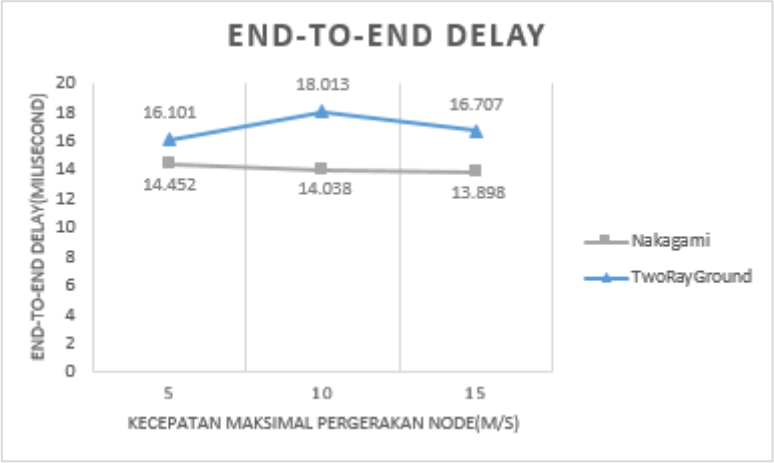
Hasil analisis *End-to-End Delay* (E2D) dari *trace file* untuk tiap simulasi dari skenario *Model Mobility Node* ditabulasikan dan diambil rata-rata dengan hasil sebagaimana pada Tabel 5.4.

Tabel 5.4 Hasil Nilai *End-to-End Delay*

Kecepatan Maksimal Perpindahan *Node* (m/s)	**E2D (Milisecond)**	
	TwoRayGround	**Nakagami**
5	16.101	14.452
10	18.013	14.038
15	16.707	13.898

Tabel 5.4 dan Gambar 5.2, menunjukkan nilai *End-to-End Delay* dari model propagasi *TwoRayGround* pada jaringan MANET menggunakan protokol *routing Ad Hoc on Demand Multipah Distance Vector* (AOMDV) dengan skenario *Model Mobility Node.*

Besar nilai *End-to-End Delay* (E2D) pada model propagasi *TwoRayGround* mengalami kenaikkan sebesar 1.91 *milisecond* pada perubahan kecepatan maksimal pergerakan *node* dari kecepatan maksimal 5 m/s ke kecepatan maksimal pergerakan *node* 10 m/s. Sedangkan pada perubahan kecepatan maksimal pergerakan *node* dari kecepatan maksimal 10 m/s ke kecepatan maksimal pergerakan *node* 15 m/s, nilai *End-to-End Delay* mengalami penurunan sebesar 1.31 *milisecond*.



Gambar 5.2 Grafik Nilai *End-to-End Delay* terhadap Kecepatan Maksimal Perpindahan *Node*

Bila dibandingan dengan model propagasi *Nakagami* yang memiliki nilai *End-to-End Delay* yang relatif stabil, nilai *End-to-End Delay* pada model propagasi *TwoRayGround* memiliki kecenderungan fluktuatif seiring dengan bertambahnya kecepatan maksimal pergerakan *node*. Nilai E2D dari model propagasi *TwoRayGround* memiliki nilai yang sedikit lebih besar dari pada nilai E2D dari model propagasi *Nakagami*, di mana nilai E2D dari model propagasi *Nakagami* sebesar 14.452 *milisecond* pada kecepatan maksimal pergerakan *node* 5 m/s, pada kecepatan maksimal pergerakan *node* 10 m/s nilai E2D sebesar 14.038 *milisecond*, dan pada kecepatan maksimal pergerakan *node* 15 m/s nilai E2D dari model propagasi *Nakagami* mencapai 13.898 *milisecond*.

Nilai *End-to-End Delay* pada model propagasi *TwoRayGround* mengalami perubahan secara fluktuatif diakibatkan karena pergerakan *node* yang dinamis sehingga memungkinkan terjadi rute putus dan antrian saat pengiriman paket berlangsung lebih besar.

5.2.1.3 Analisis *Routing Overhead* (RO)

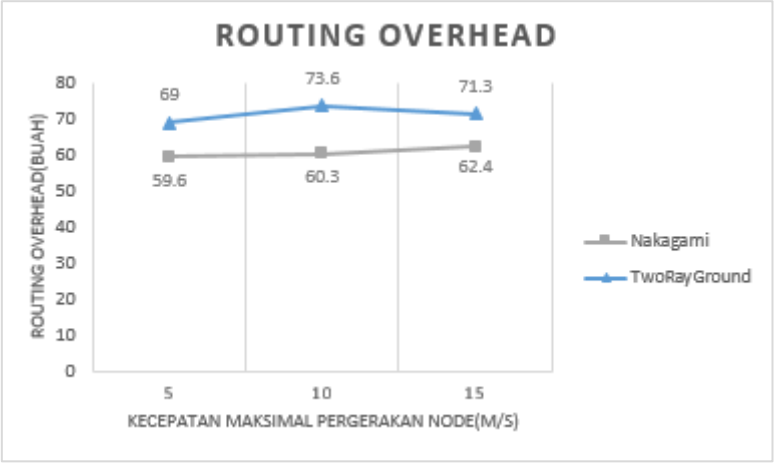
Hasil analisis *Routing Overhead* (RO) dari *trace file* untuk tiap simulasi dari skenario *Model Mobility Node* ditabulasikan dan diambil rata-rata dengan hasil sebagaimana pada Tabel 5.5.

Tabel 5.5 Hasil Nilai *Routing Overhead*

Kecepatan Maksimal Perpindahan *Node* (m/s)	**RO (Buah)**	
	TwoRayGround	**Nakagami**
5	69.0	59.6
10	73.6	60.3
15	71.3	62.4

Tabel 5.5 dan Gambar 5.3 menunjukkan nilai *Routing Overhead* dari Model propagasi *TwoRayGround* pada jaringan MANET menggunakan protokol *routing Ad Hoc on Demand Multipah Distance Vector (*AOMDV) dengan skenario *Model Mobility Node*

Besar nilai *Routing Overhead* pada model propagasi *TwoRayGround* mengalami kenaikkan sebesar 6.67 % atau sebanyak 4.6 paket pada perubahan kecepatan maksimal pergerakan *node* dari kecepatan maksimal 5 m/s ke kecepatan maksimal pergerakan *node* 10 m/s. Sedangkan pada perubahan kecepatan maksimal pergerakan *node* dari kecepatan maksimal 10 m/s ke kecepatan maksimal pergerakan *node* 15 m/s, nilai *Routing Overhead* mengalami penurunan paket sebanyak 2.3 atau sebesar 3.12 %.



Gambar 5.3 Grafik Nilai *Routing Overhead* terhadap Kecepatan Maksimal Perpindahan *Node*

Bila dibandingan dengan model propagasi *Nakagami*, nilai *Routing Overhead* pada model propagasi *TwoRayGround* memiliki kesamaan yang cenderung memiliki nilai yang fluktuatif seiring dengan bertambahnya kecepatan maksimal pergerakn *node*. Nilai RO dari model propagasi *TwoRayGround* memiliki nilai yang sedikit lebih banyak dari pada nilai RO dari model propagasi *Nakagami*, di mana nilai RO dari model propagasi *Nakagami* sebanyak 59.6 pada kecepatan maksimal pergerakan *node* 5 m/s, pada kecepatan maksimal pergerakan *node* 10 m/s nilai RO sebanyak 60.3, dan pada kecepatan maksimal pergerakan *node* 15 m/s nilai RO sebanyak 62.4.

Nilai *Routing Overhead* pada model propagasi *TwoRayGround* mengalami perubahan secara fluktuatif diakibatkan karena pergerakan *node* yang dinamis sehingga memungkinkan terjadi rute putus pada saat pengiriman paket berlangsung. Hal ini mengakibatkan jumlah *routing* paket yang dihasilkan (paket *routing send* dan *forward*) menjadi lebih banyak untuk membuat rute baru pengiriman paket.

5.2.2 Skenario *Model Traffic Load*

Untuk mendapatkan hasil analisis pengaruh beban trafik, digunakan skenario *Model Traffic Load* dengan variasi beban maksimal koneksi sebesar 10 koneksi, 20 koneksi, dan 30 koneksi. Dengan jumlah *node* sebesar 50 *node*. Hasil simulasi dari skenario *Model Traffic Load* dilakukan analisis menggunakan metrik analisis sebagai berikut:

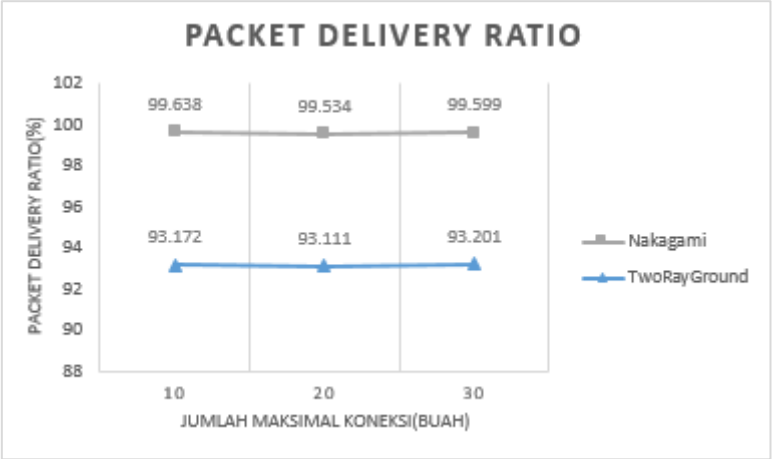
5.2.2.1 Analisis *Packet Delivery Ratio* (PDR)

Hasil analisis *Packet Delivery Ratio (PDR)* dari *trace file* untuk tiap simulasi dari skenario *Model Traffic Load* ditabulasikan dan diambil rata-rata dengan hasil sebagaimana pada Tabel 5.6.

Tabel 5.6 Hasil Nilai *Packet Delivery Ratio*

Jumlah Maksimal Koneksi (Buah)	PDR (%)	
	TwoRayGround	Nakagami
10	93.172	99.638
20	93.111	99.534
30	93.201	99.599

Tabel 5.6 dan Gambar 5.4 menunjukkan performa *Packet Delivery Ratio* hasil simulasi *Network Simulator 2* dari Model propagasi *TwoRayGround* pada jaringan MANET menggunakan protokol *routing Ad Hoc on Demand Multipah Distance Vector* (AOMDV) dengan skenario *Model Traffic Load.* Terlihat bahwa besar nilai rata-rata dari *Packet Delivery Ratio* (PDR) yang dihasilkan memiliki nilai fluktuatif seiring dengan pertambahan kecepatan maksimal dari pergerakan *node*.



Gambar 5.4 Grafik Nilai *Packet Delivery Ratio* terhadap Jumlah Beban Trafik

Packet Delivery Ratio (PDR) pada model propagasi *TwoRayGround* menunjukkan nilai 93.172% pada jumlah maksimal koneksi 10, pada saat jumlah maksimal koneksi ditambah menjadi 20 koneksi nilai PDR naik menjadi 93.111 %, dan pada saat jumlah maksimal koneksi 30 nilai PDR mengalami sedikit penurunan menjadi 93.201 %. Hasil nilai *Packet Delivery Ratio* (PDR) mengalami penurunan sebesar 0.06% pada saat jumlah koneksi bertambah dari 10 koneksi menjadi 20 koneksi. Dan mengalami kenaikan sebesar 0.09% pada saat jumlah koneksi bertambah dari 20 koneksi menjadi 30 koneksi.

Bila dibandingan dengan model propagasi *Nakagami*, nilai *Packet Delivery Ratio* pada model propagasi *TwoRayGround* memiliki kesamaan yang cenderung memiliki nilai yang stabil. Nilai PDR dari model propagasi *TwoRayGround* memiliki nilai yang lebih rendah dari pada nilai PRD dari model propagasi *Nakagami*, di mana nilai PDR dari model propagasi *Nakagami*

mencapai nilai 99.638% pada jumlah maksimal koneksi 10 buah, pada jumlah maksimal koneksi 20 buah nilai PDR mencapai 99.534 %, dan pada jumlah maksimal koneksi 30 buah nilai PDR dari model propagasi *Nakagami* mencapai 99.599 %.

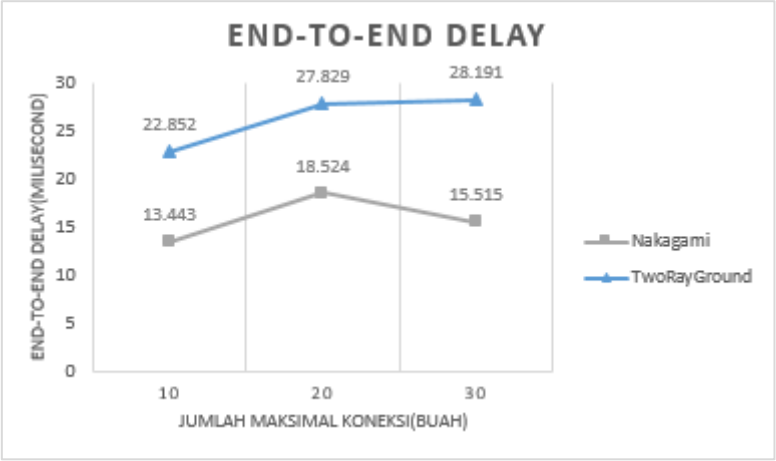
5.2.2.2 Analisis *End-to-End Delay* (E2D)

Hasil analisis *End-to-End Delay (E2D)* dari *trace file* untuk tiap simulasi dari skenario *Model Traffic Load* ditabulasikan dan diambil rata-rata dengan hasil sebagaimana pada Tabel 5.7.

Tabel 5.7 Hasil Nilai *End-to-End Delay*

Jumlah Maksimal Koneksi (Buah)	E2D (Milisecond)	
	TwoRayGround	Nakagami
10	22.852	13.443
20	27.829	18.524
30	28.191	15.515

Tabel 5.7 dan Gambar 5.5 menunjukkan nilai *End-to-End Delay* dari Model propagasi *TwoRayGround* pada jaringan MANET menggunakan protokol *routing Ad Hoc on Demand Multipah Distance Vector (AOMDV)* dengan skenario *Model Traffic Load.* Terlihat bahwa besar nilai rata-rata dari *End-to-End Delay* (E2D) yang dihasilkan menunjukkan kecenderungan bertambah besar seiring dengan bertambahnya jumlah koneksi maksimal.



Gambar 5.5 Grafik Nilai *End-to-End Delay* terhadap Jumlah Beban Trafik

Besar nilai *End-to-End Delay* (E2D) pada model propagasi *TwoRayGround* mengalami kenaikkan sebesar 4.98 *milisecond* pada saat jumlah koneksi maksimal antara 10 koneksi dan 20 koneksi dan kembali mengalamai kenaikan sebesar 0.36 *milisecond* pada jumlah maksimal koneksi dari 20 koneksi menjadi 30 koneksi

Bila dibandingan dengan model propagasi *Nakagami* yang memiliki nilai *End-toEnd Delay* yang cenderung fluktuatif, nilai *End-to-End Delay* pada model propagasi *TwoRayGround* memiliki kecenderungan semakin besar seiring dengan bertambahnya jumlah maksimal koneksi. Nilai E2D dari model propagasi *TwoRayGround* memiliki nilai yang lebih besar dari pada nilai E2D dari model propagasi *Nakagami*, di mana nilai E2D dari model propagasi *Nakagami* sebesar 15.443 *milisecond* pada jumlah maksimal koneksi 10 buah, pada jumlah maksimal koneksi 20 buah nilai E2D sebesar 18.524 *milisecond*, dan pada jumlah maksimal koneksi 30 buah nilai E2D dari model propagasi *Nakagami* mencapai 15.525 *milisecond*.

Nilai *End-to-End Delay* pada model propagasi *TwoRayGround* mengalami kenaikan seiring dengan bertambahnya beban trafik pada saat pengiriman data. Hal ini diakibatkan karena *bandwidth* yang dibutuhkan semakin besar seiring dengan bertambahnya beban trafik sehingga terjadi antrian pada saat paket data dikirimkan.

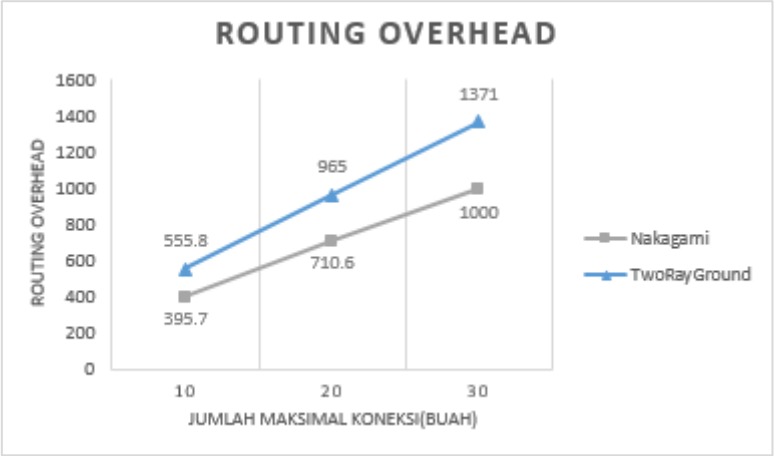
5.2.2.3 Analisis *Routing Overhead* (RO)

Hasil analisis *Routing Overhead (RO)* dari *trace file* untuk tiap simulasi dari skenario *Model Traffic Load* ditabulasikan dan diambil rata-rata dengan hasil sebagaimana pada Tabel 5.5.

Tabel 5.8 Hasil Nilai *Routing Overhead*

Jumlah Maksimal Koneksi (Buah)	RO (Buah)	
	TwoRayGround	**Nakagami**
10	555.8	395.7
20	965.0	710.6
30	1371.0	1000.0

Tabel 5.8 dan Gambar 5.6 menunjukkan nilai *Routing Overhead* dari model propagasi *TwoRayGround* pada jaringan MANET menggunakan protokol *routing Ad Hoc on Demand Multipah Distance Vector (AOMDV)* dengan skenario *Model Traffic Load*. Terlihat bahwa besar nilai rata-rata dari *Rouitng Overhead* (RO) yang dihasilkan cenderung mengalami kenaikkan seiring dengan jumlah koneksi maksimal yang digunakan.



Gambar 5.6 Grafik Nilai *Routing Overhead* terhadap Jumlah Beban Trafik

Besar nilai *Routing Overhead* pada model propagasi *TwoRayGround* mengalami kenaikkan sebesar 73.62 % atau sebanyak 409.2 paket pada saat jumlah koneksi maksimal bertambah dari 10 koneksi menjadi 20 koneksi dan kembali mengalami kenaikan sebesar 42.07% atau sebanyak 406.0 paket pada saat jumlah koneksi maksimal bertambah dari 20 koneksi menjadi 30 koneksi.

Bila dibandingan dengan model propagasi *Nakagami*, nilai *Routing Overhead* pada model propagasi *TwoRayGround* memiliki kesamaan yang cenderung memiliki nilai yang semakin meningkat seiring bertambahnya jumlah beban trafik. Nilai RO dari model propagasi *TwoRayGround* memiliki nilai yang sedikit lebih banyak dari pada nilai RO dari model propagasi *Nakagami*, di mana nilai RO dari model propagasi *Nakagami* sebanyak 59.6 pada jumlah maksimal koneksi 10 buah, pada jumlah maksimal koneksi 20 buah nilai RO sebanyak 60.3, dan pada jumlah maksimal koneksi 30 buah nilai RO sebanyak 62.4.

Nilai *Routing Overhead* pada model propagasi *TwoRayGround* semakin meningkat seiring dengan bertambahnya beban trafik. Hal ini diakibatkan meningkatnya rute gagal seiring dengan bertambahnya beban trafik, sehingga pengiriman paket *routing* menjadi bertambah dan nilai *Rouitng Overhead* menjadi lebih besar.

BAB VI
PENUTUP

Pada Bab ini diberikan kesimpulan yang dapat diambil selama pengerjaan Tugas Akhir beserta saran mengenai pengembangan yang dapat dilakukan terhadap Tugas Akhir ini kedepannya.

6.1. Kesimpulan

Pada subbab ini memaparkan mengenai kesimpulan yang dapat diambil berdasarkan serangkaian uji coba dan analisis penelitian yang dilakukan terhadap kinerja model propagasi *TwoRayGround* pada *Ad Hoc on Demand Multipath Distance Vector (AOMDV) Routing* pada MANET. Kesimpulan tersebut adalah sebagai berikut:

1. Hasil simulasi model propagasi *TwoRayGround* dengan *routing protocol* AOMDV pada lingkungan jaringan MANET dengan menggunakan dua model skenario *Model Mobility Node* dan *Model Traffic Load* menunjukkan hasil sebagai berikut:

   a. Pada simulasi menggunakan skenario *Model Mobility Node* dengan variasi kecepatan maksimal pergerakan node 5 m/s, 10 m/s, dan 15 m/s, nilai *Packet Delivery Ratio* cenderung mengalami penurunan antara rentang nilai 94.831 % dan 93.406 %, nilai End-to-End Delay cenderung fluktuatif antara rentang 16.101 *milisecond* dan 16.707 *milisecond*, dan nilai *Routing Overhead* cenderung fluktuatif antara rentang 69.0 buah paket dan 71.3 buah paket.
   b. Pada simulai menggunakan skenario *Model Traffic Load* dengan variasi jumlah maksimal koneksi 10, 20, dan 30 buah, nilai *Packet Delivery Ratio* cenderung stabil pada retang nilai 93 % , nilai *End-to-End Delay* cenderung

mengalami peningkatan antara rentang 22.852 *milisecond* dan 28.191 *milisecond*, dan nilai *Routing Overhead* cenderung mengalami peningkatan antara rentang 555.8 buah paket dan 1371 buah paket.

2. Analisis hasil simulasi model propagasi *TwoRayGround* dengan *routing protocol* AOMDV pada lingkungan jaringan MANET dengan menggunakan dua model skenario *Model Mobility Node* dan *Model Traffic Load* menunjukkan hasil sebagai berikut:

   a. Pada simulasi dengan menggunakan skenario *Model Mobility Node*, variasi kecepatan maksimal pergerakan *node* tidak memberikan pengaruh yang signifikan terhadap kinerja jaringan MANET. Hal ini dapat dilihat dari hasil nilai *End-to-End Delay* dan *Routing Overhead* yang cenderung memiliki nilai fluktuatif akibat dari pergerakan *node* yang dinamis dan bersifat acak. Sedangkan untuk *Packet Delivery Ratio* memiliki nilai yang cenderung menurun seiring dengan bertambahnya kecepatan maksimal pergerakan *node*. Penurunan nilai PDR disebabkan karena banyaknya paket *drop* yang terjadi.
   b. Pada simulai dengan menggunakan skenario *Model Traffic Load*, seiring dengan bertambahnya jumlah maksimal koneksi maka kinerja jaringan MANET cenderung mengalami penurunan yang dapat diketahui dari semakin meningkatnya nilai *End-to-End Delay* dan *Routing Overhead*. Peningkatan nilai E2D dan RO diakibatkan karena jumlah *bandwith* yang dibutuhkan semakin besar dan resiko rute gagal dalam mengirimkan paket juga lebih besar seiring dengan semakin banyaknya jumlah beban trafik jaringan MANET. Sedangkan nilai *Packet Delivery*

Ratio cenderung memiliki nilai yang stabil seiring bertambahnya jumlah maksimal koneksi.

3. Hasil perbandingan dengan model propagasi *Nakagami* menunjukkan bahwa kinerja model propagasi *TwoRayGround* pada *routing protocol* AOMDV di lingkungan jaringan MANET menggunakan dua model skenario *Model Mobility Node* dan *Model Traffic Load* memiliki nilai *Packet Delivery Ratio* yang lebih rendah, nilai *End-to-End Delay* yang lebih tinggi, dan nilai *Routing Overhead* yang lebih besar. Namun selisih nilai *Packet Delivery Ratio, End-to-End Delay*, dan *Roting Overhead* pada model propagasi *TwoRayGround* tidak begitu besar bila dibandingkan pada model propagasi *Nakagami*.

6.2. Saran

Dari hasil penelitian yang telah dilakukan terhadap kinerja model propagasi *TwoRayGround* pada *Ad Hoc on Demand Multipath Distance Vector* (AOMDV) *Routing* pada MANET, terdapat beberapa saran untuk perbaikan serta pengembangan sebagai berikut:

1. Simulasi model propagasi *TwoRayGround* pada *routing protocol* AOMDV di lingkungan jaringan MANET dapat dilakukan uji coba dengan model skenario lain untuk mengetahui hal lain yang memberikan pengaruh terhadap kinerja dari jaringan MANET.

2. Analisis dari simulasi model propagasi *TwoRayGround* pada *routing protocol* AOMDV di lingkungan jaringan MANET dapat dilakukan dengan mengunkan metrik analisis tambahan seperti *Througput* dan *Jitler* untuk medapatkan analis kinerja yang lebih baik.

3. Dapat dilakukan perbandingan kinerja model propagasi *TwoRayGround* pada *routing protocol* AOMDV di lingkungan jaringan MANET dengan menggunakan protokol *routing* maupun model propagasi yang berbeda untuk mendapatkan variasi hasil yang lebih baik.

DAFTAR PUSTAKA

[1] Adam Tedy Agustian, "Analisis Kinerja Jaringan VANET dengan Model Propogasi Free Space dan TwoRayGround pada Routing AODV", Universitas Muhammadiyah Malang, Malang, 2017.

[2] A. D. Robbins, GAWK: Effective AWK Programming, Boston, MA: Free Software Foundation, 2009.

[3] Ayyasamy, D. A., 2013. Performance Evaluation of Load Based Channel Aware Routing in MANETs with Reusable Path. International Journal of Engineering and Advanced Technology, 3(1).

[4] Denatama, M. I., 2016. Analisis Perbandingan Kinerja Protokol Routing DSDV dan OLSR Untuk Perubahan Kecepatan Mobilitas pada Standar IEEE 802.11ah. Infotel , 8(2), pp. 2085-3688.

[5] Fauzi Dwi S S, dkk "Analisis Performansi Protokol Routing AODV dan DSR pada MANET," [Online]. Available:https://www.academia.edu/9775515/Analisis_Performansi_Protokol_Routing_AODV_dan_DS pada_MANET. [Accessed 28 Sep 2018].

[6] Hanitya Triantono Widya Putra, Sukiswo, dN Imam Santoso, "Kinerja Routing AODV dan AOMDV pada Jaringan WPAN 802.15.4 ZigBee dengan Topologi Mesh", Universitas Diponegoro. Semarang. 2013.

[7] Harmandeep singh, M. singh. (2013). Effect Of Black Hole Attack On AODV Routing Protocol In MANET. International Journal of Computer Science and Technology, 2278–3091(may-june), 43– 46.

[8] Idris Skloul Ibrahim, Peter J.B King, “Robert Pooley: Performance Evaluation of Routing Protocols for MANET”, Fourth International Conference on Systems and Networks Communications, 2009.

[9] Kaur, S., 2013. An Overview of Mobile Ad hoc Network: Application, Challenges&Comparison of Routing Protocols. IOSR Journal of Computer Engineering(IOSR-JCE), 11(5), pp. 07-11.

[10] Marina, M. K., 2001. On-demand Multipath Distance Vector Routing in Ad Hoc Networks. IEEE 9th International Conference on Network Protocols (ICNP), Volume 1, pp. 14-23.

[11] Meeneghan, Paul, Declan Delaney. “An Introduction to NS, Nam and Otcl scripting”. National University of Ireland. Ireland. 2004.

[12] Raja, M. L., 2014. An Overview of MANET: Applications, Attacks and Challenges. International Journal of Computer Science and Mobile Computing, 3(1), pp. 408-417.

[13] Rao, A. R., 2014. Performance Evaluation of DSR, AOMDV and ZRP Routing Protocols in MANETS by using NS2. (IJCSIT) International Journal of Computer Science and Information Technologies, 5(1), pp. 711-714.

[14] S. L. Meshram dan P. D. Dorge, “Design and Performance Analysis of Mobile Ad Hoc Network with Reactive Routing Protocols,” Proc. IEEE International Conference on Communication and Signal Processing (ICCSP), 2017, hal. 443-447.

[15] Saha Misra, ‘Wireless Communication and Networks: 3G and beyond’, Tata McGrawHill Education Pvt. Ltd. New Delhi, 2009, pp. 86-87.

[16] T.K Philips, eta l, “ Connectivity properties of a packet radio network model”, IEEE Trans, Inform Theroy, vol.35, no.5, pages 1044-2047,sept 1989.

[17] T. S. Rappaport, “Wireless Communications: Principles and Practice”, 2nd ed., Prentice-Hall, Upper Saddle River, NJ, 2002.

LAMPIRAN

Lampiran 1. Potongan *File Node-Movement* Berisikan Posisi *Node*

```
# nodes: 50, pause: 10.00, max speed: 10.00, max
x: 510.00, max y: 510.00
#
$node_(0) set X_ 442.908505928928
$node_(0) set Y_ 95.044798061559
$node_(0) set Z_ 0.000000000000
$node_(1) set X_ 297.366193142293
$node_(1) set Y_ 365.866179789921
$node_(1) set Z_ 0.000000000000
$node_(2) set X_ 456.898541637905
$node_(2) set Y_ 78.681624991768
$node_(2) set Z_ 0.000000000000
$node_(3) set X_ 40.962025683862
$node_(3) set Y_ 222.111032672481
$node_(3) set Z_ 0.000000000000
$node_(4) set X_ 60.135951528390
$node_(4) set Y_ 205.876614749975
$node_(4) set Z_ 0.000000000000
$node_(5) set X_ 420.014072126459
$node_(5) set Y_ 303.900157548774
$node_(5) set Z_ 0.000000000000
$node_(6) set X_ 468.823505620307
$node_(6) set Y_ 64.067414795519
$node_(6) set Z_ 0.000000000000
$node_(7) set X_ 397.579402038016
$node_(7) set Y_ 384.719617634937
$node_(7) set Z_ 0.000000000000
$node_(8) set X_ 460.258527180127
$node_(8) set Y_ 275.816792706003
$node_(8) set Z_ 0.000000000000
$node_(9) set X_ 123.282675564475
$node_(9) set Y_ 68.104505512337
$node_(9) set Z_ 0.000000000000
$node_(10) set X_ 27.777430419276
$node_(10) set Y_ 139.999521679687
$node_(10) set Z_ 0.000000000000
```

Lampiran 2. Potongan *File Node-Movement* Berisikan Pembuatan GOD dari Setiap *Node*

```
$god_ set-dist 0 1 2
$god_ set-dist 0 2 1
$god_ set-dist 0 3 2
$god_ set-dist 0 4 2
$god_ set-dist 0 5 1
$god_ set-dist 0 6 1
$god_ set-dist 0 7 2
$god_ set-dist 0 8 1
$god_ set-dist 0 9 2
$god_ set-dist 0 10 2
$god_ set-dist 0 11 2
$god_ set-dist 0 12 1
$god_ set-dist 0 13 1
$god_ set-dist 0 14 2
$god_ set-dist 0 15 1
$god_ set-dist 0 16 1
$god_ set-dist 0 17 1
$god_ set-dist 0 18 3
$god_ set-dist 0 19 2
$god_ set-dist 0 20 3
$god_ set-dist 0 21 2
$god_ set-dist 0 22 1
$god_ set-dist 0 23 2
$god_ set-dist 0 24 2
$god_ set-dist 0 25 2
$god_ set-dist 0 26 3
$god_ set-dist 0 27 3
$god_ set-dist 0 28 2
$god_ set-dist 0 29 2
$god_ set-dist 0 30 2
$god_ set-dist 0 31 2
$god_ set-dist 0 32 2
$god_ set-dist 0 33 2
$god_ set-dist 0 34 2
$god_ set-dist 0 35 2
$god_ set-dist 0 36 2
$god_ set-dist 0 37 3
$god_ set-dist 0 38 3
```

Lampiran 3. Potongan *File Node-Movement* Berisikan Pergerakan *Node*

```
$ns_ at 10.019760496532 "$god_ set-dist 34 42 1"
$ns_ at 10.061585602766 "$god_ set-dist 8 13 1"
$ns_ at 10.061585602766 "$god_ set-dist 8 30 2"
$ns_ at 10.061585602766 "$god_ set-dist 8 35 2"
$ns_ at 10.061585602766 "$god_ set-dist 8 45 2"
$ns_ at 10.137222337641 "$god_ set-dist 15 16 1"
$ns_ at 10.137222337641 "$god_ set-dist 16 32 2"
$ns_ at 10.164523250866 "$god_ set-dist 0 37 2"
$ns_ at 10.164523250866 "$god_ set-dist 37 41 2"
$ns_ at 10.164523250866 "$god_ set-dist 37 42 1"
$ns_ at 10.164523250866 "$god_ set-dist 37 43 2"
$ns_ at 10.221340804011 "$god_ set-dist 18 33 1"
$ns_ at 10.296873885836 "$god_ set-dist 31 32 1"
$ns_ at 10.296873885836 "$god_ set-dist 32 37 2"
$ns_ at 10.330022072348 "$god_ set-dist 3 6 2"
$ns_ at 10.330022072348 "$god_ set-dist 6 16 1"
$ns_ at 10.330022072348 "$god_ set-dist 6 33 2"
$ns_ at 10.354547657271 "$god_ set-dist 8 36 1"
$ns_ at 10.466223346082 "$god_ set-dist 11 33 1"
$ns_ at 10.608373352934 "$god_ set-dist 7 30 2"
$ns_ at 10.608373352934 "$god_ set-dist 25 30 2"
$ns_ at 10.608373352934 "$god_ set-dist 30 34 2"
$ns_ at 10.608373352934 "$god_ set-dist 30 39 2"
$ns_ at 10.608373352934 "$god_ set-dist 30 40 2"
$ns_ at 10.608373352934 "$god_ set-dist 30 42 1"
$ns_ at 10.653194584788 "$god_ set-dist 14 16 1"
$ns_ at 10.750854373202 "$god_ set-dist 28 33 1"
$ns_ at 10.822609026567 "$god_ set-dist 3 38 1"
$ns_ at 10.849729715102 "$god_ set-dist 19 44 1"
$ns_ at 10.977886295334 "$god_ set-dist 1 13 1"
$ns_ at 11.140757384647 "$god_ set-dist 21 30 1"
$ns_ at 11.154631144818 "$god_ set-dist 23 28 1"
$ns_ at 11.154631144818 "$god_ set-dist 23 33 2"
$ns_ at 11.231001672967 "$god_ set-dist 20 36 2"
$ns_ at 11.293111931647 "$god_ set-dist 3 13 1"
$ns_ at 11.293111931647 "$god_ set-dist 3 22 2"
$ns_ at 11.331208453473 "$god_ set-dist 33 45 1"
$ns_ at 11.569236541406 "$god_ set-dist 9 20 2"
$ns  at 11.569236541406 "$god  set-dist 9 37 1"
```

Lampiran 4. Potongan *File Node-Movement* Berisikan Data *Router*

```
#
# Destination Unreachables: 0
#
# Route Changes: 3077
#
# Link Changes: 2535
#
# Node | Route Changes | Link Changes
#    0 |           102 |           89
#    1 |            86 |           78
#    2 |           110 |           70
#    3 |           116 |          109
#    4 |           122 |          111
#    5 |           104 |           99
#    6 |           154 |          118
#    7 |           110 |          103
#    8 |            96 |           93
#    9 |           180 |          126
#   10 |           121 |          113
#   11 |            88 |           78
#   12 |           192 |          121
#   13 |           156 |          137
#   14 |           100 |           94
#   15 |            94 |           86
#   16 |            93 |           75
#   17 |           122 |          107
#   18 |           108 |           96
#   19 |            84 |           82
#   20 |           131 |           58
#   21 |           171 |          129
#   22 |           125 |          103
#   23 |           102 |           81
#   24 |            98 |           78
#   25 |           104 |           93
#   26 |           162 |          107
#   27 |           211 |          139
#   28 |            72 |           67
#   29 |           156 |          115
#   30 |           174 |          154
#   31 |            68 |           68
```

Lampiran 5. Potongan *File Traffic-Connection Pattern*

```
#
# nodes: 50, max conn: 10, send rate: 4.0, seed:
1.0
#
#
# 1 connecting to 2 at time 2.5568388786897245
#
set udp_(0) [new Agent/UDP]
$ns_ attach-agent $node_(1) $udp_(0)
set null_(0) [new Agent/Null]
$ns_ attach-agent $node_(2) $null_(0)
set cbr_(0) [new Application/Traffic/CBR]
$cbr_(0) set packetSize_ 512
$cbr_(0) set interval_ 4.0
$cbr_(0) set random_ 1
$cbr_(0) set maxpkts_ 10000
$cbr_(0) attach-agent $udp_(0)
$ns_ connect $udp_(0) $null_(0)
$ns_ at 2.5568388786897245 "$cbr_(0) start"
#
# 4 connecting to 5 at time 56.333118917575632
#
set udp_(1) [new Agent/UDP]
$ns_ attach-agent $node_(4) $udp_(1)
set null_(1) [new Agent/Null]
$ns_ attach-agent $node_(5) $null_(1)
set cbr_(1) [new Application/Traffic/CBR]
$cbr_(1) set packetSize_ 512
$cbr_(1) set interval_ 4.0
$cbr_(1) set random_ 1
$cbr_(1) set maxpkts_ 10000
$cbr_(1) attach-agent $udp_(1)
$ns_ connect $udp_(1) $null_(1)
$ns_ at 56.333118917575632 "$cbr_(1) start"
```

Lampiran 6. *File* (*.tcl) Simulasi Manet dengan Protokol *Routing* AOMDV

```
#
# ======================================================
# Define options
#
# ======================================================

set val(chan) Channel/WirelessChannel
set val(prop) Propagation/TwoRayGround
set val(netif) Phy/WirelessPhy
set val(mac) Mac/802_11
set val(ifq) Queue/DropTail/PriQueue
set val(ll) LL
set val(ant) Antenna/OmniAntenna
set val(x) 510 ;# X dimension of the topography
set val(y) 510 ;# Y dimension of the topography
set val(ifqlen) 50 ;# max packet in ifq
set val(seed) 1.0
set val(adhocRouting) AOMDV
set val(nn) 50 ;# how many nodes are simulated
set val(cp) "cbr-10"
set val(sc) "scen-10-1"
set val(stop) 200 ;# simulation time

# ======================================================
# Main Program
# ======================================================

# Initialize Global Variables
# create simulator instance
set ns_ [new Simulator]

# setup topography object
set topo [new Topography]

# create trace object for ns and nam
set tracefd [open 10-1.tr w]
set namtrace [open 10-1.nam w]
```

```
$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace $val(x)
$val(y)

# define topology
$topo load_flatgrid $val(x) $val(y)

# Create God
set god_ [create-god $val(nn)]

# define how node should be created
#global node setting
set chan [new $val(chan)]
$ns_ node-config -adhocRouting $val(adhocRouting)\
       -llType $val(ll) \
       -macType $val(mac) \
       -ifqType $val(ifq) \
       -ifqLen $val(ifqlen) \
       -antType $val(ant) \
       -propType $val(prop) \
       -phyType $val(netif) \
       -channel $chan \
       -topoInstance $topo \
       -agentTrace ON \
       -routerTrace ON \
       -macTrace OFF

# Create the specified number of nodes [$val(nn)]
and "attach" them
# to the channel.
for {set i 0} {$i < $val(nn) } {incr i} {
set node_($i) [$ns_ node]
$node_($i) random-motion 0 ;# disable random
motion
}

# Define node movement model
puts "Loading connection pattern..."
source $val(sc)

# Define traffic model
puts "Loading scenario file..."
```

```
# Define node initial position in nam
for {set i 0} {$i < $val(nn)} {incr i} {

# 20 defines the node size in nam, must adjust it
according to your scenario

# The function must be called after mobility model
is defined

$ns_ initial_node_pos $node_($i) 50
}

# Tell nodes when the simulation ends
for {set i 0} {$i < $val(nn) } {incr i} {
$ns_ at $val(stop).0 "$node_($i) reset";
}

$ns_ at $val(stop).0002 "puts \"NS EXITING...\" ;
$ns_ halt"
puts $tracefd "M 0.0 nn $val(nn) x $val(x) y
$val(y) rp $val(adhocRouting)"
puts $tracefd "M 0.0 sc $val(sc) cp $val(cp) seed
$val(seed)"
puts $tracefd "M 0.0 prop $val(prop) ant
$val(ant)"
puts "Starting Simulation..."


proc stop {} {
global ns_tracefd
namtrace
$ns_ flush-trace
close $tracefd
close $namtrace
}

$ns_ run
```

Lampiran 7. Implementasi Skip *Packet Delivery Ratio* (PDR) dengan AWK

```
BEGIN {
SentNum = 0;
RecvNum = 0;
PDR = 0;
}

{
#Menghitung Paket Terkirim
if (($1 == "s") && ($4 == "AGT") && ($7 ==
"cbr")){ SentNum++; }
#Menghitung Paket Diterima
if (($1 == "r") && ($4 == "AGT") && ($7 == "cbr"))
{RecvNum++; }
}

END {
PDR = ( RecvNum / SentNum ) * 100;
print "Jumlah Paket Data Terkirim: %d /n",
SentNum;
print "Jumlah Paket Data Diterima: #d /n",
RecvNum;
print "Packet delivery ratio: %2f /n", PDR, "%";
}
```

Lampiran 8. Implementasi Skip *End-to-End Delay (*E2D) dengan AWK

```
BEGIN {
Seqno = -1;
SentNum = 0;
RecvNum = 0;
Delay = 0;
Count = 0;
E2D = 0
}
```

```
{

#Menghitung Paket Dikirim
if (($1 == "s")  && ($4 == "AGT") && ($7 ==
"cbr")){ SentNum++; }
}

#Menghitung Paket Diterima
if(($1 == "s") && ($4 == "AGT") && (Seqno < $6)) {
Seqno = $6; }
else if(($1 == "r") && ($4 == "AGT") && ($7 ==
"cbr")) {RecvNum++; }

#End-to-End Delay
if(($1 == "s") && ($4 == "AGT") && ($7 == "cbr")) {
start_time[$6] = $2; }
else if(($1 == "r") && ($7 == "cbr")) {
end_time[$6] = $2; }
else if(($1 == "D") && ($7 == "cbr")) {
end_time[$6] = -1; }
}

END {
for(i=0; i <= Seqno; i++) {
if(end_time[i] > 0) { delay[i] = end_time[i] -
start_time[i];
Count++;
}
else { delay[i] = -1; }
}

Count = 10;
for(i=0; i < Count; i++) {
if(delay[i] > 0) { Delay = Delay + delay[i];}
}

E2D = (Delay/RecvNum)*1000;

print "\n";
print "Jumlah Paket Diterima = %d",
receivedPackets;
print "Average End-to-End Delay = %2f ms", E2D;
print "\n";
```

Lampiran 9. Implementasi Skip *Routing Overhead* (RO) dengan AWK

```
BEGIN{
RoNum = 0;
}

{
if (($1 == "s" || $1 == "f") && ($4 == "RTR") &&
($7 == "cbr")){ RoNum++}
}

END{
printf("Besaran routing : %d\n", RoNum);
}
```

Lampiran 10. Hasil Simulasi Model Propagasi *TwoRayGround* untuk Skenario *Model Mobility Node* dengan Kecepatan Maksimal Pergerakan *Node* 5 m/s

Percobaan	Send	Recv	Drop	PDR (%)	E2E (ms)	RO (Paket)
1	49	47	2	95.918	10.985	49
2	50	50	0	100	22.497	65
3	49	47	2	95.918	16.185	56
4	51	48	3	94.118	18.953	74
5	49	49	0	100	19.044	71
6	50	40	10	80	20.601	99
7	53	50	3	94.34	20.988	94
8	51	48	3	94.118	10.579	53
9	46	45	1	97.826	10.271	62
10	51	49	2	96.078	10.901	67

Lampiran 11. Hasil Simulasi Model Propagasi *TwoRayGround* untuk Skenario *Model Mobility Node* dengan Kecepatan Maksimal Pergerakan *Node* 10 m/s

Percobaan	Send	Recv	Drop	PDR (%)	E2E (ms)	RO (Paket)
1	49	44	5	89.796	21.24	94
2	49	46	3	93.878	25.083	75
3	54	54	0	100	25.735	73
4	48	44	4	91.667	15.974	93
5	50	48	2	96	19.521	76
6	48	47	1	97.917	10.714	57
7	51	48	3	94.118	10.724	68
8	51	47	4	92.157	31.614	85
9	46	46	0	100	12.955	46
10	49	41	8	83.673	6.578	69

Lampiran 12. Hasil Simulasi Model Propagasi *TwoRayGround* untuk Skenario *Model Mobility Node* dengan Kecepatan Maksimal Pergerakan *Node 1*5 m/s

Percobaan	Send	Recv	Drop	PDR (%)	E2E (ms)	RO (Paket)
1	49	49	0	100	13.548	70
2	51	44	7	86.275	7.766	82
3	50	46	4	92	20.438	88
4	52	50	2	96.154	14.82	64
5	51	46	5	90.196	10.949	67
6	47	44	3	93.617	15.474	61
7	52	51	1	98.077	27.007	63
8	49	45	4	91.837	24.757	81
9	48	45	3	93.75	16.708	58
10	51	47	4	92.157	15.601	79

Lampiran 13. Hasil Simulasi Model Propagasi *TwoRayGround* untuk Skenario *Model Traffic Load* dengan Jumlah Koneksi Maksimal 10 Koneksi

Percobaan	Send	Recv	Drop	PDR (%)	E2E (ms)	RO (Paket)
1	398	373	25	93.719	30.713	598
2	380	348	32	91.579	10.624	580
3	394	366	28	92.893	19.612	552
4	375	348	27	92.8	31.711	551
5	388	365	22	94.072	28.891	541
6	383	360	22	93.995	17.9	541
7	383	372	11	97.128	36.897	519
8	387	361	26	93.282	10.812	529
9	388	374	14	96.392	17.907	506
10	389	334	55	85.861	23.462	641

Lampiran 14. Hasil Simulasi Model Propagasi *TwoRayGround* untuk Skenario *Model Traffic Load* dengan Jumlah Koneksi Maksimal 20 Koneksi

Percobaan	Send	Recv	Drop	PDR (%)	E2E (ms)	RO (Paket)
1	670	628	42	93.731	22.734	940
2	673	643	30	95.542	42.508	957
3	664	603	61	90.813	57.85	1050
4	659	610	49	92.564	34.699	1017
5	667	618	49	92.654	29.772	960
6	668	606	62	90.719	13.655	1005
7	667	623	44	93.403	19.924	984
8	663	628	35	94.721	15.109	866
9	681	643	38	94.42	16.179	911
10	671	621	50	92.548	25.866	966

Lampiran 15. Hasil Simulasi Model Propagasi *TwoRayGround* untuk Skenario *Model Traffic Load* dengan Jumlah Koneksi Maksimal 30 Koneksi

Percobaan	Send	Recv	Drop	PDR (%)	E2E (ms)	RO (Paket)
1	957	855	102	89.342	27.638	1470
2	948	905	43	95.464	40.962	1342
3	966	898	68	92.961	22.445	1320
4	952	871	81	91.492	33.486	1536
5	939	882	57	93.93	28.023	1441
6	956	901	55	94.247	29.833	1393
7	951	890	61	93.586	16.527	1306
8	954	896	58	93.92	23.498	1258
9	952	884	68	92.857	35.66	1379
10	949	894	56	94.204	23.836	1274

Lampiran 16. Hasil Simulasi Model Propagasi *Nakagami* untuk Skenario *Model Mobility Node* dengan Kecepatan Maksimal Pergerakan *Node* 5 m/s

Percobaan	Send	Recv	Drop	PDR (%)	E2E (ms)	RO (Paket)
1	50	50	0	100	10.329	50
2	48	48	0	100	10.073	48
3	50	50	0	100	18.265	71
4	49	49	0	100	23.594	69
5	50	50	0	100	14.684	50
6	49	48	1	99.959	10.107	54
7	50	50	0	100	10.034	50
8	52	52	0	100	11.959	52
9	50	50	0	100	18.371	79
10	51	51	0	100	17.11	73

Lampiran 17. Hasil Simulasi Model Propagasi *Nakagami* untuk Skenario *Model Mobility Node* dengan Kecepatan Maksimal Pergerakan *Node* 10 m/s

Percobaan	Send	Recv	Drop	PDR (%)	E2E (ms)	RO (Paket)
1	49	49	0	100	10.812	49
2	52	52	0	100	20.309	72
3	48	48	0	100	17.783	77
4	49	49	0	100	17.607	55
5	51	51	0	100	10.746	51
6	50	50	0	100	13.019	50
7	48	48	0	100	10.014	48
8	53	53	0	100	11.149	53
9	48	48	0	100	18.571	96
10	52	50	2	96.154	10.371	52

Lampiran 18. Hasil Simulasi Model Propagasi *Nakagami* untuk Skenario *Model Mobility Node* dengan Kecepatan Maksimal Pergerakan *Node 1*5 m/s

Percobaan	Send	Recv	Drop	PDR (%)	E2E (ms)	RO (Paket)
1	51	51	0	100	10.223	51
2	51	51	0	100	19.491	74
3	51	51	0	100	9.651	51
4	52	51	1	98.077	13.515	55
5	50	50	0	100	17.667	74
6	53	53	0	100	10.05	53
7	51	51	0	100	17.913	98
8	51	51	0	100	10.009	51
9	50	50	0	100	11.889	50
10	54	54	0	100	18.575	67

Lampiran 19. Hasil Simulasi Model Propagasi *Nakagami* untuk Skenario *Model Traffic Load* dengan Jumlah Koneksi Maksimal 10 Koneksi

Percobaan	Send	Recv	Drop	PDR (%)	E2E (ms)	RO (Paket)
1	384	383	1	99.74	18.103	401
2	386	382	4	98.964	11.086	394
3	384	384	0	100	11.779	384
4	390	387	3	99.231	15.96	425
5	376	367	0	100	11.144	367
6	386	385	1	99.741	23.59	401
7	385	385	0	100	10.475	390
8	386	386	0	100	11.249	399
9	379	379	0	100	9.934	379
10	387	382	5	98.708	11.113	417

Lampiran 20. Hasil Simulasi Model Propagasi *Nakagami* untuk Skenario *Model Traffic Load* dengan Jumlah Koneksi Maksimal 20 Koneksi

Percobaan	Send	Recv	Drop	PDR (%)	E2E (ms)	RO (Paket)
1	648	644	4	99.389	11.037	669
2	669	665	4	99.402	15.581	725
3	673	670	3	99.554	32.085	716
4	678	674	4	99.41	16.038	760
5	666	665	1	99.85	25.747	710
6	662	660	2	99.698	10.521	700
7	656	650	6	99.085	15.927	707
8	674	674	0	100	16.894	725
9	658	657	1	99.848	17.517	698
10	676	670	6	99.112	23.897	696

Lampiran 21. Hasil Simulasi Model Propagasi *Nakagami* untuk Skenario *Model Traffic Load* dengan Jumlah Koneksi Maksimal 30 Koneksi

Percobaan	Send	Recv	Drop	PDR (%)	E2E (ms)	RO (Paket)
1	945	938	7	99.259	11.509	981
2	949	949	0	100	17.15	1035
3	960	959	1	99.896	12.493	969
4	958	954	4	99.582	26.835	1042
5	946	932	14	98.52	16.095	1052
6	974	970	4	99.589	12.024	1031
7	958	953	5	99.478	11	970
8	957	954	3	99.687	18.096	990
9	958	955	3	99.687	18.433	988
10	942	941	1	99.894	11.515	942

Lampiran 22. Instalasi *Network Simulator 2*

Dokumentasi tetang cara instalasi *Network Simulator 2* didapat dari sumber *nsnam.com* . Tata cara instalasi *Network Simulator 2* pada sistem operasi *Ubuntu 18.04 LTS* adalah sebagai berikut:

1. Pertama melakukan *update* komponen terbaru dari sistem operasi *Ubuntu* yang dipakai dan pastikan bahwa setiap komponen ter-*update* seluruhnya. Cara melakukan *update* komponen pada *Ubuntu* adalah dengan menjalankan *command line* berikut :

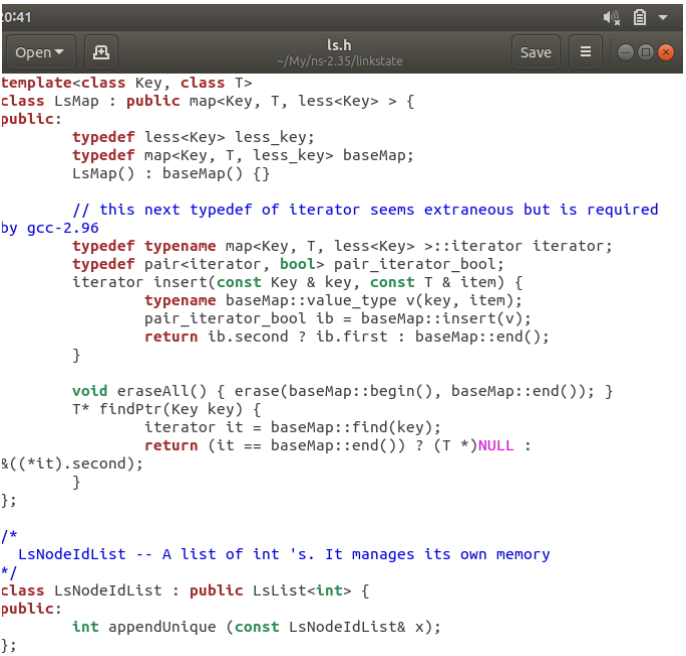
```
$ sudo apt-get update
$ sudo apt-get dist-upgrate
$ sudo apt-get update
```

2. Melakukan instalasi modul-modul dependensi dari *Network Simulator 2* seperti *built-essential, autoconf, automake, tcl 8.5-dev tk 8.5-dev, perl, xgraph, libxt-dev, libx11-dev, libxmu-dev, dan gcc-44.* Cara instalasi modul yang telah disebutkan secara berturut-turut adalah sebagai berikut:

```
$ sudo apt-get install build-essential
autoconf automake
$ sudo apt-get tcl8.5-dev tk8.5-dev
$ sudo apt-get install perl xgraph libxt-dev
libx11-dev libxmu-dev
$ sudo apt-get install gcc-4.4
```

3. Unduh *file Network Simulator versi 2.35* dari halaman *nsnam.com* dan pindahkan menuju *folder /home* kemudian *file Network Simulator 2.35* diekstrak pada folder tersebut.
4. Ubah skrip pada *ls.h* yang terletak pada folder */ns-allinone-2.35/ns-2.35/linkstate/ls.h*, line ke-137 dilakukan perubahan pada *(erase)* menjadi *(this-erase)*.

```
$ cd /ns-allinone-2.35/ns2.3.5/linkstate/
$ gedit ls.h
```




```
template<class Key, class T>
class LsMap : public map<Key, T, less<Key> > {
public:
        typedef less<Key> less_key;
        typedef map<Key, T, less_key> baseMap;
        LsMap() : baseMap() {}

        // this next typedef of iterator seems extraneous but is required
by gcc-2.96
        typedef typename map<Key, T, less<Key> >::iterator iterator;
        typedef pair<iterator, bool> pair_iterator_bool;
        iterator insert(const Key & key, const T & item) {
                typename baseMap::value_type v(key, item);
                pair_iterator_bool ib = baseMap::insert(v);
                return ib.second ? ib.first : baseMap::end();
        }

        void eraseAll() { erase(baseMap::begin(), baseMap::end()); }
        T* findPtr(Key key) {
                iterator it = baseMap::find(key);
                return (it == baseMap::end()) ? (T *)NULL :
&((*it).second);
        }
};

/*
  LsNodeIdList -- A list of int 's. It manages its own memory
*/
class LsNodeIdList : public LsList<int> {
public:
        int appendUnique (const LsNodeIdList& x);
};
```

5. Kembali ke folder */home/ns-allinone-2.35/ns-2.35*. Setelah semua step di atas dilakukan, selanjutnya masuk pada proses instalasi *Network Simulator 2* dengan menggunakan *command line* berikut:

```
$ sudo ./install
```

6. Untuk melakukan pengujian apakah aplikasi *Network Simulator 2* telah berhasil ter-*install*, dapat dilakukan dengan menuliskan *command line 'ns'* pada *terminal*. Apabila muncul tanda '%' pada *terminal* maka aplikasi *Network Simulator 2* telah berhasil di-*install* dengan benar.

BIODATA PENULIS

Ahmad Chanif Eka Wildana, biasa dipanggil dengan Chanif, lahir di Lamongan pada 28 November 1993. Penulis adalah anak pertama dari dua bersaudara yang dibesarkan di Lamongan, Jawa Timur. Penulis mulai menempuh pendidikan formal di MIM 13 Sendangagung (2000-2006), SMPM 12 Sendangagung (2006-2009), MA Al-Ishlah (2009-2012). Pada tahun 2012, penulis memulai pendidikan S1 Jurusan Teknik Informatika, Fakultas Teknologi Informasi di Institut Teknologi Sepuluh Nopember Surabaya angkatan 2012 yang terdaftar dengan NRP. 5112100701.

Di Jurusan Teknik Informatika, penulis mengambil bidang minat Arsitektur Jaringan Komputer (AJK) dan memiliki ketertarikan di bidang *Operation System Linux*, Jaringan Komputer, Jaringan Terdistribusi, dan Teknik Kompresi. Penulis dapat dihubungi melalui alamat e-mail chaniev73@gmail.com.

[*Halaman ini sengaja dikosongkan*]