



TUGAS AKHIR - K1141502

Studi Kerja Model Propagasi Nakagami dan TwoRayGround pada Protokol DSDV di Lingkungan Dinamis

ANDRE ABDIRROSYID
NRP 5112 100 186

Dosen Pembimbing
Dr. Eng. Radityo Anggoro, S.Kom., M.Sc.
Tohari Ahmad, S.Kom., MIT, Ph.D.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018



TUGAS AKHIR - KI141502

Studi Kerja Model Propagasi Nakagami dan TwoRayGround pada Protokol DSDV di Lingkungan Dinamis

ANDRE ABDIRROSYID
NRP 5112 100 186

Dosen Pembimbing
Dr. Eng. Radityo Anggoro, S.Kom., M.Sc.
Tohari Ahmad, S.Kom., MIT, Ph.D.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

[Halaman ini sengaja dikosongkan]



FINAL PROJECT - KI141502

Performance Study of Nakagami and TwoRayGround Propagation Model for DSDV on Dynamic Environment

ANDRE ABDIRROSYID
NRP 5112 100 186

Advisor
Dr. Eng. Radityo Anggoro, S.Kom., M.Sc.
Tohari Ahmad, S.Kom., MIT, Ph.D.

INFORMATICS DEPARTMENT
Faculty of Information Technology and Communication
Institut Teknologi Sepuluh Nopember
Surabaya 2018

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

STUDI KERJA MODEL PROPAGASI NAKAGAMI DAN TWORAYGROUND PADA PROTOKOL DSDV DI LINGKUNGAN DINAMIS

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:

ANDRE ABDIRROSYID
NRP. 5112 100 186

Disetujui oleh Pembimbing Tugas Akhir

Dr. Eng. Radityo Anggoro, S.Kom.
NIP: 198410162008121002

Tohari Ahmad, S.Kom., MIT., Ph.D.
NIP: 197505252003121002



SURABAYA
JANUARI, 2018

[Halaman ini sengaja dikosongkan]

STUDI KERJA MODEL PROPAGASI NAKAGAMI DAN TWRAYGROUND PADA PROTOKOL DSDV DI LINGKUNGAN DINAMIS

Nama Mahasiswa : Andre Abdirrosyid
NRP : 5112 100 186
Jurusan : Departemen Informatika FTIK - ITS
Dosen Pembimbing : Dr. Eng. Radityo Anggoro, S.Kom.,
M.Sc.
Tohari Ahmad, S.Kom., MIT, Ph.D.

ABSTRAK

Teknologi nirkabel (wireless) sudah banyak dikembangkan pada perangkat telekomunikasi. Teknologi ini memungkinkan suatu perangkat untuk mengirimkan data ke perangkat lain dalam keadaan bergerak dalam suatu jaringan yang memiliki infrastruktur yang tidak tetap. Jaringan itu disebut dengan Mobile Ad-Hoc Network (MANET).

Mobile Ad-Hoc Network (MANET) merupakan suatu jaringan yang terdiri dari wireless node yang dapat diatur secara dinamis di mana pun dan kapan pun tanpa menggunakan infrastruktur jaringan yang tetap dan pusat administrasi data. Namun, untuk implementasi MANET di dunia nyata masih sulit untuk dilakukan. Oleh karena itu, penelitian dilakukan dengan menggunakan suatu simulator. Simulator yang biasa digunakan untuk penelitian MANET adalah Network Simulator (NS).

Tujuan dari tugas akhir ini adalah untuk mengidentifikasi performa dari model propagasi Nakagami dan TwoRayGround pada protokol Destination Sequenced Distance Vector (DSDV) dalam lingkungan MANET. Skema MANET dihasilkan oleh file node-movement dan traffic pattern yang ada dalam distribusi Network Simulator dimana yang digunakan adalah NS-2. Terdapat tiga hasil performa: Packet Delivery Ratio (PDR), End-to-End Delay (E2D), dan Routing Overhead (RO).

Kata Kunci: DSDV, MANET, Nakagami, Network Simulator, NS-2, TwoRayGround.

Performance Study of Nakagami and TwoRayGround Propagation Model for DSDV on Dynamic Environment

Name : Andre Abdirrosyid
NRP : 5112 100 186
Major : Informatics Engineering, IT Dept – ITS
Advisor : Dr. Eng. Radityo Anggoro, S.Kom., M.Sc.
Tohari Ahmad, S.Kom., MIT, Ph.D.

ABSTRACT

Wireless Technology has been developed and installed on telecommunication device. This Technology allows a moving device to transmit data to other moving devices in a network. The infrastructure of such network is not fixed and called as Mobile Ad-Hoc Network (MANET).

Mobile Ad-Hoc Network (MANET) is a network consists of wireless nodes that can be configured anywhere and anytime without using fixed network infrastructure and central data administration. But, MANET is still hard to be implemented in real devices. Therefore, this study is done by using a simulator. The simulator used for the study of MANET is known as Network Simulator (NS).

The purpose of this final project is to identify performance of Nakagami and TwoRayGround transmission model on Destination Sequenced Distance Vector (DSDV) routing protocol on MANET. The MANET schemes are generated by node-movement and traffic-pattern files that are already distributed in Network Simulator which is NS-2. There are three performance results: Packet Delivery Ratio (PDR), End-to-End Delay (E2D), dan Routing Overhead (RO).

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

[Halaman ini sengaja dikosongkan]

KATA PENGANTAR

Bismillaahirrohmaanirrohiim

Puji syukur bagi Allah SWT atas segala rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul:

“Studi Kerja Model Propagasi Nakagami dan TwoRayGround pada Protokol DSDV di Lingkungan Dinamis”.

Melalui lembar ini, penulis ingin menyampaikan terima kasih dan penghormatan yang sebesar-besarnya kepada:

1. Allah SWT, karena limpahan rahmat dan karunia-Nya lah penulis dapat menyelesaikan tugas akhir dan perkuliahan di Departemen Informatika ITS.
2. Keluarga penulis, terutama Bapak Karnadi selaku ayah dari penulis dimana tanpa dukungan dari beliau, penulis tidak akan bisa mencapai sejauh ini. Teirma kasih atas kesabaran, dukungan, dan doa dari Bapak. Sungguh jasa beliau tidak akan bisa dituliskan dengan kata-kata.
3. Bapak Dr. Eng. Radityo Anggoro, S. Kom, M. Sc. selaku dosen pembimbing sekaligus dosen wali penulis dan Bapak Tohari Ahmad, S.Kom, MIT., Ph.D. yang telah memberikan bimbingan, dukungan, masukan, nasihat, dan banyak arahan kepada penulis dalam menyelesaikan Tugas Akhir dan perkuliahan penulis.
4. Bapak Dr. Darlis Herumurti, S. Kom, M. Kom. selaku Ketua Departemen Informatika ITS.
5. Teman-teman The Raid yang memberikan bumbu di masa remaja penulis.
6. Teman-teman TC 2012, kakak angkatan, dan adik angkatan yang memberikan warna dan rasa kekeluargaan di kehidupan perkuliahan penulis.

7. Teman-teman Laboratorium DTK dan AJK yang menemani penulis di masa akhir perkuliahan penulis.
8. Pihak-pihak yang tidak dapat penulis sebutkan satu per satu yang telah membantu penulis dalam menyelesaikan Tugas Akhir ini.

Penulis menyadari bahwa Tugas Akhir ini masih jauh dari kata sempurna. Oleh karena itu, kritik dan saran dapat disampaikan kepada penulis untuk perbaikan ke depannya. Semoga Allah SWT memberkati dan membalas semua kebaikan yang dilakukan.

Surabaya, Januari 2018

Andre Abdirrosyid

DAFTAR ISI

LEMBAR PENGESAHAN	vii
Abstrak.....	ix
Abstract.....	xi
KATA PENGANTAR	xiii
DAFTAR ISI.....	xv
DAFTAR GAMBAR.....	xvii
1 BAB I PENDAHULUAN.....	1
1.1. Latar Belakang	1
1.2. Rumusan Permasalahan	2
1.3. Batasan Permasalahan.....	2
1.4. Tujuan dan Manfaat	2
1.5. Metodologi.....	3
1.6. Sistematika Penulisan	4
2 BAB II TINJAUAN PUSTAKA.....	7
2.1. <i>Mobile Ad-Hoc Network</i> (MANET)	7
2.2. <i>Destination-Sequenced Distance Vector</i> (DSDV)	9
2.3. Model Propagasi <i>Nakagami</i>	9
2.4. Model Propagasi <i>TwoRayGround</i>	10
2.5. <i>Network Simulator</i>	10
2.6. <i>Generator File Node-Movement dan Traffic-Connection Pattern</i>	11
2.6.1. <i>File Node-Movement (Mobility Generator)</i>	11
2.6.2. <i>File Traffic-Connection Pattern</i>	14
2.7. <i>NS-2 Trace File</i>	15
2.8. <i>Awk</i>	16
3 BAB III PERANCANGAN SISTEM.....	19
3.1. Deskripsi Umum	19
3.2. Perancangan Skenario.....	19
3.2.1. Skenario <i>Node-Movement (Mobility Generation)</i>	20
3.2.2. <i>Traffic-Connection Pattern Generation</i>	20
3.3. Perancangan Simulasi pada NS-2	21

3.4.	Perancangan Metrik Analisis.....	22
3.4.1.	<i>Packet Delivery Ratio</i> (PDR)	22
3.4.2.	<i>End-to-End Delay</i> (E2D).....	23
3.4.3.	<i>Routing Overhead</i> (RO)	23
4	BAB IV IMPLEMENTASI.....	25
4.1.	Lingkungan Pembangunan Perangkat Lunak.....	25
4.1.1.	Lingkungan Perangkat Lunak.....	25
4.1.2.	Lingkungan Perangkat Keras.....	25
4.2.	Implementasi Skenario	25
4.2.1.	Skenario <i>File Node-Movement (Mobility Generation)</i>	26
4.2.2.	<i>File Traffic-Connection Pattern Generation</i>	30
4.3.	Implementasi Simulasi pada NS-2	31
4.4.	Implementasi Metrik Analisis	37
4.4.1.	<i>Packet Delivery Ratio</i> (PDR)	37
4.4.2.	<i>End-to-End Delay</i> (E2D).....	39
4.4.3.	<i>Routing Overhead</i> (RO)	41
5	BAB V PENGUJIAN DAN EVALUASI	43
5.1.	Lingkungan Pengujian.....	43
5.2.	Kriteria Pengujian.....	43
5.3.	Analisis <i>Packet Delivery Ratio</i> (PDR)	44
5.4.	Analisis <i>End-to-End Delay</i> (E2D).....	46
5.5.	Analisis <i>Routing Overhead</i> (RO)	48
6	BAB VI PENUTUP.....	51
6.1.	Kesimpulan.....	51
6.2.	Saran.....	52
	DAFTAR PUSTAKA.....	53
	7 LAMPIRAN	55
	BIODATA PENULIS.....	75

DAFTAR GAMBAR

Gambar 2.1. <i>Command line 'setdest'</i>	11
Gambar 2.2. <i>Command line 'setdest'</i>	12
Gambar 2.3. Potongan hasil <i>output</i> pada <i>file</i> "scena1-25.txt"	13
Gambar 2.4. Potongan "GOD" dalam <i>file</i> "scena1-25.txt."	13
Gambar 2.5. <i>Command line 'cbrgen.tcl'</i>	14
Gambar 2.6. Contoh <i>command line cbrgen.tcl</i>	15
Gambar 2.7. Koneksi cbr pada <i>file</i> "cbrtest.txt"	15
Gambar 2.8. <i>Trace</i> pengiriman paket data	16
Gambar 2.9. <i>Trace</i> penerimaan paket data	16
Gambar 4.1. <i>Command line 'setdest'</i>	26
Gambar 4.2. Implementasi <i>command line 'setdest'</i>	27
Gambar 4.3. Posisi <i>node</i> dalam X, Y, dan Z pada <i>file</i> "scena1-5.txt"	27
Gambar 4.4. Pergerakan <i>node</i> pada <i>file</i> "scena1-5.txt"	28
Gambar 4.5. GOD pada <i>file</i> "scena1-5.txt"	29
Gambar 4.6. <i>Access Point</i>	29
Gambar 4.7. Hasil animasi untuk posisi <i>node</i>	30
Gambar 4.8. <i>Command line 'cbrgen.tcl'</i>	30
Gambar 4.9. Implementasi <i>command line 'cbrgen.tcl'</i>	31
Gambar 4.10. <i>Output file "cbrtest.txt"</i>	32
Gambar 4.11. Konfigurasi awal parameter NS-2	32
Gambar 4.12. Konfigurasi <i>transmission range</i> NS-2	33
Gambar 4.13. Konfigurasi <i>trace file</i> dan pergerakan <i>node</i> NS-2	34
Gambar 4.14. Contoh hasil <i>running file tcl</i>	35
Gambar 4.15. Konfigurasi pengiriman paket data NS-2	36
Gambar 4.16. <i>Pseudocode PDR</i>	38
Gambar 4.17. Perintah untuk menjalankan "pdr.awk"	39
Gambar 4.18. Hasil <i>running pdr.awk</i>	39
Gambar 4.19. <i>Pseudocode E2D</i>	40
Gambar 4.20. Perintah untuk menjalankan "e2e.awk"	41
Gambar 4.21. Hasil <i>running "e2e.awk"</i>	41
Gambar 4.22 <i>Pseudocede RO</i>	41
Gambar 4.23 Perintah untuk menjalankan "ro.awk"	42

Gambar 4.24 Hasil <i>running</i> “ro.awk”	42
Gambar 5.1 Grafik hasil PDR	45
Gambar 5.2 Grafik hasil E2D	47
Gambar 5.3 Grafik hasil RO.....	48
Gambar 7.1 Posisi node dari potongan file “scena1-5.txt”.....	58
Gambar 7.2 Pembuatan GOD node dari potongan “scena1-5.txt”	62
Gambar 7.3 Pergerakan node dari potongan "scena1-5.txt"	66
Gambar 7.4 Informasi GOD dari potongan "scena1-5.txt"	67
Gambar 7.5 Koneksi yang digunakan pada cbrtest.txt	68
Gambar 7.6 File tcl untuk protokol <i>routing</i> DSDV	71
Gambar 7.7 Implementasi Packet Delivery Ratio pada file "pdr.awk".....	72
Gambar 7.8 Implementasi <i>Routing Overhead</i> pada file “ro.awk”	72
Gambar 7.9 Implementasi <i>End-to-End Delay</i> pada file “e2e.awk”	73

DAFTAR TABEL

Tabel 2.1. Keterangan <i>command line</i> 'setdest'	12
Tabel 2.2. Keterangan <i>command line</i> 'cbrgen.tcl'	14
Tabel 3.1. Parameter skenario <i>node-movement</i>	20
Tabel 3.2. Parameter <i>traffic-connection pattern</i>	21
Tabel 3.3. Parameter simulasi pada NS-2	21
Tabel 5.1 Spesifikasi laptop yang digunakan.....	43
Tabel 5.2 Kriteria pengujian	43
Tabel 5.3 Hasil <i>Packet Delivery Ratio</i> (PDR)	44
Tabel 5.4 Hasil <i>End-to-End Delay</i> (E2D).....	46
Tabel 5.5 Hasil <i>Routing Overhead</i>	48

[Halaman ini sengaja dikosongkan]

BAB I

PENDAHULUAN

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

1.1. Latar Belakang

Berkembangnya teknologi informasi membuat komunikasi dan penyampaian informasi menjadi lebih mudah. Dapat dilihat dari penggunaan perangkat komunikasi seperti *handphone*, *tablet*, dan lainnya yang dapat terhubung satu sama lain menggunakan teknologi nirkabel (*wireless*). Teknologi *wireless* tersebut memungkinkan suatu perangkat untuk terhubung dengan perangkat lainnya dalam suatu jaringan dalam posisi bergerak. Setelah terhubung, perangkat-perangkat tersebut membentuk suatu jaringan yang bersifat sementara dan dapat disebut dengan *Mobile Ad-Hoc Network* (MANET).

Mobile Ad-Hoc Network (MANET) adalah kumpulan dari *wireless node* yang diatur secara dinamis tanpa bergantung pada suatu jaringan infrastruktur yang tetap. Dalam jaringan tersebut, terdapat banyak *node* yang bergerak secara bebas. Akan tetapi, setiap *node* tidak terhubung secara langsung satu sama lain dikarenakan jarak propagasi yang terbatas. Pada MANET, terdapat tiga jenis protokol *routing*, yaitu *proactive*, *reactive*, dan *hybrid*.

Implementasi MANET dapat dilakukan dengan menggunakan simulasi karena implementasi pada dunia nyata tidak memungkinkan. Simulasi dapat dilakukan dengan *Network Simulator* (NS-2) dan menggunakan salah satu protokol *routing proactive*, yaitu protokol *Destination-Sequenced Distance Vector* (DSDV). Selain itu, digunakan model propagasi *Nakagami* dan *TwoRayGround* sebagai perbandingan.

Hasil yang diharapkan dari Tugas Akhir ini adalah perbandingan performa antara model propagasi *Nakagami* dan *TwoRayGround* pada protokol routing DSDV di lingkungan MANET. Perbandingan performa tersebut diukur berdasarkan *Packet Delivery Ratio* (PDR), *End-to-End Delay* (E2D), dan *Routing Overhead* (RO).

1.2. Rumusan Permasalahan

Rumusan masalah yang terdapat dalam Tugas Akhir ini adalah sebagai berikut.

1. Bagaimana performa model propagasi *Nakagami* pada protokol *routing* DSDV di lingkungan MANET?
2. Bagaimana performa model propagasi *TwoRayGround* pada protokol *routing* DSDV di lingkungan MANET?

1.3. Batasan Permasalahan

Batasan masalah yang terdapat dalam Tugas Akhir ini adalah sebagai berikut.

1. Protokol *routing* hanya dijalankan dan diuji coba pada aplikasi *Network Simulator 2* (NS-2).
2. Protokol *routing* yang diuji coba adalah DSDV (*Destination-Sequenced Distance Vector*).
3. Lingkungan yang digunakan untuk uji coba adalah *Mobile Ad Hoc Network* (MANET).
4. Model propagasi yang akan dibandingkan dalam Tugas Akhir ini adalah *Nakagami* dan *TwoRayGround*.
5. Performa yang dibandingkan adalah *Packet Delivery Ratio* (PDR), *End-to-End Delay* (E2D), dan *Routing Overhead* (RO).

1.4. Tujuan dan Manfaat

Tujuan dari pembuatan Tugas Akhir ini adalah:

1. Mendapatkan hasil performa model propagasi *Nakagami* pada protokol *routing* DSDV di lingkungan MANET.

2. Mendapatkan hasil performa model propagasi *TwoRayGround* pada protokol *routing* DSDV di lingkungan MANET.

Tugas akhir ini diharapkan dapat memberikan manfaat dalam menentukan model propagasi yang tepat untuk pengiriman data pada lingkungan MANET dengan implementasi protokol *routing* DSDV berdasarkan *Packet Delivery Ratio* (PDR), *End-to-End Delay* (E2D), dan *Routing Overhad* (RO).

1.5. Metodologi

Beberapa tahap dalam pengerjaan Tugas Akhir ini adalah sebagai berikut:

1. Penyusunan Proposal Tugas Akhir

Tahap pertama dalam proses pengerjaan Tugas Akhir ini adalah menyusun proposal Tugas Akhir. Pada proposal Tugas Akhir tersebut, diajukan cara untuk mengetahui perbandingan hasil studi dari model propagasi *Nakagami* dan *TwoRayGround* di lingkungan dinamis.

2. Studi Literatur

Pada tahap ini, dilakukan pencarian studi literatur yang nantinya akan dijadikan sebagai referensi dalam pengerjaan Tugas Akhir. Studi literatur dapat diambil dari buku, internet, materi kuliah, atau *paper* yang berhubungan dengan topik Tugas Akhir yang diajukan.

3. Analisis dan Desain Perangkat Lunak

Perancangan uji coba MANET diawali dengan melakukan pembuatan skenario pada *mobility generation* yang bersifat *random way point*. Setelah itu dibuat koneksi dengan menggunakan *file traffic-connection* yang sudah ada di dalam NS-2.

4. Implementasi Perangkat Lunak

Tugas akhir ini akan dikerjakan di sistem operasi Windows 10 64-bit, Linux Ubuntu 14.04 LTS 64-bit, dan menggunakan aplikasi NS-2.

5. Pengujian dan Evaluasi

Pada Tugas Akhir ini, pengujian dilakukan dengan menghitung *Packet Delivery Ratio* (PDR), *End-to-End Delay* (E2D), dan *Routing Overhead* (RO) pada masing-masing model propagasi *Nakagami* dan *TwoRayGround*.

6. Penyusunan Buku Tugas Akhir

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

1. Pendahuluan
 - a. Latar Belakang
 - b. Rumusan Masalah
 - c. Batasan Masalah
 - d. Tujuan
 - e. Metodologi
 - f. Sistematika Penulisan
2. Tinjauan Pustaka
3. Perancangan Sistem
4. Implementasi
5. Pengujian dan Evaluasi
6. Kesimpulan dan Saran
7. Daftar Pustaka

1.6. Sistematika Penulisan

Buku Tugas Akhir ini bertujuan untuk memberikan gambaran dari pengerjaan Tugas Akhir ini. Secara garis besar, buku Tugas Akhir ini terdiri dari beberapa bab yang dijelaskan sebagai berikut:

- Bab I Pendahuluan
Bab ini berisi latar belakang masalah, permasalahan, batasan masalah, tujuan dan manfaat Tugas Akhir, metodologi yang digunakan, dan sistematika penyusunan Tugas Akhir.
- Bab II Tinjauan Pustaka
Bab ini membahas tentang beberapa teori penunjang yang berhubungan dengan pokok bahasan dan mendasari pembuatan Tugas Akhir.
- Bab III Perancangan
Bab ini berisi tentang perancangan metode yang nantinya akan diimplementasikan dan diuji.
- Bab IV Implementasi
Bab ini membahas implementasi dari rancangan sistem atau desain yang dilakukan pada tahap perancangan. Penjelasan berupa implementasi skenario *node mobility* pada jaringan *wireless* dengan menggunakan file *node-movement* dan *traffic pattern* yang ada pada *network simulator*, konfigurasi skrip, dan skrip analisis yang digunakan untuk menguji performa model propagasi.
- Bab V Pengujian dan Evaluasi
Bab ini menjelaskan tahap pengujian sistem performa dalam skenario mobilitas *ad hoc* yang dibuat dalam distribusi *mobility network simulator*.
- Bab VI Penutup
Bab ini membahas tentang beberapa teori penunjang yang berhubungan dengan pokok bahasan dan mendasari pembuatan Tugas Akhir.

[Halaman ini sengaja dikosongkan]

BAB II

TINJAUAN PUSTAKA

Pada bab ini akan dibahas mengenai teori-teori yang berkaitan dengan implementasi perangkat lunak. Penjelasan ini bertujuan untuk memberikan gambaran atau definisi secara umum terhadap alat, protokol *routing*, dan definisi yang digunakan dalam pembuatan Tugas Akhir.

2.1. *Mobile Ad-Hoc Network* (MANET)

Mobile Ad-Hoc Network (MANET) adalah jaringan yang terdiri dari sekumpulan *node* yang bergerak (*mobile*) dan dapat berkomunikasi satu sama lain. *Mobile host* dalam jaringan MANET dapat berperan menjadi *router*. Karakteristik dari MANET adalah konfigurasi jaringan yang dinamis dan *bandwidth*, daya, dan keamanan yang terbatas. *Node* dalam MANET dapat berperan sebagai *end-user* sekaligus *router* yang menghitung dan memilih *route-path*.

Dengan berkembangnya teknologi saat ini, MANET bisa menjadi hal yang sangat membantu dapat proses pengiriman data. Misalkan jika akan melakukan komunikasi data antara 10 orang dalam area terbuka dan tidak ada jaringan internet. Pada MANET, tiap orang tersebut bertindak sebagai *node* dan dapat terkoneksi secara langsung tanpa bergantung jaringan internet untuk terhubung satu sama lain [1].

Protokol *routing* untuk MANET dapat dikategorikan menjadi tiga jenis, yaitu:

A. *Proactive*

Protokol *routing proactive* merupakan protokol berdasar pada *routing table* yang terus di-*update* dalam waktu berkala. Semua informasi mengenai perubahan akan dikirim sesuai dengan pengiriman *update* dari *routing table*. Sehingga, *routing table* akan terus di-*update* tanpa

menghilangkan fungsi untuk memilih jalur terbaik untuk pengiriman data [2]. Protokol ini memungkinkan *delay* yang lebih rendah dalam pengiriman data dalam jaringan karena jalur pengiriman data sudah dikenal [3]. Contoh protokol routing *proactive* adalah *Optimized Link State Routing (OLSR)*, *Better Approach To Mobile Ad-Hoc Network (B.A.T.M.A.N.)*, dan *Destination-Sequenced Distance-Vector (DSDV)*.

B. Reactive

Protokol *routing reactive* mencari rute pengiriman data sesuai permintaan dan menentukan *link* untuk mengirim dan menerima paket dari *node* asal ke *node* tujuan. Proses penentuan rute digunakan sesuai dengan permintaan dengan mengirimkan paket *route request (RREQ)* ke seluruh jaringan. Contoh protokol *routing reactive* adalah *Dynamic Source Routing (DSR)* dan *Ad-Hoc On-Demand Distance Vector (AODV)*. [4]

C. Hybrid

Protokol *routing hybrid* dikembangkan dengan pemikiran untuk menggabungkan kelebihan dari prokol *routing proactive* dan *reactive* sehingga didapatkan sebuah protokol *routing* yang paling efektif. Protokol *routing hybrid* menggunakan karakteristik protokol *routing proactive* dan *reactive* untuk mencari jalur terbaik sesuai dengan kondisi (*on-demand*) jaringan yang terus di-*update*. Selain itu, pada protokol *routing hybrid*, paket *Route Request (RREQ)* dan *Route Reply (RREP)* dikirimkan setelah terdapat *routing request* dengan waktu interval tertentu. Contoh dari protokol routing ini adalah *Zone Routing Protocol (ZRP)* dan *Babel* [2].

2.2. Destination-Sequenced Distance Vector (DSDV)

DSDV merupakan salah satu *Proactive Routing Protocol* yang membutuhkan setiap *node* untuk mengirimkan paket *routing update* ke seluruh *node* tetangganya secara periodik. Setiap *node* menyimpan *table routing* yang mengandung informasi yang dibutuhkan untuk sampai ke *node* tujuan [5]. Setiap informasi di *table routing* diberi nomor *sequence* untuk menghindari terjadinya *looping* [6].

DSDV ialah menyediakan *route* yang bebas *loop* untuk setiap waktu. Namun, komunikasi antar *node* bisa menjadi berlebihan karena dilakukannya *update routing* secara periodik [7]. Data yang dikirim ke seluruh *node* memiliki alamat *node* tujuan, jumlah *hop* yang dibutuhkan untuk mencapai tujuan, dan nomor *sequence*.

2.3. Model Propagasi Nakagami

Model *Nakagami* adalah model propagasi yang dapat diterapkan untuk berbagai kondisi *fading*, tergantung pada parameter m yang digunakan. *Probability Density Function* (PDF) dari *Nakagami* dinyatakan sebagai berikut:

$$p_R(R) = \frac{2^m R^{2m-1}}{\Omega^m \Gamma(m)} \exp\left(-\frac{mR^2}{\Omega}\right) \quad (1)$$

Amplitudo $R \geq 0$, $\Omega = E(R^2)$ adalah rata-rata kekuatan *fading*, $E(\cdot)$ adalah *expectation operator*, dan $\Gamma(\cdot)$ adalah fungsi *gamma*. Pada persamaan tersebut, m adalah parameter *fading* dari *Nakagami* yang menentukan keras *fading* [8]. m adalah *inverse* varian normal dari R^2 .

2.4. Model Propagasi *TwoRayGround*

Model propagasi ini berdasar pada optik geometri dan dapat digunakan untuk *direct path* dan refleksi dari *ground* antara *transmitter* dan *receiver*. Model ini dirasa akurat untuk memperkirakan kekuatan sinyal dalam skala luas dengan jarak beberapa kilometer untuk sistem *mobile radio* dengan *tower* dan antena yang tinggi. Kekuatan yang diterima dengan jarak d diberikan oleh persamaan (2):

$$\text{Pr}(d) = \frac{P_t G_t G_r h_t^2 h_r^2}{d^4 L} \quad (2)$$

- P_t = Power atau kekuatan transmisi
- G_t = Tegangan antenna *transmitter*
- G_r = Tegangan antenna *receiver*
- H_t = Tinggi antenna *transmitter*
- H_r = Tinggi antenna *receiver*
- L = asumsi sama dengan nilai L pada propagasi *free space* ($L=1$)

Berdasarkan persamaan di atas, *power loss* lebih cepat hilang jika dibandingkan dengan persamaan propagasi *free space* ketika jaraknya bertambah. Namun, model ini tidak memberikan hasil yang baik untuk jarak yang terlalu dekat. Hal tersebut dikarenakan osilasi yang disebabkan oleh sifat konstruktif dan destruktif yang merupakan kombinasi dari model ini [9].

2.5. *Network Simulator*

Network Simulator 2 (NS-2) adalah sebuah alat simulasi jaringan yang bersifat *open source* di bawah GPL (*GNU Public*

Lisence) dan banyak digunakan untuk mempelajari struktur dinamis suatu jaringan komunikasi. Simulasi dapat dilakukan pada simulasi *routing*, *multicast*, dan protokol IP (UCP, TCP, RTP, jaringan nirkabel, jaringan satelit). NS-2 juga dilengkapi dengan *tools* validasi untuk memudahkan simulasi jika dibandingkan dengan menggunakan Delphi atau C++. NS-2 dapat digunakan pada sistem operasi Windows maupun Linux [10].

2.6. *Generator File Node-Movement dan Traffic-Connection Pattern*

2.6.1. *File Node-Movement (Mobility Generator)*

Untuk menghasilkan *random movement* dari *node* dalam jaringan kabel, dibutuhkan sebuah *tools* yang disebut dengan ‘setdest’. *Tools* ini dikembangkan oleh CMU (Carnegie Mellon University). *Node movement* dihasilkan dengan kecepatan gerak yang spesifik menuju lokasi acak atau spesifik dan berada kawasan yang telah ditentukan. Ketika *node* tiba di lokasi pergerakan, *node* tersebut bisa diatur untuk berhenti sementara waktu. Setelah itu, *node* terus bergerak menuju lokasi berikutnya. Lokasi *tools* ‘setdest’ ini adalah ‘~ns/indep-utils/cmu-scen-gen/setdest’.

Sebelum menjalankan simulasi, pengguna harus menjalankan *tools* ‘setdest’ ini terlebih dahulu. *Command Line* untuk ‘setdest’ terdapat pada gambar 2.1. dan keterangannya ditunjukkan dalam tabel 2.1.

```
./setdest [-v version ] [-n num_of_nodes] [-p
pausetime] [-M maxspeed] [-t simtime] [-x maxx] [-y
maxy] > [outdir/movement-file]
```

Gambar 2.1. *Command line ‘setdest’*

Tabel 2.1. Keterangan *command line* 'setdest'

Parameter	Keterangan
-v version	Versi 'setdest' simulator yang digunakan
-n num	Jumlah <i>node</i> dalam skenario
-p pausetime	Durasi sebuah <i>node</i> tetap diam setelah tiba di lokasi pergerakan. Jika nilai ini diatur ke 0, <i>node</i> tidak akan berhenti ketika tiba di lokasi pergerakan dan akan terus bergerak
-M maxspeed	Kecepatan maksimum sebuah <i>node</i> . <i>Node</i> akan bergerak pada kecepatan acak dalam rentang [0, amxspeed]
-t simtime	Waktu simulasi
-x max x	Panjang maksimum area simulasi
-y max y	Lebar maksimum area simulasi

Output dari *Command Line* 'setdest' di atas adalah *file tcl* yang berisi jumlah *node* dan mobilitas yang akan digunakan selama simulasi. Selain berisi skrip pergerakan, *file* ini juga berisi perubahan *link* dan rute.

Sebagai contoh, jika ingin membuat sebuah skenario *node-movement* yang terdiri atas 50 *node*, bergerak dengan kecepatan maksimum 25 m/s, *pausetime* 5 detik, simulasi akan berhenti setelah 300 detik, dan dengan batas topologi 400 x 400 meter², maka *command line*-nya adalah sebagai berikut:

```
./setdest -v 1 -n 50 -p 5 -M 25 -t 300 -x 400 -y 400
> scenal-25.txt
```

Gambar 2.2. *Command line* 'setdest'

Dari *command line*, *file output* disimpan ke dalam *file* "scenal-25.txt". Isi *file* dimulai dengan posisi awal *node* dan

berlanjut menetapkan *node-movement* seperti terlihat pada Gambar 2.3.

```
$ns_ at 5.000000000000 "$node_(7) setdest
293.825438692163 31.349551053882 6.619954767393"
```

Gambar 2.3. Potongan hasil output pada file “scena1-25.txt”

Potongan dari file “scena1-25.txt.” hasil *command line* pada Gambar 2.3. mendefinisikan bahwa *node* (7) pada detik ke 5.0 mulai bergerak ke arah tujuan (293.82, 31.349) dengan kecepatan 6.62 m/s. Data tersebut dapat digunakan untuk mengubah arah dan kecepatan gerak dari *mobile node*. Arah untuk *General Operations Director 1* (GOD) juga ada dalam *file node-movement*. Objek "GOD" berisi informasi global tentang keadaan dari lingkungan jaringan dan *node* di sekitarnya. Akan tetapi isi dari *file* “GOD” tidak boleh diketahui oleh setiap bagian dalam simulasi.

Objek “GOD” dalam simulasi hanya digunakan untuk menyimpan *array* dari jumlah *hop* terpendek yang diperlukan suatu *node* untuk mencapai *node* lain. Selama simulasi, jumlah *hop* tidak akan dihitung oleh “GOD” karena akan memakan waktu. Jumlah *hop* akan dihitung di akhir simulasi. Potongan “GOD” untuk pola pergerakan terdapat pada gambar 2.4.

```
$ns_ at 5.000513200170 "$god_ set-dist 30 45 1"
```

Gambar 2.4. Potongan “GOD” dalam file “scena1-25.txt.”

Potongan tersebut menunjukkan bahwa jarak terpendek antara *node* 30 dan *node* 45 menjadi 1 *hop* pada waktu 5 detik. Program ‘setdest’ menghasilkan *file node-movement* menggunakan algoritma *random way point* [11].

2.6.2. File Traffic-Connection Pattern

Untuk menghasilkan suatu alur *traffic* yang acak, dapat digunakan skrip *tcl* yang disebut dengan ‘*cbrgen*’ yang menghasilkan *traffic load* atau beban trafik. Beban tersebut dapat dipilih antara TCP atau CBR. Skrip ini terdapat dalam direktori ‘CMU-scen-gen’ yang terletak dalam “~ns/indep-utils/cmu-scen-gen”. *Command line* untuk *cbrgen.tcl* terdapat pada gambar 2.5 dan dijelaskan pada tabel 2.2.

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

Gambar 2.5. Command line ‘cbrgen.tcl’

Tabel 2.2. Keterangan command line ‘cbrgen.tcl’

Parameter	Keterangan
-type cbr tcp	Jenis <i>traffic</i> yang digunakan TCP atau CBR
-nn nodes	Jumlah <i>node</i>
-s seed	<i>Random seed</i>
-mc connections	Jumlah koneksi
-rate rate	Jumlah paket per detik. Pada CBR, panjang paket adalah tetap yaitu sebesar 512 bytes selama simulasi

Pada CBR, *data rate* dapat dihitung sebagai berikut:

Data Rate (bits/second) = 512 bytes*8 bits/bytes * rate (packets/second defined in "cbrgen")

Pada gambar 2.6, ditunjukkan *command line* untuk membuat sebuah *file* koneksi cbr antara 50 *node*, memiliki nilai *seed* 1.0, *max connection* 1, dan jumlah paket per detik sebanyak 1.0. *Output* dari *command line* tersebut disimpan pada *file* “cbrtest.txt”.

```
ns cbrgen.tcl -type cbr -nn 50 -seed 1.0 -mc 1 -rate
1.0 > cbrtest.txt
```

Gambar 2.6. Contoh command line cbrgen.tcl

Dari file “cbrtest.txt”, dihasilkan koneksi CBR yang terlihat seperti pada Gambar 2.7..

```
#
# 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_ 1
$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"
```

Gambar 2.7. Koneksi cbr pada file “cbrtest.txt”

Agent yang digunakan adalah UDP. Koneksi UDP tersebut adalah *setup* antara *node* 1 dan 2. Jumlah sumber UDP dipilih antara *node* 0-50 [11].

2.7. NS-2 Trace File

Setelah melakukan *running* pada NS-2, akan didapatkan *file output* berupa NS-2 Trace File yang berekstensi .tr. File tersebut berisi *log* pengiriman dan penerimaan paket yang terjadi selama simulasi. Tercatat jenis paket sesuai jenis protokol *routing* yang digunakan. Contoh pengiriman data paket pada NS-2 Trace File dapat dilihat pada Gambar 2.8.

```
s 2.556838879 1 AGT --- 50 cbr 512 [0 0 0 0] ---
---- [1:0 2:0 32 0] [0] 0 1
```

Gambar 2.8. Trace pengiriman paket data

Pada kolom pertama, huruf “s” menandakan pengiriman paket (*send*). Kolom ke dua berisi waktu pengiriman paket pada detik 2.55. Kolom ke tiga merupakan *node* tempat *event* terjadi yaitu pada *node* 1. Kolom ke empat bertuliskan AGT yang menandakan pengiriman paket data. Kolom ke lima merupakan tempat terjadinya *event* spesial seperti *collision*. Kolom ke enam merupakan *id* unik paket. Kolom ke tujuh berisi tipe paket yang dikirimkan yaitu *cbr*. Kolom ke delapan merupakan ukuran paket dalam *byte* yaitu 512.

Untuk penerimaan paket tidak jauh berbeda dengan pengiriman paket. Yang menjadi pembeda adalah kolom pertama dimana penerimaan paket berisi huruf “r” yang menandakan penerimaan paket (*receive*). Untuk kolom lainnya berisi informasi yang sama dengan pengiriman. Contoh penerimaan paket data dapat dilihat pada Gambar 2.9.

```
r 2.556838879 1 AGT --- 50 cbr 512 [0 0 0 0] ---
---- [1:0 2:0 32 0] [0] 0 1
```

Gambar 2.9. Trace penerimaan paket data

2.8. Awk

Awk adalah sebuah pemrograman seperti pada *shell* atau C yang memiliki karakteristik yaitu sebagai *tools* yang cocok *filter* / manipulasi. Awk adalah penggabungan dari nama lengkap sang author, yaitu : Alfred V. Aho, Peter J. Weinberger dan Brian W. Kernighan. Awk atau juga disebut Gawk (GNU awk), yaitu bahasa pemrograman umum dan *utility* standard POSIX 1003.2. Jika kecepatan merupakan hal yang penting, awk adalah bahasa yang sangat sesuai. Awk sangat baik untuk manipulasi *file* teks. Secara umum bahasa pemrograman awk dapat digunakan untuk

mengelola *database* sederhana, membuat laporan, memvalidasi data, menghasilkan indeks dan menampilkan dokumen, membuat algoritma yang digunakan untuk mengubah bahasa komputer ke bahasa lainnya. Dengan kata lain awk menyediakan fasilitas yang dapat memudahkan untuk memecah bagian data untuk proses selanjutnya, mengurutkan data dan menampilkan komunikasi jaringan yang sederhana.

Fungsi dasar awk adalah untuk mencari *file* per baris (atau unit teks lain) yang berisi pola tertentu. Ketika suatu baris sesuai dengan pola, awk melakukan aksi yang khusus pada baris tersebut. awk tetap memproses baris *input* sedemikian hingga mencapai akhir baris *input*. Program pada awk berbeda dari program di kebanyakan bahasa lain, karena program awk bersifat “data-driven” yang mana diperlukan pendeskripsian data yang dikehendaki untuk bekerja dan kemudian apa yang akan dilakukan saat data tersebut ditemukan. Kebanyakan bahasa lainnya bersifat “procedural” maka dari itu diharuskan mendeskripsikannya secara detail setiap langkah program yang harus dijalankan. Ketika bekerja dengan bahasa prosedural, biasanya sangat sulit untuk mendeskripsikan data yang hendak diproses oleh program. Oleh karena itu, program awk sering kali terasa lebih mudah untuk ditulis dan dibaca [11].

Pada tugas akhir ini AWK digunakan untuk membuat *script* dalam penghitungan *Packet Delivery Ratio* (PDR), *End-to-End Delay* (E2D), dan *Routing Overhead* (RO) dari hasil *trace* NS-2.

[Halaman ini sengaja dikosongkan]

BAB III

PERANCANGAN SISTEM

Pada bab ini dijelaskan mengenai dasar perancangan dari perangkat lunak yang akan dibangun dalam Tugas Akhir ini. Secara khusus akan dibahas mengenai deskripsi umum sistem, perancangan skenario, alur, serta gambaran implementasi sistem yang diterapkan pada *Network Simulator 2 (NS-2)*.

3.1. Deskripsi Umum

Dalam Tugas Akhir ini, dilakukan analisis performa model tranmisi Nakagami dan TwoRayGround pada MANET. Pembuatan skenario MANET menggunakan *Mobility Generator* yang bersifat *Random Way Point* dengan cara *men-generate file node-movement (mobility generation)* dan membuat koneksi antar *node* menggunakan *file traffic-connection pattern* yang telah ada pada *Network Simulator-2 (NS-2)*. Skenario tersebut kemudian dijalankan pada NS-2 dengan menggunakan protokol *routing DSDV* pada sistem operasi Linux Ubuntu.

Terdapat tiga kecepatan maksimum yang berbeda untuk pergerakan dari satu *node* ke *node* lain di setiap skenario, yaitu 5 m/s, 10 m/s, dan 15 m/s. Uji coba akan menghasilkan sebuah *trace file* yang nantinya digunakan untuk menganalisis *Packet Delivery Ratio (PDR)*, *End-to-End Delay (E2D)*, dan *Routing Overhead (RO)* yang kemudian digunakan untuk menganalisis performa model.

3.2. Perancangan Skenario

Perancangan skenario dimulai dengan membuat skenario pada *mobility generation* yang bersifat *random way point*. Setelah itu dibuat koneksi dengan menggunakan *file traffic-connection* yang sudah ada di dalam NS-2. Untuk melihat pergerakan *node*, pembuatan skenario uji coba dibedakan berdasarkan kecepatan maksimum dan dibagi menjadi tiga, yaitu 5 m/s, 10 m/s, dan 15 m/s. Penjelasan untuk perancangan skenario *pada mobility*

generator dan pembuatan koneksi antar *node* adalah sebagai berikut:

3.2.1. Skenario *Node-Movement (Mobility Generation)*

Skenario *mobility generation* dibuat dengan melakukan *generate file node-movement* atau bisa disebut '*setdest*'. Setelah menjalankan perintah '*setdest*', akan didapat *file output* berupa *file text* yang nantinya digunakan untuk simulasi.

Tabel 3.1. Parameter skenario *node-movement*

No.	Parameter	Spesifikasi
1	Jumlah <i>Node</i>	50
2	Waktu Simulasi	100 detik
3	Area	510 m x 510 m
3	Kecepatan Maksimal	- 5 m/s - 10 m/s - 15 m/s
5	Sumber <i>Traffic</i>	CBR
6	Waktu Jeda (detik)	10
7	Ukuran Paket	512 bytes
8	<i>Rate</i> Paket	1 paket per detik
9	Jumlah maksimal koneksi	1
10	Model mobilitas yang digunakan	<i>Random Way Point</i>

3.2.2. *Traffic-Connection Pattern Generation*

Traffic-Connection dibuat dengan menjalankan program "*cbrogen.tcl*" yang sudah terdapat dalam NS-2. Nantinya program ini akan menghasilkan *file output* dalam bentuk *file text* dan digunakan untuk menghubungkan *node* satu dengan *node* lain pada skenario selama simulasi.

Tabel 3.2. Parameter *traffic-connection pattern*

No.	Parameter	Spesifikasi
1	-type cbr tcp	CBR
2	-nn <i>nodes</i>	10
3	-s <i>seed</i>	1.0
4	-mc <i>connections</i>	1
5	-rate <i>rate</i>	1
6	<i>Agent</i>	UDP

3.3. Perancangan Simulasi pada NS-2

Perancangan simulasi dilakukan dengan menggabungkan skenario mobilitas, *traffic-connection pattern*, dan skrip tcl. Skrip tcl diberikan parameter-parameter untuk percobaan simulasi. Pada tabel 3.3. dapat dilihat perancangan sistem MANET yang akan digunakan.

Tabel 3.3. Parameter simulasi pada NS-2

No.	Parameter	Spesifikasi
1	<i>Network simulator</i>	NS- 2.35
2	<i>Routing Protocol</i>	DSDV
3	Waktu simulasi	100 detik
4	Waktu Pengiriman Paket Data	0 – 100 detik
5	Area simulasi	510 m x 510 m
6	Banyak <i>node</i>	50
7	Radius transmisi	100 m
8	Tipe koneksi	UDP
9	Tipe data	<i>Constant Bit Rate (CBR)</i>
10	<i>Source / Destination</i>	Statik (<i>Node 1 / Node 2</i>)

No.	Parameter	Spesifikasi
11	Kecepatan generasi paket	1 paket per detik
12	Ukuran paket data	512 bytes
13	Protokol MAC	IEEE 802.11
14	Mode Propagasi	- <i>Nakagami</i> - <i>TwoRayGround</i>
15	Tipe Antena	<i>OmniAntenna</i>
16	Tipe Interface Queue	<i>Droptail/PriQueue</i>
17	Tipe Peta	MANET (<i>random way point</i>)
18	Tipe Kanal	<i>Wireless channel</i>
19	Tipe <i>trace</i>	<i>Old Wireless Format Trace</i>

3.4. Perancangan Metrik Analisis

Pada Tugas Akhir ini, metrik yang akan dianalisis adalah *Packet Delivery Ratio* (PDR), *End-to-End Delay* (E2D), dan *Routing Overhead* (RO). Penjelasan dari masing-masing metrik analisis adalah sebagai berikut:

3.4.1. *Packet Delivery Ratio* (PDR)

Metrik PDR ini didapatkan dengan membandingkan paket yang dikirim dengan paket yang diterima. PDR dihitung dengan menggunakan persamaan (1) dimana *received* adalah jumlah paket data yang berhasil diterima dan *sent* adalah banyaknya paket data yang dikirim. Nilai PDR dinyatakan dalam bentuk persen.

$$PDR = \frac{received}{sent} \times 100$$

(1)

3.4.2. *End-to-End Delay (E2D)*

Nilai *End-to-End Delay* (E2D) didapat dengan menghitung rata-rata *delay* antara waktu paket ketika diterima dan dikirim. E2D dihitung dengan menggunakan persamaan (2) dimana $t_{received[i]}$ adalah waktu penerimaan paket, $t_{sent[i]}$ adalah waktu pengiriman paket, i adalah urutan / id ke- i , dan $sent$ adalah banyaknya paket data yang dikirimkan.

$$E2D = \frac{\sum_{i \leq sent}^{i=0} t_{received[i]} - t_{sent[i]}}{sent}$$

(2)

3.4.3. *Routing Overhead (RO)*

Nilai *Routing Overhead* adalah jumlah paket kontrol *routing* yang ditransmisikan per paket data yang terkirim ke tujuan selama simulasi terjadi. RO dihitung berdasarkan jumlah paket *routing* yang ditransmisikan. Pada *trace file*, RO ditandai dengan baris yang berisi paket dengan tipe *send (s)* / *forward (f)* dan *header* paket dari protokol *routing* DSDV.

[Halaman ini sengaja dikosongkan]

BAB IV IMPLEMENTASI

Bab ini membahas tentang implementasi dari perancangan perangkat lunak. Implementasi yang dijelaskan meliputi lingkungan pembangunan perangkat lunak, implementasi skenario, implementasi simulasi pada NS-2, dan implementasi matrik analisis.

4.1. Lingkungan Pembangunan Perangkat Lunak

Pembangunan perangkat lunak dilakukan pada lingkungan pengembangan sebagai berikut:

4.1.1. Lingkungan Perangkat Lunak

Perangkat lunak yang digunakan untuk pengembangan sistem adalah:

- Sistem Operasi Ubuntu 14.04 LTS 64-bit untuk lingkungan NS-2;
- *Network Simulator 2* versi 2.35.
- *Patch DSDV* versi 1.0

4.1.2. Lingkungan Perangkat Keras

Spesifikasi perangkat keras yang digunakan untuk implementasi perangkat lunak Tugas Akhir adalah:

- *Processor* Intel(R) Core i7-2630QM CPU @2.90GHz;
- *Storage* 1TB;
- RAM 4 GB DDR3.

4.2. Implementasi Skenario

Implementasi skenario mobilitas MANET dipelajari dalam kondisi pada beban *traffic* dan mobilitas/pergerakan *node*-nya. Dua model yang digunakan untuk studi simulasi pada jaringan MANET adalah *mobility generation*, yang digunakan untuk mempelajari pengaruh mobilitas dari *node* pada kinerja keseluruhan jaringan dan *traffic-connection generation*, yang digunakan untuk

mempelajari pengaruh beban *traffic* pada jaringan. Implementasi skenarionya adalah sebagai berikut:

4.2.1. Skenario *File Node-Movement (Mobility Generation)*

Pada saat simulasi skenario, digunakan *tools generate default* yang dimiliki oleh NS-2 yaitu ‘setdest’ untuk *mobility generation*. Program ‘setdest’ ini akan menghasilkan *file node-movement* dengan menggunakan algoritma *Random Way Point*. *File* skenario *node-movement* tersebut akan digunakan di setiap simulasi yang ditandai dengan jeda waktu. Simulasi dilakukan dengan menggunakan kecepatan maksimum yang berbeda. Hal tersebut bertujuan untuk mempelajari efek mobilitas dari *node*. Untuk menghasilkan gerakan acak pada *node*, digunakan *command line* yang dapat dilihat pada Gambar 4.1.

```
./setdest [-v version ] [-n num_of_nodes] [-p
pausetime] [-M maxspeed] [-t simtime] [-x maxx] [-y
maxy] > [outdir/movement-file]
```

Gambar 4.1. Command line ‘setdest’

Pada *command line* di atas, ditunjukkan secara berturut-turut versi ‘setdest’ *simulator*, jumlah *node* dalam skenario, *pausetime* (waktu jeda), kecepatan maksimum, waktu simulasi, panjang area, lebar area, dan *output* dari *command line* tersebut. *File output* yang dihasilkan tersimpan dalam direktori yang sama dengan ‘setdest’ pada direktori “ns/indep-utils/cmu-scenegen/setdest”. Implementasi *command line* ‘setdest’ dengan berbagai kecepatan maksimal yang berbeda dan *node* sebanyak 50 dapat dilihat pada gambar 4.2. Setiap kecepatan maksimum dibuat 10 *file* untuk satu protokol *routing* (DSDV) dan satu model propagasi (*Nakagami* atau *TwoRayGround*).

```

./setdest -v 1 -n 50 -p 10 -M 5 -t 100 -x 510 -y 510
> scenal-5.txt
./setdest -v 1 -n 50 -p 10 -M 10 -t 100 -x 510 -y
510 > scenal-10.txt
./setdest -v 1 -n 50 -p 10 -M 15 -t 100 -x 510 -y
510 > scenal-15.txt

```

Gambar 4.2. Implementasi *command line* 'setdest'

Implementasi *command line* 'setdest' di atas menunjukkan versi 'setdest' yang digunakan adalah 1, jumlah *node* sebanyak 50, *pausetime* selama 10 detik, kecepatan maksimal (5 m/s, 10 m/s, 15 m/s), waktu simulasi selama 100 detik, panjang area simulasi 510 meter, lebar area simulasi 510 meter, dan *file output* (scenal-5.txt, scenal-10.txt, scenal-15.txt).

Penempatan setiap *node* dalam simulasi terdapat pada sumbu X, Y, dan Z. Penempatan tiap *node* dapat dilihat pada *file hasil generate* 'setdest' seperti pada gambar 4.3.

```

#
# nodes: 50, pause: 10.00, max speed: 5.00, max x:
510.00, max y: 510.00
#
$node_(0) set X_ 135.750954250386
$node_(0) set Y_ 249.694883657278
$node_(0) set Z_ 0.000000000000
$node_(1) set X_ 182.814657191152
$node_(1) set Y_ 191.562026090664
$node_(1) set Z_ 0.000000000000
$node_(2) set X_ 74.711349536656
$node_(2) set Y_ 204.677829249526
$node_(2) set Z_ 0.000000000000

```

Gambar 4.3. Posisi *node* dalam X, Y, dan Z pada file "scena1-5.txt"

Selama simulasi dijalankan, diberikan posisi awal untuk *node* dan pergerakan *node* berikutnya. Dari potongan *file* "scena1-5.txt", baris pertama didefinisikan untuk *node* (0) pada detik ke 10 bergerak ke arah tujuan (59.64, 493.03) dengan kecepatan 2.15

m/s. Baris perintah ini dapat digunakan untuk mengubah arah dan kecepatan pada pergerakan *node*.

```

$ns_ at 10.000000000000 "$node_(0) setdest
59.642927959964 493.035963718193 2.153551659533"
$ns_ at 10.000000000000 "$node_(1) setdest
503.468798876626 222.357522370379 2.402489064196"
$ns_ at 10.000000000000 "$node_(2) setdest
24.523026847930 213.164639586640 1.639526471501"
$ns_ at 10.000000000000 "$node_(3) setdest
439.600772572905 374.604226364214 3.912637298654"
$ns_ at 10.000000000000 "$node_(4) setdest
375.889599010124 490.336170338542 0.207585460315"
$ns_ at 10.000000000000 "$node_(5) setdest
234.751915917359 192.108064416907 4.409341618219"
$ns_ at 10.000000000000 "$node_(6) setdest
106.533621069741 117.854338359018 1.091111771984"
$ns_ at 10.000000000000 "$node_(7) setdest
116.550187238594 480.254163484291 3.971165658674"
$ns_ at 10.000000000000 "$node_(8) setdest
295.669024935727 228.224030709965 1.546599362440"
$ns_ at 10.000000000000 "$node_(9) setdest
446.675789854155 197.584119226666 0.750016360777"
$ns_ at 10.000000000000 "$node_(10) setdest
331.137126436579 28.485312162653 4.346742651919"

```

Gambar 4.4. Pergerakan *node* pada file "scena1-5.txt"

Pada file "scena1-5.txt" juga terdapat penentuan GOD dari setiap *node*. GOD adalah singkatan dari *General Operations Director* yang berfungsi untuk menyimpan informasi global tentang jumlah dan pergerakan *node*. Saat simulasi, GOD menyimpan *array* yang berisi *hop* terpendek yang diperlukan sebuah *node* untuk mencapai *node* lainnya.


```

$god_ set-dist 0 1 1
$god_ set-dist 0 2 1
$god_ set-dist 0 3 2
$god_ set-dist 0 4 2
$god_ set-dist 0 5 1

```

Gambar 4.5. GOD pada file "scena1-5.txt"

Informasi dimuat ke objek GOD dari *file node-movement*. Pada gambar 4.6., baris pertama menunjukkan jalan terpendek antara *node 4* dan *node 37* adalah 2 *hop* pada detik 10.22392734041.

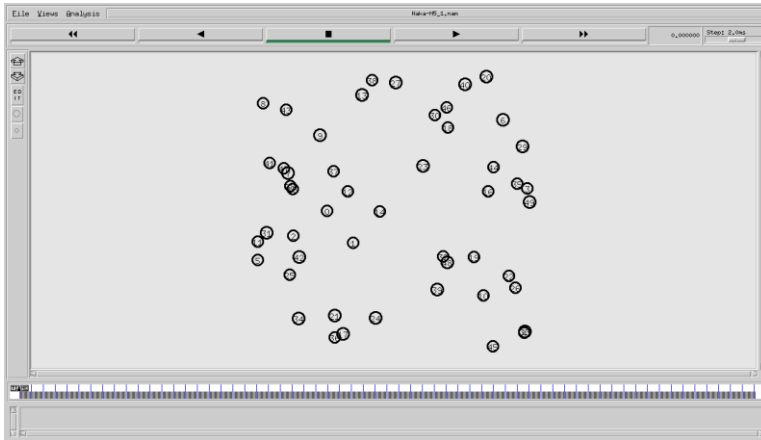
```

$ns_ at 10.223927349041 "$god_ set-dist 4 37 2"
$ns_ at 10.223927349041 "$god_ set-dist 22 37 2"
$ns_ at 10.223927349041 "$god_ set-dist 26 37 2"
$ns_ at 10.223927349041 "$god_ set-dist 32 37 1"
$ns_ at 10.223927349041 "$god_ set-dist 33 37 2"
$ns_ at 10.223927349041 "$god_ set-dist 37 45 2"
$ns_ at 10.341944268871 "$god_ set-dist 7 22 2"
$ns_ at 10.341944268871 "$god_ set-dist 9 22 2"
$ns_ at 10.341944268871 "$god_ set-dist 13 22 2"
$ns_ at 10.341944268871 "$god_ set-dist 15 22 2"

```

Gambar 4.6. Access Point

Gambar 4.7. menunjukkan animasi untuk posisi dari *node* yang didapat dengan menjalankan *file* "Naka_1-5.nam" setelah menjalankan tcl yang menghasilkan *file* .tr dan .nam.



Gambar 4.7. Hasil animasi untuk posisi *node*

4.2.2. File Traffic-Connection Pattern Generation

Pergerakan antar *node* dapat diatur menggunakan skrip *traffic-scenario generator*. Skrip tersebut terdapat dalam direktori “~/ns/indep-utils/cmu-scen-gen/” dan disimpan dalam bentuk *file* *cbrgen.tcl*. Untuk menjalankan skrip ini, ditentukan terlebih dahulu tipe *traffic-connection*-nya (CBR atau TCP), banyak *node*, koneksi maksimum dalam jaringan, *random seed*, dan *rate* yang nilai kebalikannya digunakan untuk menghitung waktu interval antar paket CBR. Hasil skrip tersebut kemudian dalam sebuah *file* *traffic*. *Command line* untuk menjalankan skrip ‘*cbrgen*’ dapat dilihat pada gambar 4.7.

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

Gambar 4.8. *Command line* ‘*cbrgen.tcl*’

Contoh implementasi *command line* ‘*cbrgen.tcl*’ antara 10 *node*, *seed* 1, koneksi maksimum 1, dan jumlah paket per detik 1, dan disimpan dalam *file* “*cbrtest.txt*” dapat dilihat pada gambar 4.8. *File* tersebut nantinya akan digunakan untuk simulasi NS-2.

```
ns cbrgen.tcl CBR -nn 10 -seed 1.0 -mc 1 -rate 1 >
cbrtest.txt
```

Gambar 4.9. Implementasi *command line* 'cbrgen.tcl'

Kemudian Gambar 4.9. menunjukkan isi dari *file* “cbrtest.txt”. *file* tersebut menghasilkan koneksi CBR dan menggunakan *Agent* UDP. Koneksi UDP di sini merupakan konfigurasi antara *node* ke-1 dan 2.

4.3. Implementasi Simulasi pada NS-2

Untuk melakukan simulasi MANET pada NS-2, skenario *node-movement (mobility generation)* dan *traffic-connection generation* yang disimpan dalam bentuk txt diberikan parameter yang sesuai dengan perancangan agar dapat dijalankan pada NS-2. Parameter-parameter tersebut dibuat dengan menggunakan bahasa tcl/otcl.

Gambar 4.10. menunjukkan skrip konfigurasi awal parameter-parameter yang digunakan untuk menjalankan MANET pada NS-2. Parameter-parameter tersebut disimpan pada *file* tcl. Dalam *file* tersebut, baris pertama merupakan konfigurasi tipe *channel* yang digunakan yaitu *Wireless Channel*. Baris kedua merupakan tipe propagasi yang digunakan yaitu *Nakagami*. Ketika simulasi dijalankan untuk propagasi *TwoRayGround*, maka pada baris ke dua tipe propagasi diganti dengan *TwoRayGround*. Tipe Mac yang digunakan adalah Mac 802.11. Pada *file* tersebut juga berisi konfigurasi tipe *queue* dari *interface*, tipe *link layer*, tipe *antenna* dan jumlah maksimal *packet* pada *interface queue*. Baris ke-9 sampai baris ke-17 berturut-turut merupakan koordinat x serta koordinat y sebesar 510 meter, jumlah maksimal paket dan *node* yaitu 50 *node*, besarnya *seed* yaitu 0.0, protokol *routing* yang digunakan yaitu DSDV, waktu simulasi diakhiri pada detik ke-100, *file traffic-connection* yang digunakan yaitu cbrtest.txt dan terakhir ialah *file* skenario *node-movement (mobility generation)* yang digunakan adalah “scena1-5.txt”.

```

# nodes: 10, max conn: 1, send rate: 1, 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_ 1
$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"
#

```

Gambar 4.10. Output file "cbrtest.txt"

```

set val(chan) Channel/WirelessChannel;
set val(prop) Propagation/Nakagami;
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 opt(x) 510;
set opt(y) 510;
set val(ifqlen) 50;
set val(nn) 50;
set val(seed) 0.0;
set val(adhocRouting) DSDV;
set val(stop) 100
set val(cp) "cbrtest.txt";
set val(sc) "scen1-5.txt";

```

Gambar 4.11. Konfigurasi awal parameter NS-2

Gambar 4.11. menunjukkan pengaturan dari *transmission range* yang digunakan saat simulasi. Nilai yang diubah ialah *RXThresh_* (*Receiver Sensitivity Threshold*). Nilai $1.42681e-08$ pada variabel tersebut memiliki artian bahwa *range* atau jangkauan yang dapat dicapai adalah sejauh 100 meter.

```
Phy/WirelessPhy set RXThresh_ 1.42681e-08
```

Gambar 4.12. Konfigurasi *transmission range* NS-2

Pada gambar 4.12., ditunjukkan skrip untuk pengaturan variabel global. *set ns* untuk pembuatan simulator baru, *set tracefc* untuk pengaturan nama *trace file* yang berekstensi *.tr*, dan *set namtrace* untuk pengaturan nama *file network animator* yang berekstensi *.nam*. Kedua file tersebut nantinya merupakan *output* dari program tcl ini.

```

# 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 Naka-M5_1.tr w]
set namtrace    [open Naka-M5_1.nam w]

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

# set up topology object
set topo        [new Topography]
$topo load_flatgrid $opt(x) $opt(y)

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

#global node setting
$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) \
                -channelType $val(chan) \
                -topoInstance $topo \
                -agentTrace ON \
                -routerTrace ON \
                -macTrace OFF \
                -movementTrace ON \

for {set i 0} {$i < $val(nn)} {incr i} {
    set node_($i) [$ns_ node]
    $node_($i) random-motion 0 ;#disable
    random motion}

```

Gambar 4.13. Konfigurasi *trace file* dan pergerakan node NS-2

Set topo merupakan pengaturan untuk objek topografi berdasarkan luas koordinat yang telah dikonfigurasi sebelumnya. *Create-god* dan *node-config -channelType* adalah konfigurasi yang dilakukan pada *node* yang telah dibuat. Pada *create-god*, dilakukan implementasi *node* yang dibuat sesuai dengan parameter pada *set-val(nn)* sedangkan untuk *node-config -channelType* merupakan konfigurasi *node* sesuai dengan parameter pada gambar 4.10. Untuk membuat pergerakan *node*, dilakukan perulangan. *Node* yang telah dibuat tidak dapat bergerak secara acak karena pergerakan *node* merupakan *trace file* yang sebelumnya dihasilkan oleh *mobility generator*.

Bagian akhir dari skrip yang digunakan untuk penempatan awal *node* untuk skenario *node-movement (mobility generator)*. File skenario *node-movement (mobility generation)* dan *traffic-connection pattern* akan dipanggil untuk memulai pengiriman paket data yang dimulai pada detik ke-0 dan diakhiri pada detik ke-100. Potongan akhir skrip dapat dilihat pada gambar 4.13.

Gambar 4.14. adalah contoh hasil *running file* “Naka-M5 1.tcl”.

```

andre@ANDRE-PC: ~/ns-allinone-2.35/ns-2.35/Nakagami/Nakagami-M5/1
andre@ANDRE-PC:~/ns-allinone-2.35/ns-2.35/Nakagami/Nakagami-M5/1$ ns Naka-M5_1.t
num_nodes is set 50
INITIALIZE THE LIST xListHead
Loading connection pattern...
Loading scenario file...
Starting Simulation...
channel.cc:sendUp - Calc highestAntennaZ_ and distCST_
highestAntennaZ_ = 1.5, distCST_ = 17976931348623157081452742373170435679807056
95351438246423432132688946418276846754670353751698604991057655128207624549009038
797826204144723168738177180919299881250404026184124858368.0
SORTING LISTS ..DONE!
NS EXITING...
andre@ANDRE-PC:~/ns-allinone-2.35/ns-2.35/Nakagami/Nakagami-M5/1$

```

Gambar 4.14 Contoh hasil *running file* tcl

```

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

# Define traffic model
puts "Loading scenario file..."
source $val(sc)

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

    # 20 defines the node size in nam, must adjust
    # according to your scenario
    # The function must be called after mobility
    # model is defined

    $ns_ initial_node_pos $node_($i) 20
}

# 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) "stop"
$ns_ at $val(stop).0002 "puts \"NS EXITING...\" ;
$ns_ halt"

puts $tracefd "M 0.0 nn $val(nn) x $opt(x) y
$opt(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..."
$ns_ run

```

Gambar 4.15. Konfigurasi pengiriman paket data NS-2

4.4. Implementasi Metrik Analisis

Setelah menjalankan *file tcl*, akan didapat sebuah *trace file* yang berekstensi *.tr*. *File* tersebut nantinya akan dianalisis dengan tiga metrik, yaitu *Packet Delivery Ratio* (PDR), *End-to-End Delay* (E2D), dan *Routing Overhead* (RO). Implementasi dari masing-masing metrik menggunakan pemrograman AWK dan dijelaskan pada subbab berikutnya.

4.4.1. *Packet Delivery Ratio* (PDR)

Nilai PDR didapat dengan menghitung jumlah paket data terkirim yang dilakukan oleh *node 1* dan jumlah paket data yang diterima oleh *node 2* pada satu *trace file*. Penjumlahan paket terkirim dilakukan apabila pada baris *trace* yang bersangkutan mengandung semua kondisi dimana kolom pertama mengandung huruf “s” yang menandakan *send packet*, kolom ke-3 menunjukkan bahwa *node* yang melakukan pengiriman adalah *node 1*, kolom ke-4 dan kolom ke-7 mengandung huruf “AGT” dan “cbr”. Hal itu menandakan pengiriman paket yang dilakukan adalah pengiriman paket data. Pencatatan jumlah paket yang diterima dilakukan apabila pada baris *trace* yang bersangkutan mengandung semua kondisi dimana kolom pertama mengandung huruf “r” yang menandakan *received packet*, kolom ke-3 menunjukkan bahwa *node* yang menerima paket data adalah *node 2*, kolom ke-4 dan kolom ke-7 mengandung huruf “AGT” dan “cbr” yang menandakan penerimaan paket yang diterima adalah penerimaan paket data. Penghitungan dilakukan sampai baris terakhir *trace file*, dan hasilnya adalah hasil hitung nilai PDR simulasi skenario.

Pseudocode PDR ditunjukkan Gambar 4.15. dan implementasinya dapat dilihat pada Lampiran.

```

ALGORITMA PDR(trace file)
//Input: trace file simulasi skenario
//Ouput: jumlah packet sent, packet received, dan
//      PDR
BEGIN (
sent ← 0
recv ← 0
recv_id ← 0
pdr ← 0)

#count packet send
(if ($1 == "s" and $3 == "_1_" and $4 == "AGT" and
    $7 == "cbr" )
    sent +1;
)

#count packet receive
if ($1 == "r" and $3 == "_0_" and $4 == "AGT" and
    $7 == "AGT")
    recv +1;
)
END (
pdr ← ( recv / sent ) * 100
print sent
print recv
print pdr)

```

Gambar 4.16. Pseudocode PDR

Contoh perintah untuk menghasilkan nilai PDR dari *trace file* model propagasi *Nakagami* dengan protokol DSDV dan kecepatan maksimal perpindahan *node* sebesar 5 m/s adalah pada Gambar 4.16.

```
awk -f pdr.awk Naka-M5_1.tr
```

Gambar 4.17. Perintah untuk menjalankan “pdr.awk”

Contoh *output* dari menjalankan skrip tersebut ditunjukkan pada Gambar 4.17.

```
Transmitted packet(s) = 95
Received packet(s) = 45
Packet Delivery Ratio = 47.3684 %
```

Gambar 4.18. Hasil *running pdr.awk*

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

Perhitungan E2D dilakukan dengan menghitung rata-rata *delay* antara waktu paket ketika diterima dan dikirim. selisih waktu paket data terkirim yang dilakukan oleh *node* 1 dan waktu paket data diterima oleh *node* 2 di dalam satu *trace file*. Pencatatan waktu paket terkirim pada kolom ke-2 dilakukan apabila pada baris *trace* yang bersangkutan mengandung semua kondisi yaitu kolom pertama mengandung huruf “s” yang menandakan *send packet*, kolom ke-3 menunjukkan *node* yang melakukan pengiriman adalah *node* 1, kolom ke-4 dan kolom ke-7 mengandung huruf “AGT” dan “cbr” yang menunjukkan pengiriman paket yang dilakukan adalah pengiriman paket data. Perhitungan waktu dan pencatatan ID serta jumlah paket diterima dilakukan apabila baris *trace* yang bersangkutan mengandung kondisi dimana yaitu kolom pertama mengandung huruf “r” yang menandakan *received packet*, kolom ke-3 menunjukkan *node* yang menerima paket adalah *node* 2, kolom ke-4 dan kolom ke-7 mengandung huruf “AGT” dan “cbr” yang menunjukkan penerimaan paket yang adalah penerimaan paket data. Perhitungan dilakukan sampai baris terakhir *trace file*.

Pseudocode E2D ditunjukkan pada Gambar 4.18. dan implementasinya dapat dilihat pada Lampiran.

```

ALGORITMA E2D(trace file)
//Input: trace file simulasi skenario
//Ouput: jumlah packet received, total delay,
dan // E2D
BEGIN (
  for i in pkt_id
    pkt_id[i] ← 0
  for i in pkt_sent
    pkt_sent[i] ← 0
  for i in pkt_rcv
    pkt_rcv[i] ← 0
  delay = avg_delay ← 0
  rcv ← 0
  rcv_id ← 0)

  (if ($1 == "s" and $3 == "_1_" and $4 == "AGT" and
    $7 == "cbr")
    pkt_sent[$6] ← $2

  if ($1 == "r" and $3 == "_2_" and $4 == "AGT"
    and $7 == "cbr" and rcv_id != $6 )
    rcv + 1
  rcv_id ← $6
    pkt_rcv[$6] ← $2;
  )

END (
  for i in pkt_rcv
    delay += pkt_rcv[i] - pkt_sent[i]
  avg_delay ← delay / rcv;
  print rcv
  print delay
  print avg_delay

```

Gambar 4.19. Pseudocode E2D

Contoh perintah untuk analisis E2D dari *trace file* model propagasi *Nakagami* dengan protokol DSDV dan kecepatan maksimal perpindahan *node* sebesar 5 m/s adalah pada Gambar 4.19 dan hasilnya dapat dilihat pada Gambar 4.20.

```
awk -f e2e.awk Naka-M5_1.tr
```

Gambar 4.20. Perintah untuk menjalankan “e2e.awk”

```
Total Packet(s) Receive = 45
Total Delay = 0.608213 second
Average Packet Delivery Delay = 0.0135158 second
```

Gambar 4.21. Hasil *running* “e2e.awk”

4.4.3. *Routing Overhead (RO)*

Implementasi perhitungan metrik *Routing overhead* DSDV dihitung apabila kondisi-kondisi yang ada terpenuhi. Kondisi tersebut yaitu pada saat kolom pertama diawali dengan huruf “s” yang berarti *send packet* atau huruf “f” yang berarti *forward packet*, kolom ke-4 mengandung huruf “RTR” yang berarti paket *routing* dan kolom ke-7 mengandung “DSDV” yang berarti paket *routing* DSDV.

Pseudocode untuk RO dapat dilihat pada Gambar 4.21 dan implementasinya dapat dilihat pada Lampiran.

```
ALGORITMA RO-DSDV(trace file)
//Input: trace file simulasi skenario
//Output: jumlah routing overhead protokol DSDV
BEGIN (
  rt_pkts ← 0)
  (if (($1=="s" || $1=="f") && $4 == "RTR" &&
    $7 == "DSDV")
    rt_pkts + 1)
)
END (
  print rt_pkts)
```

Gambar 4.22 *Pseudocode* RO

Contoh perintah untuk analisis RO dari *trace file* model propagasi *Nakagami* dengan protokol DSDV dan kecepatan maksimal perpindahan *node* sebesar 5 m/s dapat dilihat pada Gambar 4.22.

```
awk -f ro.awk dsdv_outnakaa3.tr
```

Gambar 4.23 Perintah untuk menjalankan ro.awk

Contoh *output* dari menjalankan skrip di atas seperti pada Gambar 4.23.

```
total no of routing packets      1474
```

Gambar 4.24 Hasil *running* ro.awk

BAB V PENGUJIAN DAN EVALUASI

Bab ini membahas mengenai pengujian dari skenario NS-2 yang telah dibuat. Pengujian fungsionalitas akan dibagi ke dalam beberapa skenario pengujian.

5.1. Lingkungan Pengujian

Uji coba dilakukan pada sebuah laptop yang telah terpasang sistem operasi yaitu Windows dan Linux. Spesifikasi laptop yang digunakan ditunjukkan pada Tabel 5.1.

Tabel 5.1 Spesifikasi laptop yang digunakan

Komponen	Spesifikasi
CPU	Intel(R) Core i7-2630QM CPU @2.90GHz
Sistem Operasi	Linux Ubuntu 14.04 LTS 64-bit (NS-2, DSDV, <i>Mobility Generation</i> , <i>Traffic-Connection Generation</i> , Nakagami, TwoRayGround) & Windows 10
Memori	4GB DDR3
Media Penyimpanan	1TB

5.2. Kriteria Pengujian

Skenario yang diuji menggunakan beberapa kriteria. Skenario tersebut dihasilkan oleh *mobility generator default* dari NS-2. Kriteria dari skenario ditunjukkan pada Tabel 5.2.

Tabel 5.2 Kriteria pengujian

Kriteria	Spesifikasi
Skenario	MANET (<i>Random Way Point</i>), <i>mobility generator</i>
Kecepatan Maksimal Perpindahan <i>Node</i> (m/s)	5, 10, 15
Jumlah Percobaan	10 kali

Kriteria	Spesifikasi
Jarak <i>Node</i> 1 dan <i>Node</i> 2	Acak
Posisi Awal <i>Node</i>	Acak
Pergerakan	Acak
Protokol <i>Routing</i>	DSDV
Pengiriman Paket Data	<i>TwoRayGround</i> 0 - 100 detik <i>Nakagami</i> 100 – 200 detik

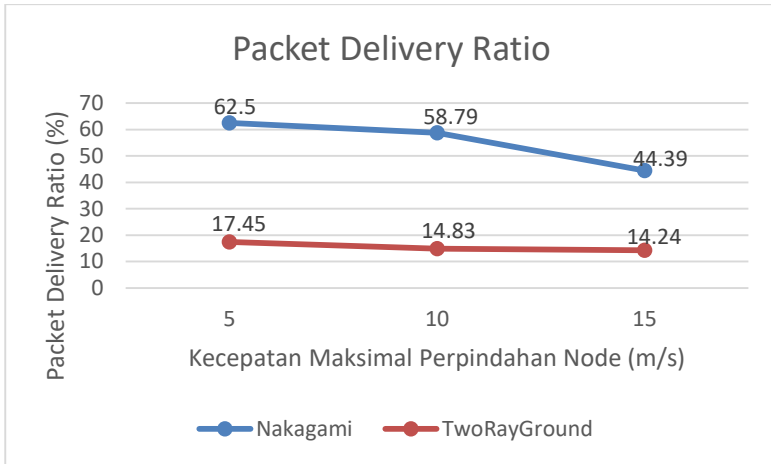
5.3. Analisis Packet Delivery Ratio (PDR)

Setelah *trace file* dijalankan dengan program skenario *node-movement (mobility generation)*, model propagasi *Nakagami*, dan model propagasi *TwoRayGround*, akan didapatkan nilai PDR dengan menjalankan *script* “pdr.awk”. Hasil PDR tersebut kemudian dirata-ratakan menjadi seperti pada Tabel 5.3.

Tabel 5.3 Hasil Packet Delivery Ratio (PDR)

Kecepatan Maksimal Perpindahan <i>Node</i> (m/s)	PDR (%)	
	<i>Nakagami</i>	<i>TwoRayGround</i>
5	62,5	17,45
10	58,79	14,83
15	44,39	14,24

Tabel 5.3 dan Gambar 5.1 menunjukkan performa PDR yang dihasilkan oleh model propagasi *Nakagami* dan *TwoRayGround* pada jaringan MANET dengan menggunakan skenario *node-movement (mobility generaion)* yang bersifat *Random Way Point*.



Gambar 5.1 Grafik hasil PDR

Dapat dilihat dari hasil uji PDR di atas, nilai yang dihasilkan oleh propagasi *Nakagami* memiliki nilai yang lebih tinggi daripada model propagasi *TwoRayGround* walaupun PDR yang dihasilkan oleh kedua model propagasi tersebut sama-sama menurun seiring dengan bertambahnya kecepatan maksimal *node* pada kedua model propagasi tersebut.

Pada saat kecepatan maksimal perpindahan *node* sebesar 10 m/s dan 15 m/s, model propagasi *Nakagami* mengalami penurunan hingga 14.4% sedangkan model propagasi *TwoRayGround* mengalami penurunan hanya sebesar 0.59%.

Nilai PDR yang dihasilkan model *TwoRayGround* lebih stabil daripada model *Nakagami*, namun semua PDR yang dimiliki model *TwoRayGround* bernilai di bawah 20%. Hal tersebut disebabkan model propagasi *TwoRayGround* tidak dapat mengidentifikasi lingkungan sekitar area simulasi. Selain itu, model propagasi *TwoRayGround* juga tidak memerhatikan pengaruh ketinggian *antenna* pada *node*. Kedua faktor tersebut menyebabkan pengiriman paket antar *node* lebih cepat dan dinamis. Pergerakan *node* yang lebih dinamis tersebut dapat

menyebabkan putusnya rute pada saat pengiriman paket data atau kegagalan pembentukan tabel *routing* pada DSDV sehingga paket data tidak dapat sampai ke tujuan. Berbeda dengan model *Nakagami* yang dapat mengidentifikasi lingkungan sekitar area simulasi dan pengaruh ketinggian *antenna* pada *node* yang ada, sehingga model *Nakagami* memiliki nilai PDR yang tinggi.

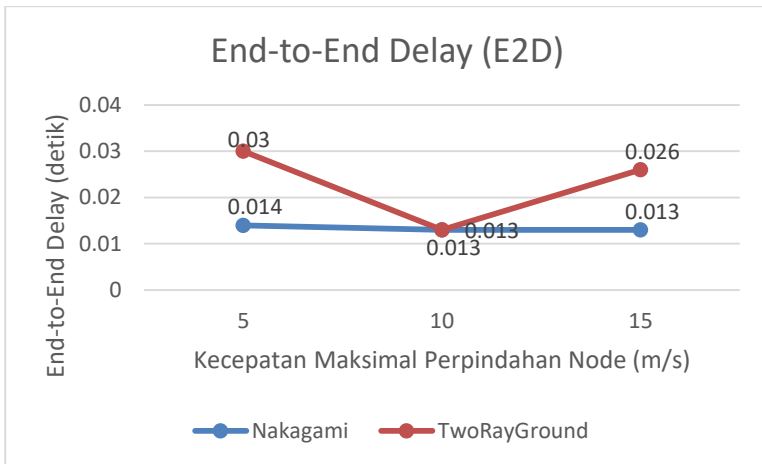
Hal lain yang dapat mempengaruhi penurunan atau pun peningkatan nilai PDR adalah penempatan dan pergerakan acak yang diimplementasikan pada skenario yang dihasilkan oleh *file node-movement (mobility generation)*.

5.4. Analisis *End-to-End Delay (E2D)*

Trace file hasil menjalankan program skenario *node-movement (mobility generation)* menggunakan model propagasi *Nakagami* dan *TwoRayGround* kemudian dianalisis nilai *End-to-End Delay* melalui *script "e2e.awk"*. Hasil tiap perhitungan E2D kemudian dirata-rata menjadi seperti pada Tabel 5.4.

Tabel 5.4 Hasil *End-to-End Delay (E2D)*

Kecepatan Maksimal Perpindahan <i>Node</i> (m/s)	E2D (detik)	
	<i>Nakagami</i>	<i>TwoRayGround</i>
5	0,014	0,03
10	0,013	0,013
15	0,013	0,026



Gambar 5.2 Grafik hasil E2D

Nilai *End-to-End Delay* pada model propagasi *TwoRayGround* lebih fluktuatif jika dibandingkan dengan model propagasi *Nakagami*. Pada saat kecepatan maksimal perpindahan *node* 10 m/s, nilai E2D *TwoRayGround* turun sebesar 0.017 detik ke 0.013 detik. Setelah itu, kembali naik ke 0.026 detik. Sedangkan pada model propagasi *Nakagami*, nilai E2D hampir tidak ada perubahan yaitu hanya mengalami penurunan dari kecepatan maksimal perpindahan model 5 m/s ke 10 m/s sebesar 0.001 detik. Hasil naik dan turun tersebut disebabkan karena *delay* dari masing-masing kecepatan maksimal berbeda pada skenario yang diuji coba. Uji coba menghasilkan nilai *delay* yang terdapat pada kecepatan maksimal 10 m/s lebih kecil dibandingkan dengan nilai *delay* pada kecepatan maksimal 5 m/s dan 15 m/s.

Berbeda dengan *TwoRayGround*, nilai E2D pada model propagasi *Nakagami* memiliki nilai yang lebih baik dan lebih. Hal tersebut dapat disebabkan oleh mobilitas pengiriman paket antar *node* yang dinamis pada *node-movement (mobility generation)* yang dihasilkan oleh model propagasi *TwoRayGround*. Mobilitas yang sangat dinamis tersebut dapat menyebabkan terputusnya rute saat pengiriman data sehingga terjadi antrena ketika menunggu rute yang baru terbentuk kembali.

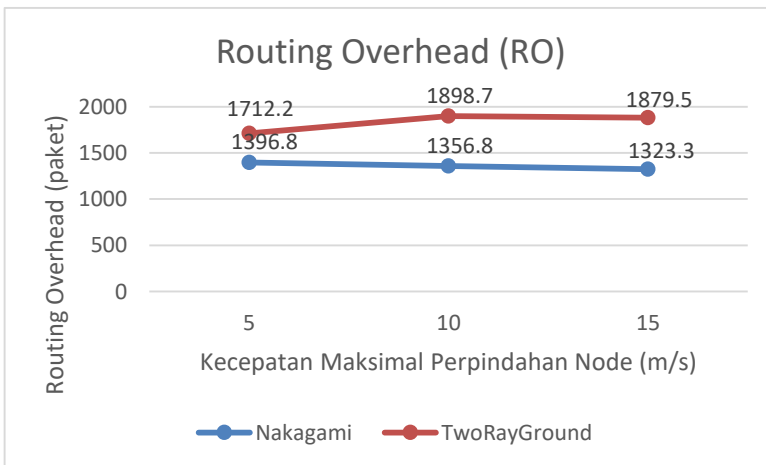
5.5. Analisis *Routing Overhead (RO)*

Nilai *Routing Overhead (RO)* didapat dari running program skenario *node-movement (mobility generation)*, model propagasi *Nakagami*, model propagasi *TwoRayGround*, dan skrip “ro.awk”. Hasil uji RO kemudian dirata-rata menjadi seperti tabel 5.5.

Trace file hasil menjalankan program skenario *node-movement (mobility generation)* menggunakan model propagasi *Nakagami* kemudian dianalisis nilai *Routing Overhead* melalui skrip “ro.awk”. Hasil tiap perhitungan RO skenario ditabulasikan dan dirata-ratakan menjadi Gambar 5.5.

Tabel 5.5 Hasil *Routing Overhead*

Kecepatan Maksimal Perpindahan <i>Node</i> (m/s)	<i>Routing Overhead</i> (paket)	
	<i>Nakagami</i>	<i>TwoRayGround</i>
5	1396,8	1712,2
10	1356,8	1898,7
15	1323,3	1879,5



Gambar 5.3 Grafik hasil RO

Hasil *Routing Overhead* untuk model propagasi *TwoRayGround* memiliki nilai yang fluktuatif. Pada saat kecepatan maksimal dari 5 m/s ke 10 m/s, nilai RO mengalami kenaikan sebesar 186.5 paket. Kemudian turun sebesar 19.2 paket pada kecepatan maksimal 15 m/s.

Berbeda dengan model propagasi *TwoRayGround* yang fluktuatif, nilai RO dari model propagasi *Nakagami* terus mengalami penurunan pada kecepatan maksimal 5 m/s, 10 m/s, dan 15 m/s.

Nilai RO yang dimiliki oleh model propagasi *TwoRayGround* memiliki nilai yang lebih besar dibandingkan dengan nilai RO model propagasi *Nakagami*. Hal tersebut dikarenakan pengiriman paket *routing* jenis *send* atau pun *forward* yang dihasilkan oleh model *TwoRayGround* lebih banyak dari *Nakagami*. Sehingga, semakin banyak *routing send* dan *forward* yang dihasilkan, pengiriman paket data yang dilakukan juga lebih besar untuk sampai ke *node* tujuan.

Hasil analisis pada metrik-metrik di atas berbeda antara model propagasi *Nakagami* dan *TwoRayGround*. Hal tersebut dikarenakan faktor lokasi dan pergerakan pada *node*.

[Halaman ini sengaja dikosongkan]

BAB VI PENUTUP

Pada bab ini akan diberikan kesimpulan yang diperoleh selama pengerjaan Tugas Akhir dan saran mengenai pengembangan yang dapat dilakukan terhadap Tugas Akhir ini di masa yang akan datang.

6.1. Kesimpulan

Kesimpulan yang dapat diambil dalam Tugas Akhir ini adalah sebagai berikut:

1. Performa pada skenario MANET yang dihasilkan oleh *node-movement (mobility generation)* dan dijalankan menggunakan model propagasi *Nakagami* dan protokol *routing DSDV* dengan penambahan kecepatan maksimal perpindahan *node* adalah sebagai berikut:
 - Nilai *Packet Delivery Ratio (PDR)* yang dihasilkan yang dihasilkan semakin menurun secara stabil dari 62.5% pada kecepatan maksimal 5 m/s ke 44.39% pada kecepatan maksimal 15 m/s.
 - Nilai *End-to-End Delay (E2D)* yang dihasilkan cenderung stabil. Nilai E2D mengalami penurunan hanya sebesar 0.001 detik dari 0.014 detik pada kecepatan maksimal 5 m/s ke 0.013 detik pada kecepatan maksimal 10 m/s yang kemudian mengalami nilai yang sama yaitu 0.013 detik pada kecepatan maksimal 15 m/s.
 - Nilai *Routing Overhead (RO)* mengalami penurunan namun cenderung stabil. Dari kecepatan maksimal 5 m/s ke 10 m/s, nilai RO turun dari 1396.8 paket ke 1356.8 paket. Kemudian mengalami penurunan kembali ke 1323.3 paket pada kecepatan maksimal 15 m/s.
2. Performa pada skenario MANET yang menghasilkan *node-movement (mobility generation)* dan dijalankan menggunakan model propagasi *TwoRayGround* dan protokol *routing DSDV*

dengan penambahan kecepatan maksimal perpindahan *node* adalah sebagai berikut:

- Nilai Packet Delivery Ratio yang dihasilkan memiliki nilai yang kecil dan mengalami penurunan pada tiap kecepatan. Nilai PDR pada kecepatan maksimal 5 m/s hanya sebesar 17.45% yang kemudian turun ke 14.83% pada kecepatan maksimal 10 m/s. Setelah itu turun lagi ke nilai 14.24% pada kecepatan maksimal 15 m/s.
- Nilai End-to-End Delay (E2D) bersifat fluktuatif. Pada kecepatan maksimal 5 m/s, E2D bernilai 0.03 detik. Kemudian turun ke nilai 0.013 detik pada kecepatan maksimal 10 m/s. E2D mengalami kenaikan kembali pada kecepatan maksimal 15 m/s dengan nilai 0.026 detik.
- Nilai Routing Overhead (RO) juga bersifat fluktuatif. Nilai RO mengalami peningkatan dari 1712.2 paket pada kecepatan maksimal 5 m/s ke 1898.7 paket pada kecepatan maksimal 10 m/s yang kemudian mengalami penurunan pada kecepatan maksimal 15 m/s dengan nilai 1879.5 paket.

6.2. Saran

Dalam pengerjaan Tugas Akhir ini terdapat beberapa saran untuk perbaikan serta pengembangan sistem yang telah dikerjakan sebagai berikut:

1. Penerapan model propagasi *Nakagami* dan *TwoRayGround* dapat diujicobakan pada lingkungan VANET.
2. Uji coba dapat dilakukan dengan melakukan pengurangan atau penambahan jumlah *node* dan penambahan jumlah percobaan untuk skenario.
3. Kecepatan maksimal dari skenario bisa ditambah untuk mendapatkan hasil yang lebih variatif.
4. Parameter-parameter yang digunakan dapat dimodifikasi, seperti pada parameter *transmission range* atau *pausetime*.

DAFTAR PUSTAKA

- [1] A. Suprpto, "Mengenal Mobile Ad-Hoc Network (MANET)," 7 September 2014. [Online]. Available: <http://agungsuprpto.net/mengenal-mobile-ad-hoc-network-manet/>. [Accessed 21 Juni 2017].
- [2] A. Line and M. Salman, Implementasi dan Analisis Kinerja Routing Protocol B.A.T.M.A.N-Adv (Better Approach To Mobile Ad-Hoc Networking Advanced) pada Jaringan Berbasis Wireless Mesh, Depok: Universitas Indonesia, 2013.
- [3] R. D. Pambudi and M. Yasa, "Implementasi Protokol Routing Better Approach To Ad-Hoc Mobile Network (B.A.T.M.A.N.) pada Wireless Mesh Networking," Universitas Muhammadiyah Malang (UMM), Malang.
- [4] B. D. Shivhare, C. Wahi and S. Shivhare, "Comparison of Proactive and Reactive Routing Protocols in Mobile Adhoc Network Using Routing Protocol Property," *International Journal of Emerging Technology and Advanced Engineering*, vol. 2, no. 3, 2012.
- [5] V. Garousi, Analysis of Network Traffic in Ad-Hoc Networks based on DSDV Protocol with Emphaiss on Mobility and Communication Patterns, Ottawa: Department of System and Computer Engineering, Carleton University, 2005.
- [6] M. F. Rochman, Analisis Kerja Protokol Destination-Sequenced Distance-Vector (DSDV) pada Jaringan Wireless Ad-Hoc, Bogor: Institut Pertanian Bogor, 2007.
- [7] A. Mukherjee, S. Bandyopadhyay and D. Saha, Location Management and Routing in Mobile Wireless Network, Norwood: Artech House, 2003.

- [8] M. K. Simon, J. K. Omura, R. A. Scholtz and B. K. Levitt, Spread Spectrum Communications, New York: McGraw-Hill, 1994.
- [9] A. Saputra, "Pengetahuan Tentang Jaringan," 25 Maret 2009. [Online]. Available: <http://de-monk.blogspot.co.id/>. [Accessed 21 Juni 2017].
- [10] R. Baumann, Engineering and simulation of mobile ad hoc routing protocols for VANET on Highways and in cities, Institute of Technology Zurich, 2014.
- [11] O. S. GNU, "<https://www.gnu.org/software/gawk/manual/gawk.html>," [Online]. [Accessed November 2017].

LAMPIRAN

1	#
2	# nodes: 50, pause: 10.00, max speed: 5.00, max
3	x: 510.00, max y: 510.00
4	#
5	\$node_(0) set X_ 135.750954250386
6	\$node_(0) set Y_ 249.694883657278
7	\$node_(0) set Z_ 0.000000000000
8	\$node_(1) set X_ 182.814657191152
9	\$node_(1) set Y_ 191.562026090664
10	\$node_(1) set Z_ 0.000000000000
11	\$node_(2) set X_ 74.711349536656
12	\$node_(2) set Y_ 204.677829249526
13	\$node_(2) set Z_ 0.000000000000
14	\$node_(3) set X_ 498.793769266713
15	\$node_(3) set Y_ 290.289597045685
16	\$node_(3) set Z_ 0.000000000000
17	\$node_(4) set X_ 492.805215151395
18	\$node_(4) set Y_ 28.871413975780
19	\$node_(4) set Z_ 0.000000000000
20	\$node_(5) set X_ 9.951753399792
21	\$node_(5) set Y_ 160.764896136033
22	\$node_(5) set Z_ 0.000000000000
23	\$node_(6) set X_ 454.892554487486
24	\$node_(6) set Y_ 415.304532885398
25	\$node_(6) set Z_ 0.000000000000
26	\$node_(7) set X_ 65.373899109608
27	\$node_(7) set Y_ 318.704139664877
28	\$node_(7) set Z_ 0.000000000000
29	\$node_(8) set X_ 20.033257849728
30	\$node_(8) set Y_ 445.693230692808
31	\$node_(8) set Z_ 0.000000000000
32	\$node_(9) set X_ 123.399070330664
33	\$node_(9) set Y_ 387.464792694123
34	\$node_(9) set Z_ 0.000000000000
35	\$node_(10) set X_ 418.695669632568
36	\$node_(10) set Y_ 96.836002439235
37	\$node_(10) set Z_ 0.000000000000
38	\$node_(11) set X_ 9.932581646363
39	\$node_(11) set Y_ 194.284431464309
40	\$node_(11) set Z_ 0.000000000000
41	\$node_(12) set X_ 173.644322042358
42	\$node_(12) set Y_ 284.975200769222

43	\$node_(12)	set Z_	0.000000000000
44	\$node_(13)	set X_	199.604589181895
45	\$node_(13)	set Y_	460.436798185309
46	\$node_(13)	set Z_	0.000000000000
47	\$node_(14)	set X_	231.498877660317
48	\$node_(14)	set Y_	249.078008578677
49	\$node_(14)	set Z_	0.000000000000
50	\$node_(15)	set X_	73.802255755493
51	\$node_(15)	set Y_	289.599801382692
52	\$node_(15)	set Z_	0.000000000000
53	\$node_(16)	set X_	427.986335661532
54	\$node_(16)	set Y_	285.494592074043
55	\$node_(16)	set Z_	0.000000000000
56	\$node_(17)	set X_	165.398612342056
57	\$node_(17)	set Y_	26.498884938603
58	\$node_(17)	set Z_	0.000000000000
59	\$node_(18)	set X_	354.711998191219
60	\$node_(18)	set Y_	401.430845513850
61	\$node_(18)	set Z_	0.000000000000
62	\$node_(19)	set X_	401.636635949933
63	\$node_(19)	set Y_	166.016710717575
64	\$node_(19)	set Z_	0.000000000000
65	\$node_(20)	set X_	424.945723947303
66	\$node_(20)	set Y_	493.107151832964
67	\$node_(20)	set Z_	0.000000000000
68	\$node_(21)	set X_	150.094300955444
69	\$node_(21)	set Y_	60.295996691931
70	\$node_(21)	set Z_	0.000000000000
71	\$node_(22)	set X_	464.882394195427
72	\$node_(22)	set Y_	131.697105719475
73	\$node_(22)	set Z_	0.000000000000
74	\$node_(23)	set X_	310.138259604284
75	\$node_(23)	set Y_	330.972045711751
76	\$node_(23)	set Z_	0.000000000000
77	\$node_(24)	set X_	224.186319557660
78	\$node_(24)	set Y_	55.935298105362
79	\$node_(24)	set Z_	0.000000000000
80	\$node_(25)	set X_	68.265767679382
81	\$node_(25)	set Y_	133.963275371204
82	\$node_(25)	set Z_	0.000000000000
83	\$node_(26)	set X_	477.145613906041
84	\$node_(26)	set Y_	110.473191202251
85	\$node_(26)	set Z_	0.000000000000
86	\$node_(27)	set X_	260.771651412478

87	\$node_(27)	set Y_	482.963389383253
88	\$node_(27)	set Z_	0.000000000000
89	\$node_(28)	set X_	68.952128445274
90	\$node_(28)	set Y_	295.548883214166
91	\$node_(28)	set Z_	0.000000000000
92	\$node_(29)	set X_	490.361537778564
93	\$node_(29)	set Y_	367.359902390945
94	\$node_(29)	set Z_	0.000000000000
95	\$node_(30)	set X_	330.822193026779
96	\$node_(30)	set Y_	423.774416317483
97	\$node_(30)	set Z_	0.000000000000
98	\$node_(31)	set X_	26.963546946276
99	\$node_(31)	set Y_	210.659694654685
100	\$node_(31)	set Z_	0.000000000000
101	\$node_(32)	set X_	345.748218424998
102	\$node_(32)	set Y_	167.614025264913
103	\$node_(32)	set Z_	0.000000000000
104	\$node_(33)	set X_	494.738326483772
105	\$node_(33)	set Y_	31.094867148840
106	\$node_(33)	set Z_	0.000000000000
107	\$node_(34)	set X_	84.646449854239
108	\$node_(34)	set Y_	54.907826776809
109	\$node_(34)	set Z_	0.000000000000
110	\$node_(35)	set X_	480.506987917575
111	\$node_(35)	set Y_	299.198264447795
112	\$node_(35)	set Z_	0.000000000000
113	\$node_(36)	set X_	149.680806173386
114	\$node_(36)	set Y_	20.026842962388
115	\$node_(36)	set Z_	0.000000000000
116	\$node_(37)	set X_	147.672159046822
117	\$node_(37)	set Y_	321.895680593863
118	\$node_(37)	set Z_	0.000000000000
119	\$node_(38)	set X_	217.011749394807
120	\$node_(38)	set Y_	487.116652934408
121	\$node_(38)	set Z_	0.000000000000
122	\$node_(39)	set X_	335.821853364107
123	\$node_(39)	set Y_	106.901493003944
124	\$node_(39)	set Z_	0.000000000000
125	\$node_(40)	set X_	386.223832304719
126	\$node_(40)	set Y_	479.547043718813
127	\$node_(40)	set Z_	0.000000000000
128	\$node_(41)	set X_	31.685296001505
129	\$node_(41)	set Y_	336.397299145031
130	\$node_(41)	set Z_	0.000000000000

131	\$node_ (42)	set X_	85.695081535861
132	\$node_ (42)	set Y_	166.594112087408
133	\$node_ (42)	set Z_	0.000000000000
134	\$node_ (43)	set X_	61.475711014808
135	\$node_ (43)	set Y_	433.571081269250
136	\$node_ (43)	set Z_	0.000000000000
137	\$node_ (44)	set X_	437.554667648247
138	\$node_ (44)	set Y_	329.046132143493
139	\$node_ (44)	set Z_	0.000000000000
140	\$node_ (45)	set X_	436.493102647496
141	\$node_ (45)	set Y_	3.963881704138
142	\$node_ (45)	set Z_	0.000000000000
143	\$node_ (46)	set X_	352.417926754087
144	\$node_ (46)	set Y_	437.233882670907
145	\$node_ (46)	set Z_	0.000000000000
146	\$node_ (47)	set X_	57.657417911744
147	\$node_ (47)	set Y_	327.082450500026
148	\$node_ (47)	set Z_	0.000000000000
149	\$node_ (48)	set X_	353.991479631566
150	\$node_ (48)	set Y_	156.788900325096
151	\$node_ (48)	set Z_	0.000000000000
152	\$node_ (49)	set X_	503.331492525281
153	\$node_ (49)	set Y_	265.798646653099
154	\$node_ (49)	set Z_	0.000000000000

Gambar 7.1 Posisi node dari potongan file “scena1-5.txt”

1	\$god_	set-dist	0	1	1
2	\$god_	set-dist	0	2	1
3	\$god_	set-dist	0	3	2
4	\$god_	set-dist	0	4	2
5	\$god_	set-dist	0	5	1
6	\$god_	set-dist	0	6	2
7	\$god_	set-dist	0	7	1
8	\$god_	set-dist	0	8	1
9	\$god_	set-dist	0	9	1
11	\$god_	set-dist	0	10	2
12	\$god_	set-dist	0	11	1
13	\$god_	set-dist	0	12	1
14	\$god_	set-dist	0	13	1
15	\$god_	set-dist	0	14	1
16	\$god_	set-dist	0	15	1
17	\$god_	set-dist	0	16	2

18	\$god_	set-dist	0	17	1
19	\$god_	set-dist	0	18	2
20	\$god_	set-dist	0	19	2
21	\$god_	set-dist	0	20	2
22	\$god_	set-dist	0	21	1
23	\$god_	set-dist	0	22	2
24	\$god_	set-dist	0	23	1
25	\$god_	set-dist	0	24	1
26	\$god_	set-dist	0	25	1
27	\$god_	set-dist	0	26	2
28	\$god_	set-dist	0	27	2
29	\$god_	set-dist	0	28	1
30	\$god_	set-dist	0	29	2
31	\$god_	set-dist	0	30	2
32	\$god_	set-dist	0	31	1
33	\$god_	set-dist	0	32	1
34	\$god_	set-dist	0	33	2
35	\$god_	set-dist	0	34	1
36	\$god_	set-dist	0	35	2
37	\$god_	set-dist	0	36	1
38	\$god_	set-dist	0	37	1
39	\$god_	set-dist	0	38	2
40	\$god_	set-dist	0	39	1
41	\$god_	set-dist	0	40	2
42	\$god_	set-dist	0	41	1
43	\$god_	set-dist	0	42	1
44	\$god_	set-dist	0	43	1
45	\$god_	set-dist	0	44	2
46	\$god_	set-dist	0	45	2
47	\$god_	set-dist	0	46	2
48	\$god_	set-dist	0	47	1
49	\$god_	set-dist	0	48	1
50	\$god_	set-dist	0	49	2
51	\$god_	set-dist	1	2	1
52	\$god_	set-dist	1	3	2
53	\$god_	set-dist	1	4	2
54	\$god_	set-dist	1	5	1
55	\$god_	set-dist	1	6	2
56	\$god_	set-dist	1	7	1
57	\$god_	set-dist	1	8	2
58	\$god_	set-dist	1	9	1
59	\$god_	set-dist	1	10	2
60	\$god_	set-dist	1	11	1
61	\$god_	set-dist	1	12	1

62	\$god_	set-dist	1	13	2
63	\$god_	set-dist	1	14	1
64	\$god_	set-dist	1	15	1
65	\$god_	set-dist	1	16	2
66	\$god_	set-dist	1	17	1
67	\$god_	set-dist	1	18	2
68	\$god_	set-dist	1	19	1
69	\$god_	set-dist	1	20	2
70	\$god_	set-dist	1	21	1
71	\$god_	set-dist	1	22	2
72	\$god_	set-dist	1	23	1
73	\$god_	set-dist	1	24	1
74	\$god_	set-dist	1	25	1
75	\$god_	set-dist	1	26	2
76	\$god_	set-dist	1	27	2
77	\$god_	set-dist	1	28	1
78	\$god_	set-dist	1	29	2
79	\$god_	set-dist	1	30	2
80	\$god_	set-dist	1	31	1
81	\$god_	set-dist	1	32	1
82	\$god_	set-dist	1	33	2
83	\$god_	set-dist	1	34	1
84	\$god_	set-dist	1	35	2
85	\$god_	set-dist	1	36	1
86	\$god_	set-dist	1	37	1
87	\$god_	set-dist	1	38	2
88	\$god_	set-dist	1	39	1
89	\$god_	set-dist	1	40	2
90	\$god_	set-dist	1	41	1
91	\$god_	set-dist	1	42	1
92	\$god_	set-dist	1	43	2
93	\$god_	set-dist	1	44	2
94	\$god_	set-dist	1	45	2
95	\$god_	set-dist	1	46	2
96	\$god_	set-dist	1	47	1
97	\$god_	set-dist	1	48	1
98	\$god_	set-dist	1	49	2
99	\$god_	set-dist	2	3	3
100	\$god_	set-dist	2	4	3
101	\$god_	set-dist	2	5	1
102	\$god_	set-dist	2	6	3
103	\$god_	set-dist	2	7	1
104	\$god_	set-dist	2	8	1
105	\$god_	set-dist	2	9	1

106	\$god_	set-dist	2	10	2
107	\$god_	set-dist	2	11	1
108	\$god_	set-dist	2	12	1
109	\$god_	set-dist	2	13	2
110	\$god_	set-dist	2	14	1
111	\$god_	set-dist	2	15	1
112	\$god_	set-dist	2	16	2
113	\$god_	set-dist	2	17	1
114	\$god_	set-dist	2	18	2
115	\$god_	set-dist	2	19	2
116	\$god_	set-dist	2	20	3
117	\$god_	set-dist	2	21	1
118	\$god_	set-dist	2	22	3
119	\$god_	set-dist	2	23	2
120	\$god_	set-dist	2	24	1
121	\$god_	set-dist	2	25	1
122	\$god_	set-dist	2	26	3
123	\$god_	set-dist	2	27	2
124	\$god_	set-dist	2	28	1
125	\$god_	set-dist	2	29	3
126	\$god_	set-dist	2	30	2
127	\$god_	set-dist	2	31	1
128	\$god_	set-dist	2	32	2
129	\$god_	set-dist	2	33	3
130	\$god_	set-dist	2	34	1
131	\$god_	set-dist	2	35	3
132	\$god_	set-dist	2	36	1
133	\$god_	set-dist	2	37	1
134	\$god_	set-dist	2	38	2
135	\$god_	set-dist	2	39	2
136	\$god_	set-dist	2	40	3
137	\$god_	set-dist	2	41	1
138	\$god_	set-dist	2	42	1
139	\$god_	set-dist	2	43	1
140	\$god_	set-dist	2	44	2
141	\$god_	set-dist	2	45	2
142	\$god_	set-dist	2	46	2
143	\$god_	set-dist	2	47	1
144	\$god_	set-dist	2	48	2
145	\$god_	set-dist	2	49	3
146	\$god_	set-dist	3	4	2
147	\$god_	set-dist	3	5	3
148	\$god_	set-dist	3	6	1
149	\$god_	set-dist	3	7	2

150	\$god_	set-dist	3	8	3
151	\$god_	set-dist	3	9	2
152	\$god_	set-dist	3	10	1
153	\$god_	set-dist	3	11	3
154	\$god_	set-dist	3	12	2
155	\$god_	set-dist	3	13	2
156	\$god_	set-dist	3	14	2
157	\$god_	set-dist	3	15	2
158	\$god_	set-dist	3	16	1
159	\$god_	set-dist	3	17	2
160	\$god_	set-dist	3	18	1
161	\$god_	set-dist	3	19	1
162	\$god_	set-dist	3	20	1
163	\$god_	set-dist	3	21	2
164	\$god_	set-dist	3	22	1
165	\$god_	set-dist	3	23	1
166	\$god_	set-dist	3	24	2
167	\$god_	set-dist	3	25	3
168	\$god_	set-dist	3	26	1
169	\$god_	set-dist	3	27	2
170	\$god_	set-dist	3	28	2
171	\$god_	set-dist	3	29	1
172	\$god_	set-dist	3	30	1
173	\$god_	set-dist	3	31	3
174	\$god_	set-dist	3	32	1
175	\$god_	set-dist	3	33	2
176	\$god_	set-dist	3	34	3
177	\$god_	set-dist	3	35	1
178	\$god_	set-dist	3	36	2
179	\$god_	set-dist	3	37	2
180	\$god_	set-dist	3	38	2
181	\$god_	set-dist	3	39	1
182	\$god_	set-dist	3	40	1
183	\$god_	set-dist	3	41	3
184	\$god_	set-dist	3	42	3
185	\$god_	set-dist	3	43	3
186	\$god_	set-dist	3	44	1
187	\$god_	set-dist	3	45	2
188	\$god_	set-dist	3	46	1
189	\$god_	set-dist	3	47	3
190	\$god_	set-dist	3	48	1
191	\$god_	set-dist	3	49	1

Gambar 7.2 Pembuatan GOD node dari potongan “scena1-5.txt”

1	\$ns_ at 10.000000000000 "\$node_(0) setdest 59.642927959964 493.035963718193 2.153551659533"
2	\$ns_ at 10.000000000000 "\$node_(1) setdest 503.468798876626 222.357522370379 2.402489064196"
3	\$ns_ at 10.000000000000 "\$node_(2) setdest 24.523026847930 213.164639586640 1.639526471501"
4	\$ns_ at 10.000000000000 "\$node_(3) setdest 439.600772572905 374.604226364214 3.912637298654"
5	\$ns_ at 10.000000000000 "\$node_(4) setdest 375.889599010124 490.336170338542 0.207585460315"
6	\$ns_ at 10.000000000000 "\$node_(5) setdest 234.751915917359 192.108064416907 4.409341618219"
7	\$ns_ at 10.000000000000 "\$node_(6) setdest 106.533621069741 117.854338359018 1.091111771984"
8	\$ns_ at 10.000000000000 "\$node_(7) setdest 116.550187238594 480.254163484291 3.971165658674"
9	\$ns_ at 10.000000000000 "\$node_(8) setdest 295.669024935727 228.224030709965 1.546599362440"
10	\$ns_ at 10.000000000000 "\$node_(9) setdest 446.675789854155 197.584119226666 0.750016360777"
11	\$ns_ at 10.000000000000 "\$node_(10) setdest 331.137126436579 28.485312162653 4.346742651919"
12	\$ns_ at 10.000000000000 "\$node_(11) setdest 207.381711209984 3.202633484301 2.825679392028"
13	\$ns_ at 10.000000000000 "\$node_(12) setdest 226.699308658358 213.687282329648 2.796211077741"
14	\$ns_ at 10.000000000000 "\$node_(13) setdest 63.770308961213 17.215559878581 2.940407741490"

15	\$ns_ at 10.000000000000 "\$node_(14) setdest 186.976459953260 38.665512521385 0.197337563393"
16	\$ns_ at 10.000000000000 "\$node_(15) setdest 271.238166065314 349.066826204280 3.310391370571"
17	\$ns_ at 10.000000000000 "\$node_(16) setdest 22.974568656345 365.907796702539 0.737246728583"
18	\$ns_ at 10.000000000000 "\$node_(17) setdest 366.744663599620 233.202130765664 3.738325109606"
19	\$ns_ at 10.000000000000 "\$node_(18) setdest 210.674932194032 410.965202572002 0.814595197895"
20	\$ns_ at 10.000000000000 "\$node_(19) setdest 383.509985972214 125.607315175310 4.879879220392"
21	\$ns_ at 10.000000000000 "\$node_(20) setdest 29.694999991115 238.919873993223 2.103747490928"
22	\$ns_ at 10.000000000000 "\$node_(21) setdest 87.138014634770 208.007986509628 4.740416538205"
23	\$ns_ at 10.000000000000 "\$node_(22) setdest 87.359217638317 325.007645120283 3.656203580436"
24	\$ns_ at 10.000000000000 "\$node_(23) setdest 466.628998464167 247.189038648577 3.750482973433"
25	\$ns_ at 10.000000000000 "\$node_(24) setdest 446.646892878356 176.659380957752 3.606706569017"
26	\$ns_ at 10.000000000000 "\$node_(25) setdest 109.561818137033 277.435926277347 4.604389548235"
27	\$ns_ at 10.000000000000 "\$node_(26) setdest 38.567043531673 252.765443228467 4.370094103268"
28	\$ns_ at 10.000000000000 "\$node_(27) setdest 387.589557699074 174.900611166220 1.493883124955"

29	\$ns_ at 10.000000000000 "\$node_(28) setdest 16.714023104058 88.793390900601 0.163259571176"
30	\$ns_ at 10.000000000000 "\$node_(29) setdest 468.112884936268 365.982847191494 1.540320819567"
31	\$ns_ at 10.000000000000 "\$node_(30) setdest 273.948984791382 322.680195441349 4.988797434529"
32	\$ns_ at 10.000000000000 "\$node_(31) setdest 329.261765239399 337.973914064601 3.316220249928"
33	\$ns_ at 10.000000000000 "\$node_(32) setdest 121.876186786743 461.567506060480 2.657408719589"
34	\$ns_ at 10.000000000000 "\$node_(33) setdest 414.650205708398 342.611564603449 1.479766361367"
35	\$ns_ at 10.000000000000 "\$node_(34) setdest 20.320039400032 270.787191517590 4.342951670600"
36	\$ns_ at 10.000000000000 "\$node_(35) setdest 330.335833944812 55.672723632289 0.707231691830"
37	\$ns_ at 10.000000000000 "\$node_(36) setdest 107.446526521556 41.773211538984 1.064084475238"
38	\$ns_ at 10.000000000000 "\$node_(37) setdest 506.740537118640 295.589291672821 2.664967252480"
39	\$ns_ at 10.000000000000 "\$node_(38) setdest 101.789542036649 485.773499992417 2.853684081595"
40	\$ns_ at 10.000000000000 "\$node_(39) setdest 207.755535406329 481.159804438064 0.311399837903"
41	\$ns_ at 10.000000000000 "\$node_(40) setdest 151.688437923136 32.469249429089 0.915451623829"
42	\$ns_ at 10.000000000000 "\$node_(41) setdest 369.804927860253 31.472811274311 4.142431221350"

43	\$ns_ at 10.000000000000 "\$node_(42) setdest 317.150012533460 501.385665768343 0.590782436282"
44	\$ns_ at 10.000000000000 "\$node_(43) setdest 175.274521689187 463.268969580519 2.487844231648"
45	\$ns_ at 10.000000000000 "\$node_(44) setdest 120.940589901442 335.866672792534 2.450564021644"
46	\$ns_ at 10.000000000000 "\$node_(45) setdest 467.399629449733 470.217056210420 1.327207263340"
47	\$ns_ at 10.000000000000 "\$node_(46) setdest 311.959269821534 221.057355930081 4.647775763119"
48	\$ns_ at 10.000000000000 "\$node_(47) setdest 230.119765464894 398.801585971548 0.101797926746"
49	\$ns_ at 10.000000000000 "\$node_(48) setdest 179.774797717845 61.616003649805 1.992615799993"
50	\$ns_ at 10.000000000000 "\$node_(49) setdest 157.222736722866 463.011210399291 0.397432466891"

Gambar 7.3 Pergerakan node dari potongan "scena1-5.txt"

1	\$ns_ at 10.223927349041 "\$god_ set-dist 4 37 2"
2	\$ns_ at 10.223927349041 "\$god_ set-dist 22 37 2"
3	\$ns_ at 10.223927349041 "\$god_ set-dist 26 37 2"
4	\$ns_ at 10.223927349041 "\$god_ set-dist 32 37 1"
5	\$ns_ at 10.223927349041 "\$god_ set-dist 33 37 2"
6	\$ns_ at 10.223927349041 "\$god_ set-dist 37 45 2"
7	\$ns_ at 10.341944268871 "\$god_ set-dist 7 22 2"

8	\$ns_ at 10.341944268871 "\$god_ set-dist 9 22 2"
9	\$ns_ at 10.341944268871 "\$god_ set-dist 13 22 2"
10	\$ns_ at 10.341944268871 "\$god_ set-dist 15 22 2"
11	\$ns_ at 10.341944268871 "\$god_ set-dist 22 23 1"
12	\$ns_ at 10.341944268871 "\$god_ set-dist 22 28 2"
13	\$ns_ at 10.341944268871 "\$god_ set-dist 22 38 2"
14	\$ns_ at 10.350504627520 "\$god_ set-dist 0 38 1"
15	\$ns_ at 10.380685441746 "\$god_ set-dist 2 22 2"
16	\$ns_ at 10.380685441746 "\$god_ set-dist 5 22 2"
17	\$ns_ at 10.380685441746 "\$god_ set-dist 22 24 1"
18	\$ns_ at 10.380685441746 "\$god_ set-dist 22 25 2"
19	\$ns_ at 10.380685441746 "\$god_ set-dist 22 34 2"
20	\$ns_ at 10.380685441746 "\$god_ set-dist 22 42 2"
21	\$ns_ at 10.710936666675 "\$god_ set-dist 6 38 2"
22	\$ns_ at 10.710936666675 "\$god_ set-dist 6 41 3"
23	\$ns_ at 10.710936666675 "\$god_ set-dist 6 47 3"
24	\$ns_ at 10.850734947661 "\$god_ set-dist 10 43 2"
25	\$ns_ at 10.850734947661 "\$god_ set-dist 14 43 1"
26	\$ns_ at 10.850734947661 "\$god_ set-dist 16 43 2"
27	\$ns_ at 10.850734947661 "\$god_ set-dist 19 43 2"
28	\$ns_ at 10.981405979017 "\$god_ set-dist 1 10 1"

Gambar 7.4 Informasi GOD dari potongan "scena1-5.txt"

```

1 #
2 # nodes: 10, max conn: 1, send rate: 1, seed:
3 1.0
4 #
5 #
6 # 1 connecting to 2 at time 2.5568388786897245
7 #
8 set udp_(0) [new Agent/UDP]
9 $ns_ attach-agent $node_(1) $udp_(0)
10 set null_(0) [new Agent/Null]
11 $ns_ attach-agent $node_(2) $null_(0)
12 set cbr_(0) [new Application/Traffic/CBR]
13 $cbr_(0) set packetSize_ 512
14 $cbr_(0) set interval_ 1
15 $cbr_(0) set random_ 1
16 $cbr_(0) set maxpkts_ 10000
17 $cbr_(0) attach-agent $udp_(0)
18 $ns_ connect $udp_(0) $null_(0)
19 $ns_ at 2.5568388786897245 "$cbr_(0) start"
20 #

```

Gambar 7.5 Koneksi yang digunakan pada cbrtest.txt

```

1 # =====
2 # Define options
3 #=====
4
5 set val(chan) Channel/WirelessChannel
6 set val(prop) Propagation/Nakagami
7 set val(netif) Phy/WirelessPhy
8 set val(mac) Mac/802_11
9 set val(ifq) Queue/DropTail/PriQueue
10 set val(ll) LL
11 set val(ant) Antenna/OmniAntenna
12 set opt(x) 510 ;
13 set opt(y) 510 ;
14 set val(ifqlen) 50 ;# max packet in ifq
15 set val(nn) 50 ;# how many nodes are
16 simulated
17 set val(seed) 0.0
18 set val(adhocRouting) DSDV
19 set val(stop) 100 ;# simulation time
20 set val(cp) "cbrtest.txt";#<-- traffic file

```



```

17 set val(sc) "scenal-5.txt" ;#<-- mobility
file

18 Phy/WirelessPhy set RXThresh_ 1.42681e-08
;#100m
19 #=====
20 # Main Program
21 #=====
22 #Initialize Global Variables
23 #create simulator instance
24
25 set ns_ [new Simulator]
26
27 set tracefd [open Naka-M5_1.tr w]
28 set namtrace [open Naka-M5_1.nam w]
29
30 # $ns_ use-newtrace
31 $ns_ trace-all $tracefd
32
33 $ns_ namtrace-all-wireless $namtrace $opt(x)
$opt(y)
34
35 # Setting up Topography Object
36
37 set topo [new Topography]
38 $topo load_flatgrid $opt(x) $opt(y)
39
40 # Create God
41 set god_ [create-god $val(nn)]
42
43 Phy/WirelessPhy set RXThresh_ 1.42681e-08
44
45 # Create nn mobilenodes [$val(nn)] and attach
them to channel
46
47 set chan_1_ [new $val(chan)]
48
49 # Configure the nodes
50
51 $ns_ node-config -adhocRouting
$val(adhocRouting) \
52 -llType $val(ll) \
53 -macType $val(mac) \
54 -channel $chan_1_ \

```

```

55         -ifqType $val(ifq) \
56         -ifqLen $val(ifqlen) \
57         -antType $val(ant) \
58         -propType $val(prop) \
59         -phyType $val(netif) \
60         -topoInstance $topo \
61         -agentTrace ON \
62         -routerTrace ON \
63         -macTrace OFF \
64         -movementTrace ON \
65
66     for {set i 0} {$i < $val(nn)} {incr i} {
67     set node_($i) [$ns_ node]
68     $node_($i) random-motion 0; #disable random
69     motion
70     }
71
72
73     # Provide Initial Location of Mobile Nodes
74     traffic
75     puts "Loading connection pattern..."
76     source $val(cp)
77
78     puts "Loading scenario file..."
79     source $val(sc)
80
81     # Set a TCP connection between node_(0) and
82     node_(1) sumo
83
84     #Setup a TCP connection
85
86     # Define node initial position in nam
87
88     for {set i 0} {$i < $val(nn)} {incr i} {
89     # 20 defines the node size for nam
90     $ns_ initial_node_pos $node_($i) 20
91     }
92
93     # Telling nodes when the simulation ends
94
95     for {set i 0} {$i < $val(nn)} { incr i} {
96     $ns_ at $val(stop).0 "$node_($i) reset"

```

```

96 }
97
98 # Ending nam and the simulation
99 $ns_ at $val(stop).0002 "puts \"NS EXITING...\"";
   $ns_ halt"
100
101 puts $tracefd "M 0.0 nn $val(nn) x $opt(x) y
102 $opt(y) rp $val(adhocRouting)"
   puts $tracefd "M 0.0 sc $val(sc) cp $val(cp)
103 seed $val(seed)"
   puts $tracefd "M 0.0 prop $val(prop) ant
104 $val(ant)"
105
106 puts "Starting Simulation..."
107 $ns_ run

```

Gambar 7.6 File tcl untuk protokol *routing* DSDV

```

1 BEGIN {
2   sent=0;
3   recv=0;
4   pdr=0;
5   }
6   {
7     #count packet send
8     if ($1 == "s" && $3 == "_1_" && $4 == "AGT" &&
9         $7 == "cbr")
10      {
11        sent++;
12      }
13     #count packet receive
14     if ($1 == "r" && $3 == "_2_" && $4 == "AGT" &&
15         $7 == "cbr")
16      {
17        recv++;
18      }
19   }
20 END {
21   pdr = ( recv / sent ) * 100
22   print "Transmitted packet (s):", sent;
23   print "Received packet (s):", recv;
24   print "Packet delivery ratio:", pdr, "%";
25 }

```

Gambar 7.7 Implementasi Packet Delivery Ratio pada file "pdr.awk"

```

1 BEGIN {
2   rt_pkts = 0;
3 }
4 {
5   if (($1 == "s" || $1 == "f") && ($4 == "RTR"))
6     rt_pkts++;
7 }
8 END {
9   printf ("Total number of routing packets\t%d\n",
10  rt_pkts);
11 }

```

Gambar 7.8 Implementasi Routing Overhead pada file "ro.awk"

```

1 BEGIN{
2   for ( i in pkt_id)
3     {
4         pkt_id[i] = 0;
5     }
6   for ( i in pkt_sent)
7     {
8         pkt_sent[i] = 0;
9     }
10  for ( i in pkt_recv )
11    {
12        pkt_recv[i] = 0;
13    }
14    delay = avg_delay = 0;
15    recv = 0;
16    recv_id = 0;
17  }
18  {
19    # count packet send
20    if ( $1 == "s" && $3 == "_1_" && $4 == "AGT" &&
21        $7 == "cbr" )
22      {
23        pkt_sent[$6] = $2;
24      }
25    # count packet receive
26

```

```
27 if ( $1 == "r" && $3 == "_2_" && $4 == "AGT" &&
28 $7 == "cbr" && recv_id != $6 )
29 {
30     recv++;
31     recv_id = $6;
32     pkt_recv[$6] = $2;
33 }
34 }
35 END{
36 for (i in pkt_recv)
37     {
38         delay += pkt_recv[i] - pkt_sent[i];
39     }
40
41     avg_delay = delay / recv;
42
43     print "Total Packet(s) Receive =", recv;
44     print "Total Delay =", delay, "second";
45     print "Average Packet Delivery Delay = ",
46     avg_delay, "second";
}
```

Gambar 7.9 Implementasi *End-to-End Delay* pada file “e2e.awk”

Halaman ini sengaja dikosongkan]

BIODATA PENULIS



Penulis bernama Andre Abdirrosyid. Penulis lahir di Bojonegoro, 30 Agustus 1994 sebagai anak sulung dari tiga bersaudara. Penulis menempuh pendidikan formal di TK Banjarejo (1999-2000), SDN Mlinjeng (2000-2006), SMP Plus Ar-Rahmat Bojonegoro (2006-2009) dan SMAN 10 Malang Sampoerna Academy (2009-2012). Pada tahun 2012, penulis diterima sebagai mahasiswa S1 Departemen Informatika

Fakultas Teknologi Informasi dan Komunikasi, Institut Teknologi Sepuluh Nopember Surabaya (ITS).

Pada tahun kedua dan ketiga perkuliahan, penulis mengikuti beberapa organisasi di antaranya Keluarga Muslim Informatika (KMI), Sampoerna Academy Alumni (SAA), dan Putih Abu-abu Scholarship (PAS). Di Departemen Informatika ITS penulis mengambil bidang minat Arsitektur dan Jaringan Komputer (AJK) dan mengambil topik Tugas Akhir tentang MANET. Penulis dapat dihubungi di LINE/facebook abdirrosyid atau melalui email andre.abdirrosyid@gmail.com.