ANALISIS KINERJA *ROUTING PROTOCOL OPTIMIZED LINK STATE ROUTING* (OLSR) DAN *AD HOC ON DEMAND DISTANCE VEKTOR* (AODV) PADA *VEHICULAR AD HOC NETWORK* (VANET)

SKRIPSI

Oleh

Ahmad Junaidi

NIM 121910201026

PROGRAM STUDI STRATA 1 TEKNIK ELEKTRO
JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNIK
UNIVERSITAS JEMBER
2019

ANALISIS KINERJA *ROUTING PROTOCOL OPTIMIZED LINK STATE ROUTING* (OLSR) DAN *AD HOC ON DEMAND DISTANCE VEKTOR* (AODV) PADA *VEHICULAR AD HOC NETWORK* (VANET)

SKRIPSI

Diajukan guna memenuhi tugas akhir dan memenuhi salah satu syarat
untuk menyelesaikan Program Studi Teknik Elektro (S1)
dam mencapai gelar Sarjana Teknik

Oleh:

Ahmad Junaidi

NIM 121910201026

PROGRAM STUDI STRATA 1 TEKNIK ELEKTRO
JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNIK
UNIVERSITAS JEMBER
2019

Persembahan

Skripsi ini saya persembahkan untuk:

1. Kedua orang tua saya, Sutikno, almh. Ernawati, adik saya Siti Junaida yang telah menjadi semangat untuk menyelesaikan tugas akhir ini, serta teman teman seperjuangan yang sudah membantu menemani hingga penelitian ini dapat diselesaikan.
2. Guru-guru SDN Pancoran 1, SMP Negeri 3 Bondowoso, SMK Negeri 1 Bondowoso dan semua dosen-dosen program studi Teknik Elektro Universitas Jember.
3. Almamater yang saya banggakan, Program Studi Teknik Elektro Fakultas Teknik Universitas Jember.

Motto

"Jangan pernah menyalahkan rencana Tuhan"

PERNYATAAN

Saya yang bertanda tangan di bawah ini :

Nama : Ahmad Junaidi

NIM : 121910201026

menyatakan dengan sesungguhnya bahwa karya tulis ilmiah yang berjudul "**Analisis Kinerja *Routing Protocol Optimized Link State Routing* (OLSR) dan *Ad Hoc On Demand Distance Vektor* (AODV) Pada *Vehicular Ad Hoc Network* (VANET)**" adalah benar-benar hasil karya sendiri, kecuali kutipan yang sudah saya sebutkan sumbernya, belum pernah diajukan pada institusi manapun, dan bukan karya jiplakan. Saya bertanggung jawab atas keabsahan dan kebenaran isinya sesuai dengan sikap ilmiah yang harus dijunjung tinggi.

Demikian pernyataan ini saya buat dengan sebenarnya, tanpa ada tekanan dan paksaan dari pihak manapun serta bersedia mendapat sanksi akademik jika ternyata dikemudian hari pernyataan ini tidak benar.

Jember, 27 Desember 2019

Yang menyatakan,

Ahmad Junaidi

NIM. 121910201026

SKRIPSI

ANALISIS KINERJA *ROUTING PROTOCOL OPTIMIZED LINK STATE ROUTING* (OLSR) DAN *AD HOC ON DEMAND DISTANCE VEKTOR* (AODV) PADA *VEHICULAR AD HOC NETWORK* (VANET)

Oleh

Ahmad Junaidi

NIM 121910201026

Pembimbing:

Dosen Pembimbing Utama : Dodi Setiabudi, S.T., M.T.

NIP. 198405312008121004

Dosen Pembimbing Anggota : Widya Cahyadi S.T.,M.T.

NIP. 198511102014041001

PENGESAHAN

Skripsi berjudul **"Analisis Kinerja *Routing Protocol Optimized Link State Routing* (OLSR) dan *Ad Hoc On Demand Distance Vektor* (AODV) Pada *Vehicular Ad Hoc Network* (VANET)"** telah diuji dan disahkan pada:

Hari, tanggal : Jum'at, 27 Desember 2019

Tempat : Ruang Ujian 1 Fakultas Teknik Universitas Jember

Dosen Pembimbing Utama,	Dosen Pembimbing Anggota,
Dodi Setiabudi, S.T., M.T.	**Widya Cahyadi S.T.,M.T.**
NIP. 198405312008121004	**NIP. 198511102014041001**

Penguji 1,	Penguji 2,
Andrita Ceriana Eska, S.T., M.T.	**Wahyu Muldayani, S.T., M.T.**
NRP. 760014640	**NRP. 760016799**

Mengesahkan
Dekan,

Dr. Ir. Entin Hidayah, M.U.M.
NIP. 196612151995032001

Analisis Kinerja *Routing Protocol Optimized Link State Routing* (OLSR) dan *Ad Hoc On Demand Distance Vektor* (AODV) Pada *Vehicular Ad Hoc Network* (VANET)

Ahmad Junaidi

Jurusan Teknik Elektro, Fakultas Teknik, Universitas Jember

ABSTRAK

Vehicular Ad Hoc Network (VANET) merupakan salah satu sarana pengembangan teknologi komunikasi nirkabel antar kendaraan yang memungkinkan terjadinya pertukaran data dan pengambilan keputusan secara cepat dan efisien. Proses berkendaraan yang tidak aman didarat cenderung meningkatkan resiko kecelakaan, untuk itu teknologi VANET dikembangkan dengan tujuan dapat memperkecil resiko kecelakaan sehingga meningkatkan kenyamanan berkendara. Pada tugas akhir ini, akan dilakukan penelitian akan dilakukan dengan menggunakan *software* simulasi jaringan NS-2 dengan 2 buah *routing* protocol yaitu *routing protocol proactive Optimized Link State Routing* (OLSR) dan *routing protocol reactive Ad Hoc On-demand Distance Vector* (AODV). Melalui kedua *routing protocol* di atas akan dilakukan perrbandingan performansi menggunakan parameter *Quality of Service* (QoS) dengan sekenario perubahan jumlah dan kecepatan *node*. Setelah dilakukan penelitian terlihat bahwa *routing* protokol OLSR yang tetap memiliki hasil *packet delivery ratio* yang besar daripada *routing* protokol AODV. Karena nilai *packet delivery ratio*nya yaitu 99,33%; 99,41%; dan 99,11%. Hal ini dikarenakan paket yang diterima pada *routing* protokol OLSR lebih besar daripada *routing* yang lainnya. Serta nilai *packet delivery ratio* OLSR selalu stabil. *Routing* protokol OLSR juga tetap memiliki hasil *deley* yang stabil daripada *routing* protokol AODV. Karena nilai *delay* yang diperoleh yaitu 1,174292 s; 1, 174292 s dan 1.187146 s. Dari hasil analisis yang telah dilakukan dapat disimpulkan bahwa dari hasil simulasi pada penelitian ini *routing* protokol OLSR memiliki kinerja yang lebih baik dalam pengiriman data.

Kata Kunci: VANET, NS-2, AODV, OLSR, *Packet delivery ratio* (PDR), *Throughput, dan Delay*

Performance Analysis of Optimized Link State Routing Protocol (OLSR) and Ad Hoc On Demand Distance Vector (AODV) on Vehicular Ad Hoc Network (VANET)

Ahmad Junaidi

Electrical Engineering, Engineering Faculty, Jember University

ABSTRACT

The Vehicular Ad Hoc Network (VANET) is one of the means of developing wireless communication technology between vehicles that enables the exchange of data and decision making quickly and efficiently. The unsafe driving process on land tends to increase the risk of accidents, therefore VANET technology was developed with the aim of minimizing the risk of accidents thereby increasing driving comfort. In this final project, research will be conducted using NS-2 network simulation software with 2 routing protocols, namely the proactive Optimized Link State Routing (OLSR) routing protocol and the Ad Hoc On-demand Distance Vector (AODV) reactive routing protocol. Through the two routing protocols above, performance comparison using the Quality of Service (QoS) parameter will be compared with the scenario of the change in the number and speed of nodes. After doing research, it appears that the OLSR routing protocol that still has a greater packet delivery ratio than the AODV routing protocol. Because the packet delivery ratio is 99.33%; 99.41%; and 99.11%. This is because the packet received on the OLSR routing protocol is bigger than the other routing. And the OLSR packet delivery ratio is always stable. OLSR routing protocol also has a stable delay than the AODV routing protocol. Because the delay value obtained is 1.174292 s; 1, 174292 s and 1.187146 s. From the results of the analysis it can be concluded that from the simulation results in this study the OLSR routing protocol has a better performance in sending data.

.

Keywords: VANET, NS-2, AODV, OLSR, *Packet delivery ratio* (PDR), *Throughput, and Delay*

RINGKASAN

Analisis Kinerja *Routing Protocol Optimized Link State Routing* (OLSR) dan *Ad Hoc On Demand Distance Vektor* (AODV) Pada *Vehicular Ad Hoc Network* (VANET); Ahmad junaidi, 121910201026; 2019; 82 halaman; Jurusan Teknik Elektro Fakultas Teknik Universitas Jember.

teknologi informasi yang praktis dan efisien sangatlah dibutuhkan. Jaringan ad hoc merupakan teknologi *wireless LAN* (WLAN) yang tidak memerlukan suatu infrastruktur seperti *base station* pada jaringannya. Akan tetapi, teknologi *wireless* sering kali memiliki keterbatasan *resource*. Walaupun teknologi jaringan *wireless* berkembang sangat pesat namun, teknologi *wireless* memiliki beberapa kekurangan seperti topologi yang berubah - ubah, *error rate* yang tinggi, konsumsi daya yang besar, keterbatasan *bandwidth*, dan juga jangkauan area yang terbatas. *Vehicular Ad Hoc Network* (VANET) merupakan salah satu sarana pengembangan teknologi komunikasi nirkabel antar kendaraan yang memungkinkan terjadinya pertukaran data dan pengambilan keputusan secara cepat dan efisien. Proses berkendaraan yang tidak aman didarat cenderung meningkatkan resiko kecelakaan, untuk itu teknologi VANET dikembangkan dengan tujuan dapat memperkecil resiko kecelakaan sehingga meningkatkan kenyamanan berkendara. Pada tugas akhir ini, akan dilakukan penelitian akan dilakukan dengan menggunakan *software* simulasi jaringan NS-2 dengan 2 buah *routing* protocol yaitu *routing protocol proactive Optimized Link State Routing* (OLSR) dan *routing protocol reactive Ad Hoc On-demand Distance Vector* (AODV). Melalui kedua *routing protocol* di atas akan dilakukan perrbandingan performansi menggunakan parameter *Quality of Service* (QoS) dengan sekenario perubahan jumlah dan kecepatan *node.* Skenario penelitian yang disusun yaitu skenario perubahan jumlah *node* (6 *node,* 12 *node,* dan 18 *node*), dan perubahan waktu node (10, 20, dan 30 m/s) sebagai tolak ukur performansi jaringan VANET.

Dari hasil penelitian, Pada skenario perubahan jumlah *node* didapatkan bahwa *routing* protokol OLSR yang tetap memiliki hasil *packet delivery ratio* yang

besar daripada *routing* protokol AODV. Nilai *packet delivery ratio*nya yaitu 99,33%; 99,41%; dan 99,11%. Paket yang diterima pada *routing* protokol OLSR lebih besar daripada *routing* yang lainnya. Serta nilai *packet delivery ratio* OLSR selalu stabil. Pada *routing* AODV saat kondisi *node* berjumlah 18 *node*, performansinya menurun yaitu sebesar 98,65%. Sifat AODV yang selalu menyebarkan sinyal ke *node* tetangga untuk mendapatkan *rute* yang diinginkan sehingga nilai *packet delivery ratio* menurun karena sistem ini cocok saat kondisi *node* yang diam. Nilai *throughput* pada *routing* protokol AODV mengalami penurunan seiring penambahan jumlah node yaitu 674,08 kbps, 722,56 kbps, dan 626,13 kbps. Pada *routing* protokol OLSR mengalami peningkatan seiring dengan penambahan jumlah node yaitu 793,60 kbps, 818,72 kbps, dan 759,89 kbps. *Routing* protokol OLSR memiliki hasil *delay* yang stabil daripada *routing* protokol AODV. Nilai *delay* yang diperoleh yaitu 1,174292 s; 1, 174292 s dan 1.187146 s. Nilai *delay* pada *routing* protokol AODV Mengalami kondisi naik turun seiring penambahan jumlah *node* yaitu 1,670176s; 1,368799 s; dan 1,945753 s.

Pada skenario 2 perubahan kecepatan *node* di dapatkan bahwa *routing* protokol OLSR tetap memiliki hasil *packet delivery ratio* yang besar daripada *routing* protokol AODV. Nilai *packet delivery ratio*nya yaitu 99,41%; 99,15%; dan 99,46%. Paket yang diterima pada *routing* protokol OLSR lebih besar daripada *routing* yang lainnya. OLSR merupakan *routing* proaktif yang dimana *routing* tersebut selalu meng-*update* informasi seluruh rute, baik dibutuhkan atau tidak, serta nilai *packet delivery ratio* OLSR selalu stabil. Pada *routing* AODV saat kondisi kecepatan *node* sebesar 20 m/s performansinya menurun yaitu 98,82%. Sifat AODV yang selalu menyebarkan sinyal ke *node* tetangga untuk mendapatkan *rute* yang diinginkan sehingga nilai *packet delivery ratio* menurun karena sistem ini cocok saat kondisi *node* yang diam. Nilai *throughput* pada *routing* protokol AODV mengalami kenaikan seiring penambahan kecepatan pada *node* yaitu 722,56 kbps, 721,23 kbps, dan 745,55 kbps. Nilai *throughput* pada *routing* protokol OLSR mengalami kondisi naik turun seiring dengan penambahan kecepatan *node* yaitu 818,72 kbps, 678,29 kbps dan 719,73kbps. Pada nilai *delay routing* protokol OLSR dan AODV sama – sama mengalami kondisi naik turun, kondisi nilai *delay* terbesar

dari kedua *routing* protokol tersebut pada saat kecepatan *node* sebesar 20m/s yaitu 1,681654 s dan 1,365784 s.

PRAKATA

Puji syukur kehadirat Allah SWT yang maha kuasa atas segalanya, karena dengan ridho, hidayah dan petunjukNya, penulis dapat menyelesaikan skripsi ini. Selama penyusunan skripsi ini penulis mendapat bantuan berbagai pihak yang turut memberikan bantuan berupa motivasi, inspirasi, bimbingan, doa, fasilitas dan dukungan lainnya yang membantu memperlancar pesngerjaan skripsi ini. untuk itu penulis mengucapkan terimakasih kepada.

1. Ibu Dr. Ir. Entin Hidayah, M.U.M., Selaku Dekan Fakultas Teknik Universitas Jember;
2. Bapak Dr. Bambang Sri Kaloko, S.T., M.T., Selaku Ketua Jurusan Teknik Elektro Universitas Jember;
3. Bapak Dodi Setia Budi, S.T., M.T., dan Bapak Widya Cahyadi, S.T., M.T. Selaku dosen pembimbing yang telah membimbing tugas akhir ini;
4. Bapak Andrita Ceriana Eska, S.T., M.T., dan bapak Wahyu Muldayani, S.T., M.T., Selaku dosen penguji yang sudah memberikan saran untuk memperbaiki tugas akhir ini;
5. Kedua Orang tua, Bapak Sutikno dan Almh. Ibu Ernawati yang telah membesarkan, mendidik, mendoakan tiada henti, memberi motivasi semangat, menitikkan air mata dan memberi kasih sayang yang tak pernah habis serta pengorbanannya selama ini;
6. Adik perempuan saya Siti Junaida, Keluarga besar Bani Ardjumo, dan Bani Ali;
7. Keluarga Pak Lek Brother, Teman ngopi, nongkrong yang selalu memberikan do’a, dukungan, bantuan, hiburan dan kenangan yang tidak terlupakan;
8. Keluarga besar Teknik Elektro 2012 SATE UJ, serta semua pihak yang tidak dapat disebutkan satu per satu, yang telah mendukung dalam penyelesaian skripsi ini.

Jember, 27 Desember 2019

Penulis,

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1. PENDAHULUAN

1.1 Latar Belakang

Dewasa ini, teknologi informasi yang praktis dan efisien sangatlah dibutuhkan. Jaringan *ad hoc* merupakan teknologi *wireless LAN* (WLAN) yang tidak membutuhkan suatu infrastruktur *base station* pada jaringannya. Akan tetapi, teknologi *wireless* sering kali memiliki keterbatasan *resource*. Walaupun teknologi jaringan *wireless* berkembang sangat pesat namun, teknologi *wireless* memiliki beberapa kekurangan seperti topologi yang berubah - ubah, *error rate* yang tinggi, konsumsi daya yang besar,keterbatasan *bandwidth*, dan juga jangkauan area yang terbatas. (Irfan, 2016)

Vehicular Ad Hoc Network (VANET) merupakan salah satu sarana pengembangan teknologi komunikasi nirkabel antar kendaraan yang memungkinkan terjadinya pertukaran data dan pengambilan keputusan secara cepat dan efisien. Proses berkendaraan yang tidak aman didarat cenderung meningkatkan resiko kecelakaan, untuk itu teknologi VANET dikembangkan dengan tujuan dapat memperkecil resiko kecelakaan sehingga meningkatkan kenyamanan berkendara. *Vehicular Ad HocNetwork* (VANET) merupakan subset dari *MobileAd Hoc Network* (MANET) yang terkhusus digunakan sebagai teknologi jaringan mobile.

Teknologi *VANET* (*Vehicular Adhoc Network*) berkembang sangat pesat. Penelitian terus dilakukan agar teknologi ini mampu melayani berbagai aplikasi dengan situasi yang berbeda seperti di perkotaan dan pedesaan. Tujuan utama dari *Vehicular Ad-Hoc Networks* adalah membangun jaringan yang kuat antara kendaraan yang terus bergerak sehingga kendaraan dapat saling berkomunikasi untuk keselamatan bagi para pengguna jalan. *VANET* memiliki potensi untuk dikembangkan karena fiturnya yang sangat banyak dan cukup menjanjikan.

Saat ini, sistem komunikasi antar kendaraan atau biasa disebut dengan *Vehicular Communication System* masih terus dikembangkan. Di beberapa kendaraan modern, sistem ini sudah diimplementasikan bahkan di beberapa negara, telah ada regulasi frekuensi radio yang digunakan untuk aplikasi

Vehicular Communication System. Walaupun sudah dikembangkan dan di standarisasi, sistem komunikasi ini belum dapat digunakan secara massal karena beberapa alasan salah satunya yaitu belum ditemukannya sistem *routing* yang tepat untuk aplikasi ini, selain itu, standar jaringan baru yang masih dikembangkan untuk digunakan pada sistem ini membuat teknologi tidak dapat segera diimplementasikan.(Singh, 2011)

Dari latar belakang masalah yang muncul dari penelitian penelitian yang telah dilakukan sebelumnya, penelitian akan dilakukan dengan menggunakan *software* simulasi jaringan NS2 dengan 2 buah *routing* protocol yaitu *routing protocol reactive* AODV (Ad Hoc *On-demand Distance Vector*) dan *routing protocol proactive* OLSR (*Optimized Link State Routing*). Melalui kedua *routing protocol* di atas akan dilakukan perrbandingan performansi menggunakan parameter *Quality of Service* (QoS) untuk menentukan *raouting protocol* yang paling cepat dalam proses pengiriman data diantara keduanya.

1.2 Rumusan Masalah

Dari latar belakang di atas maka diambil rumusan masalah sebagai berikut:

1. Bagaimana performansi kinerja *routing protocol* AODV dan OLSR ?
2. Bagaimana hasil perbandingan dari hubungan perubahan jumlah dan kecepatan *node* pada *routing protocol* AODV dan OLSR?

1.3 Batasan Masalah

Dalam membatasi materi yang akan diteliti pada tugas akhir ini, penulis memberikan batasan sebagai berikut:

1. Pengujian dilakukan berbasis simulasi dengan menggunakan software simulasi NS2.
2. *Routing protocol* yang digunakan *routing protocol routing protocol reactive On-demand Distance Vector Routing* (AODV) dan *Optimized Link State Routing* (OLSR)**.**

3. Parameter yang diuji dalam percobaan ini adalah (*Packet Delivery Ratio* (PDR)**,** *throughput, delay*) yang diukur melalui software simulasi NS2.
4. Pergerakan *node* diatur secara manual dan setiap sekenario mengalami pergerakan yang sama, skenario penelitian yang dilakukan hanya fokus pada jumlah node dan kecepatan *node.*

1.4 Tujuan Penelitian

Adapun tujuan penelitian ini adalah sebagai berikut:

1. Mengetahui kinerja *routing protocol* AODV dan OLSR dengan menggunakan NS-2.
2. Memperoleh hasil perbandingan dari hubungan perubahan jumlah dan kecepatan *node* pada *routing protocol* AODV dan OLSR.

1.5 Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan manfaat sebagi berikut:

1. Peneliti mampu merancang simulasi jaringan NS2 dan menganalisa hasil simulasi nya.
2. Mengetahui perbandingan dari hubungan perubahan jumlah dan kecepatan *node* pada *routing protocol* AODV dan OLSR.

1.6 Sistematika Penelitian

Secara garis besar penyusunan proposal skripsi ini adalah sebagai berikut:

BAB 1. PENDAHULUAN

Berisi tentang latar belakang, rumusan masalah, Batasan masalah, tujuan pembahasan, manfaat pembahasan dan sistematika pembahasan.

BAB 2. TINJAUAN PUSTAKA

Berisi tentang tinjauan pustaka yang menguraikan pendapat-pendapat atau hasil hasil penelitian terdahulu yang berkaitan dengan penelitian yang dilakukan, landasan teori merupakan penjabaran dari tinjauan pustaka.

BAB 3. METODOLOGI PENELITIAN

Menjelaskan tentang metode kajian yang digunakan untuk menyelesaikan tugas akhir.

BAB 4. HASIL DAN PEMBAHASAN

Berisi tentang analisa yang telah didapat dari proses perhitungan.

BAB 5. PENUTUP

Berisi tentang kesimpulan yang diperoleh dari pembuatan tugas akhir ini dan saran dari penulis untuk pengembangan lebih lanjut.

DAFTAR PUSTAKA

LAMPIRAN

BAB 2. TINJAUAN PUSTAKA

Tinjauan pustaka adalah daftar acuan didalam melakukan penelitian tugas akhir ini. Tinjauan pustaka digunakan untuk memperdalam wawasan didalam mengembangkan pengetahuan didalam melakukan penelitian sehingga terjadi peningkatan dan perkembangan.

2.1 Matriks Perumusan Masalah

Tabel 2.1 Matriks Perumusan Masalah

No	Masalah	Solusi	Metode	Hasil	Judul/Tahun
1	Teknologi jaringan *wireless* berkembang sangat pesat namun, teknologi *wireless* memiliki beberapa kekurangan seperti topologi yang berubah – ubah, *error – rate* yang	Menguji jaringan *wireless* dengan standar IEEE 802.11ah dan *routing* Proaktif *DSDV* dan *OLSR* pada skenario *node* bergerak secara acak.	• komunikasi manet dengan *Routing DSDV* dan *OLSR* menggunakan NS-3 Pergerakan *node* menggunakan model *Random direction mobility model*	• *Throughput* • *Packet Delivery Ratio* Rata Rata *Delay*	Irfan Denatama, (2016). "Analisis Perbandingan Kinerja Protokol *Routing DSDV* dan *OLSR* untuk Perubahan Kecepatan Mobilitas pada Standar IEEE 802.11ah"

	tinggi, konsumsi daya yang besar,keterbatasan *bandwidth*, dan jangkauan.				
2.	Masalah terbesar pada system *VANET* adalah implementasi yang cukup mahal dan menyeluruh, serta sistem *routing* yang belum optimal	Pengujian sistem *routingAODV* pada 2 *software simulator* trafik yang berbeda dengan parameter yang sama.	• Simulasi *software* menggunakan peta nyata yang di hubungkan dengan *software* simulator trafik yakni SUMO dan *VanetMobisim* dengan hasil output di proses kembali menggunakan *software* NS2	• *Packet Delivery Ratio* • *End to End Delay* • *Routing Overhead*	Firdaus Nutrihadi, (2016). "Studi Kinerja VANET *Scenario Generators: SUMO* dan *VanetMobisim* untuk Implementasi *Routing Protocol AODV* menggunakan *Network Simulator 2 (NS2)"*
3.	Perlunya dipelajari protokol	Pengujian sistem *routing OLSR*	• Simulasi komunikasi manet	• *Through put* • *Overhea*	*Analisis Perbandinga n Routing*

	routing baru yang mengadopsi kebaikan OLSR dan AODV tetapi sekaligus dapat mengurangi kelemahan keduanya.	*dan AODV* bisa menggunakan *software* lain, dan menambah jumlah parameter pembanding	dengan *Routing OLSR* dan *AODV* menggunakan NS2	• *Delay*	*Protokol OLSR (Proaktif) dan AODV (Reaktif) pada MANET*
4.	Perlunya standar jaringan *wireless* dengan parameter yang sesuai dengan karakteristik dari lalu lintas untuk diaplikasikan di dalam sistem *VANET*.	Menggunakan standar *wireless* IEEE802.11p yang telah dirancang khusus untuk aplikasi *DSRC*.	• Melakukan simulasi untuk membandingkan kinerja standar *wireless* 802.11a dengan 802.11p	• *End-to-End Delay* • *Packet Delivery Ratio*	Kapil Dev, Sarita Singh. (2013). "*Implementation of Dedicated Short Range Communication (DSRC) IEEE802.11p in NS2 and its Perfomance Analysis Over IEEE802.11*".
5.	Perbedaan	Menguji	• Melakukan	• *PDR vs*	Pranav

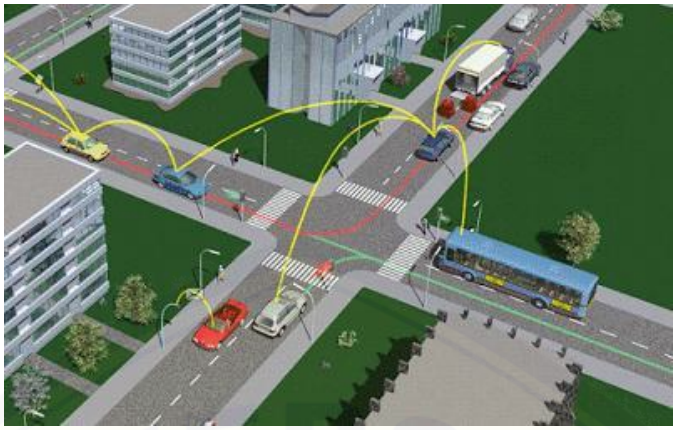
	kepadatan *node* pada skenario *VANET* di wilayah *urban* dan *highway* akan mempengaruhi kinerja algoritma *routing* yang digunakan.	protokol *routing AODV, OLSR,* dan *DSR* pada 2 skenario VANET yakni *highway* dan *urban.*	simulasi untuk mendapatkan parameter kualitas jaringan menggunakan *software* NS2 dan *Mobility generator MOVE* pada simulator lalu lintas *SUMO.*	*Node Density* • *End-to-End delay vs node density*	Kumar Singh, Kapang lego, Dr. Thermichon Tuithung. (2011).“*Simulation based Analysis of Adhoc Routing Protocol in Urban and Highway Scenario of VANET”*
6.	Berkembangnya sistem komunikasi antar keadaan (*VANet*) namun terbatasnya kemampuan *routing* pada *nodenode* bergerak tanpa bantuan	Menguji Sistem *VANET* pada *software* simulasi dengan parameter *routing* dan *data rate* yang berbeda.	• Menguji 3 Buah *Routing* Yaitu *AODV, DSDV* dan *DSR* • Membandingkan Nilai *Throughput, Delay* serta *PDF* berdasarkan Kecepatan *Node, Transmit*	• *Delay* • *Packet Delivery Fraction (PDF)* • *Through put*	Arifin, (2011). “Analisis Performansi *Routing AODV* pada Jaringan *VANet”*

	infrastruktur		*Power*, dan *Data Rate* yang berbeda		

Dari seluruh penelitian yang dilakukan pada matriks diatas, secara umum, pengujian performansi jaringan *wireless* dilakukan dengan proses simulasi dengan menggunakan *software Network Simulator* baik NS2 maupun NS-3. Sebagian besar penelitian juga menguji kinerja berbagai macam *routing protocol* dengan skenario simulasi yang bebeda-beda. Selain itu penelitian di atas juga pada umumnya mengganti standar jaringan wireless yang digunakan, diantaranya 802.11a, p, ah dan n. Hal ini dilakukan peneliti dengan tujuan untuk mencari standar jaringan *wireless*yang sesuai dengan aplikasi VANET. Dari seluruh penelitian tersebut, didapatkan data mengenai nilai parameter *Quality of Service* dari *routing*, standar jaringan wireless, dan juga pengaruh kepadatan *node*. Pada tugas akhir ini, akan dilakukan penelitian akan dilakukan dengan menggunakan *software* simulasi jaringan NS-2 dengan 2 buah *routing* protocol yaitu *routing protocol proactive Optimized Link State Routing* (OLSR) dan *routing protocol reactive Ad Hoc On-demand Distance Vector* (AODV). Melalui kedua *routing protocol* di atas akan dilakukan perrbandingan performansi menggunakan parameter *Quality of Service* (QoS) dengan sekenario perubahan jumlah dan kecepatan *node.*

2.2 VANET (Vehicular Ad-Hoc Network)

Vehicular Ad hoc Network (VANET) adalah sebuah bentuk baru dalam komunikasi data untuk kendaraan yang bergerak dengan kecepatan tinggi dan pertukaran informasi terjadi di jalan raya. VANET merupakan kategori dari *Mobile Ad-hoc Network* (MANET) di mana *node* tediri dari kendaraan dan *Roadside Unit* (RSU). VANET telah menjadi sebuah infrastruktur komunikasi penting bagi ITS.



Gambar 2.1 *Vehicular Ad.Hoc Network* (VANET)
(sumber : Abduh I, Dahlia. N, dkk. 2017)

VANET menggunakan *Dedicated Short Range Communication* (DSRC) untuk peningkatan keselamatan mengemudi dan kenyamanan dalam berkendara. Jaringan VANET memadukan kemampuan komunikasi antar kendaraan dan komunikasi antar kendaraan dengan infrastruktur sepanjang jalan. Setiap kendaraan dapat bergabung ke jaringan VANET melalui komunikasi nirkabel antar kendaraan atau ke infrastruktur. Sehingga memungkinkan kendaraan dalam jarak 100 sampai 300 meter dapat saling terhubung satu sama lain dan pada gilirannya, membentuk sebuah jaringan dengan cakupan yang luas.(Abduh I, Dahlia. N, dkk. 2017)

2.2.1 Komunikasi Vanet

a. *Vehichel to Vehicle Communication* (V2V) komunikasi yang terjadi antara satu *node* dengan node lainnya di dalam jaringan komunikasi.
b. *Vehicle to Infrastruktur Communication* (V2I) komunikasi yang terjadi antara *node* dengan infrastruktur yang berada di jalan raya.
c. *Vehicle to Roadside* (V2R)
d. Gabungan antara komunikasi V2V dan V2I

2.2.2 Karakteristik VANET

a. Topologi dinamis tingkat tinggi perubahan topologi pada VANET disebabkan oleh pergerakan dari kendaraan dengan kecepatan tinggi.

b. Sering terputusnya jaringan hasil dari topologi dinamis dapat diamati bahwa pemutusan sering terjadi antara dua kendaraan ketika sedang bertukar informasi.
c. Pemodelan mobilitas pola mobilitas kendaraan tergantung pada lingkungan lalu lintas , jalan terstruktur, kecepatan kendaraan, perilaku mengemudi dan sebagainya.
d. Daya baterai dan kapasitas penyimpanan dalam kendaraan daya baterai dan penyimpanan tidak terbatas. Hal ini berguna untuk komunikasi yang efektif dan membuat rute keputusan.

2.3 Routing Protokol

2.3.1 Proaktive routing

Algoritma dari Golongan protokol ini akan mengelola daftar tujuan dan rute terbaru masing-masing serta bersifat broadcast sehingga sistem pendistribusian table routingnya selalu diupdate secara periodik, Maka dari itu perlu penggambaran keseluruhan node jaringan serta setiap node akan merespon perubahan dalam mengupdate agar terjadi konsistensi routing tabel . Hal ini akan memperlambat aliran data jika terjadi restrukturisasi routingnya. Beberapa contoh algoritma *proactive routing* yaitu *Intrazone Routing Protocol* (IARP), *Linked Cluster Architecture* (LCA) , *Witness Aided Routing* (WAR) , *Optimized Link State Routing* (OLSR) , *Better Approach to Mobile Ad hoc Network* (BATMAN) , *Highly Dynamic Destination Sequenced Distance Vector routing protocol* (DSDV), *Fisheye state routing* (FSR).

2.3.2 Reactive Routing

Algoritma protokol *Reactive Routing* bersifat on demand ,pada intinya node sumber yang akan menentukan node tujuan sesuai prosedur yang diinginkannya . Jadi *routing table* yang ada pada *node* hanyalah informasi route ke tujuan saja. Beberapa contoh algoritma reactive routing adalah *Associativity Based Routing* (ABR), *Ad Hoc On Demand Distance Vector* (AODV),*Ad Hoc On Demand Multipath Distance Vector*,*Backup Source Routing* (BSR), *Dynamic*

Source Routing (DSR),*Ant Routing algorithm for mobile adhoc networks* (ARAMA). (Endah Maya Megawati, 2016)

2.4 *Optimized Link Stated Routing* (OLSR)

Optimized Link Stated Routing (OLSR) merupakan jenis *link stated routing* yang dioptimalkan untuk *mobile ad-hoc network*. OLSR juga merupakan salah satu tipe protokol *routing* yang bersifat *proactive* atau *table driven*. Dengan sifat proaktif, rute komunikasi akan segera disediakan saat ada dua node yang ingin berkomunikasi, hal ini dikarenakan informasi *routing* yang ada sebelumnya disimpan pada *table routing*. Berbeda dengan protokol *routing reactive* yang hanya akan menyediakan rute komunikasi saat dibutuhkan saja. Untuk pemeliharaan informasi *routing*pada *routing* table, setiap waktu OLSR perlu meng-update informasi rute yang ada dengan cara melakukan*flooding broadcast packet* pada semua node yang ada pada jaringan manet. Namun jika flooding dilakukan pada semua node yang ada akan mengakibatkan terjadinya *overload data*. Untuk mengatasi masalah ini OLSR memiliki *node multi point relay* (MPR)yang akan memilih *node* tertentu saja yang akan menerima *broadcast packet*. Kemampuan ini juga menjadikan OLSR memiliki kemungkinan *routing overhead* paling rendah disbanding dengan protokol *routing* lain.(Zahrul Ahmad, Tanpa Tahun)

2.4.1 Tahapan Kerja OLSR

Secara umum langkah-langkah kerja dalam OLSR dapat diurutkan sebagai berikut:

a. Link Sensing (pendeteksian hubungan)

   Link sensing dilakukan dengan mengirimkan pesan HELLO secara periodic dan berkesinambungan. Hasil dari *link sensing* adalah *local link set* yang menyimpan informasi hubugan antara interface yang ada pada node tersebut dengan node-node tetangga.

b. Neighbour detection (pendeteksian node tetangga)

   Node mengirim pesan HELLO akan menerima informasi alamat alamat dari node tetangganya beserta link statusnya.

c. MPR selection

MPR selection (memilih MPR) melalui pesan HELLO node utama akan menentukan sejumlah node tetangga untuk dipilih sebagai *multipoint relay* (MPR) yang bertugas meneruskan paket-paket control ke dalam jaringan.

d. Pengiriman TC (Topology Control) messages.

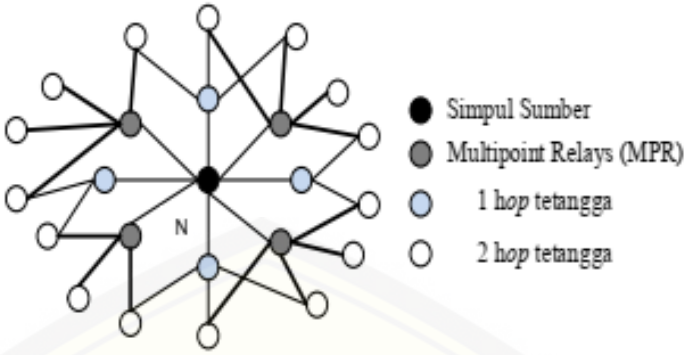
TC Messages dikirimkan untuk memberikan informasi routing kepada setiap node yang ada pada jaringan yang akan digunakan untuk penentuan jalur.

e. Route calculation (perhitungan jalur)

Berdasarkan informasi rute yang didapat dari paket-paket control seperti HELLO dan TC maka setiap node akan memiliki routing table yang berisi informasi rute yang dapat dilalui untuk dpat mengirimkan data ke node-node ainnya yang ada pada jaringan.

2.4.2 Pemilihan MPR

Tujuan dari Multipoint Relay (MPR) adalah meminimalisir penggunaan overhead yang pesan broadcast pada jaringan dengan cara mengurangi overhead yang pesan broadcast pada jaringan dengan cara mengurangi retransmisi (pentransmisian ulang) pada daerah yang sama. Setiap *node* pada jaringan akan memilih sejumlah *hop* tetangga 1-*hop*nya yang bersifat simetris yang akan melakukan transmisi ulang pesan-pesannya. Sejumlah *node* tetangga itulah yang disebut MPR. Setiap tetangga yang tidak terpilih sebagai *node* MPR akan tetap menerima pesan tersebut dan memproses namun tidak akan meneruskan atau mengirimkan kembali pesan-pesan tersebut. Pemilihan node untuk dijadikan MPR selain harus bersifat simetris juga harus dapat menjangkau sejumlah node tetangga 2-hop. Makin sedikit jumlah MPR maka penggunaan *control traffic overhead* yang digunakan dalam *protocol routing* makin sedikit. Perbandingan kinerja pengiriman paket untuk OLSR dan *link state protocol* pada umumnya.



Gambar 2.2 Pemilihan simpul MPR pada OLSR
(sumber : Alamsyah, Eko. S, dkk.2018)

2.4.3 Algoritma Pemilihan MPR

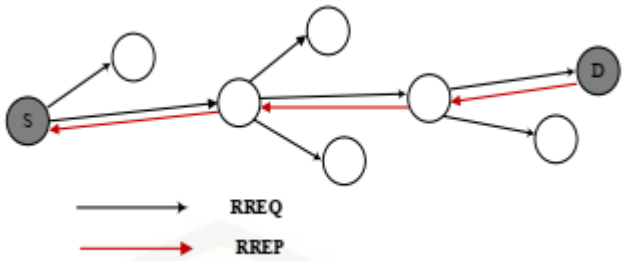
Langkah-langkah pemilihan MPR adalah sebagai berikut:

a. Setiap node melakukan broadcast ke node tetangga 1 hop secara berkala dengan menggunakan hello messages.
b. Berdasarkan broadcast, setiap node memilih subset node tetanggan 1 hop yang terkecil yang bisa menjangkau semua node tetangga 2 hop. Subset node yang terpilih ini lalu dijadikan node MPR.
c. Node MPR melakukan broadcast topology control (TC) messages setiapinterfal TC untuk menginformasikan link state.
   - TC Message berisi tetangga 1 hop yang terpilih sebagai MPR.
   - Hanya MPR yang dapat meneruskan TC Message.
   - TC mMessage digunakan untuk perhitungan routing table.

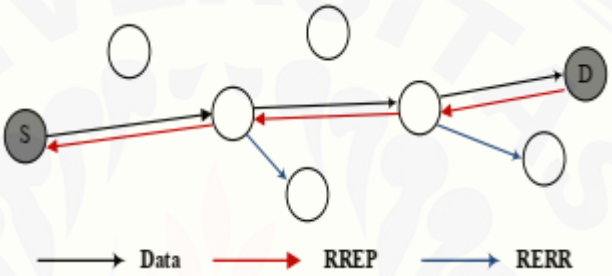
2.5 *Ad Hock On Demand Distance Vektor* (AODV)

AODV adalah *on demand routing*, dimana algoritma ini akan membangun rute antara node hanya apabila diinginkan oleh *source node*. AODV memelihara rute tersebut sepanjang masih dibutuhkan oleh *source node*. AODV menggunakan sequence number untuk memastikan bahwa rute yang dihasilkan adalah *loop-free* dan memliki informasi routing yang paling update. AODV menciptakan suatu rute dengan menggunakan *route request* (RREQ) dan *route reply* (RREP). Ketika

source node menginginkan suatu rute menuju *destination node* tetapi belum mempunyai rute yang benar, maka source node akan menginisialisasi route *discovery process* untuk menemukan rute ke *destination* n*ode*. *Source node* akan mem-broadcast paket RREQ menuju node tetangganya . RREQ paket berisi *source addres, destination adress, hop counter, source and destinasion sequence number,* dan *broadcast ID.* Nlai Broadcast ID akan bertambah satu setiap *source node* mengirimkan RREQ yang baru dan digunakan sebagai identifikasi sebuah paket RREQ. Jika *node* yang menerima RREQ memiliki informasi rute menuju *destination node,* maka node tersebut akan mengirim paket RREP kembali menuju *source node*. Tetapi jika tidak mengetahui maka *node* tersebut akan membroadcast ulang RREQ ke node tetangganya setelah menambahkan nilai hop *counter*. Node yang menerima RREQ dengan nilai *source addres and broadcast ID* yang sama dengan nilai RREQ yang diterima sebelumnya akan membuang RREQ tersebut. Pada saat RREQ mengalir menuju *node* yang diinginkan maka *node* tersebut akan menciptakan *reverse path* menuju *node,* setiap *node* akan membaca RREQ dan mengidentifikasi alamat *node* tetangga. Ketika *destination node* atau *node* yang memiliki informasi *rute* menuju *destination* menerima RREQ maka node tersebut akan membandingkan nilai *destination sequence number* yang dia miliki dengan nilai *destination sequence number* yang ada di RREQ. Jika nilai *destination sequence number* yang ada di RREQ lebih besar dari nilai yang dimiliki oleh node maka paket RREQ tersebut akan di*broadcast* kembali ke node tetangganya, sebaliknya jika nilai pada *destination sequence number* yang ada di *node* lebih besar atau sama dengan nilai yang ada di RREQ maka node tersebut akan mengirim (RREP) menuju *source* node dengan menggunakan *reverse path* yang telah dibentuk oleh RREQ . *Intermediate* node yang menerima RREP akan mengupdate informasi timeout (masa aktif rute) jalur yang telah diciptakan. Informasi *rute source* ke *destination a*kan dihapus apabila waktu timeoutnya habis.



Gambar 2.3 Mekanisme penemuan *rute*
(sumber : Alamsyah, Eko. S, dkk.2018)



Gambar 2.4 Mekanisme pengiriman data dan kesalahan *rute*
(sumber : Alamsyah, Eko. S, dkk.2018)

Di dalam AODV setiap *node* bertanggung jawab untuk memelihara informasi rute yang telah disimpan di dalam routing table nya. Pada saat pengiriman data apabila terjadi perubahan topologi yang mengakibatkan suatu node tidak dapat dituju dengan menggunakan informasi rute yang ada di routing table, maka suatu node akan route error packet (RRER) ke node tetangganya dan node tetangganya akan mengirim kembali RRER demikian seterusnya hingga menuju source node .Setiap node yang memperoleh RRER ini akan menghapus informasi yang mengalami error di dalam routing table-nya. Kemudian source node akan melakukan route discovery process kembali apabila rute tersebut masih diperlukan.

2.6 Network Simulator (NS)

NS adalah suatu simulator jaringan terkendali atau c*up*likan yang dikembangkan pada *UC Berkeley* yang menirukan variasi jaringan IP. Simulator ini menempatkan protokol jaringan seperti TCP dan *UP*D, perilaku sumber traffik seperti FTP, Telnet, Web, Cbr Dan Vbr, mekanisme pengembangan antrian pengarah seperti Ekor Juntai, RED dan CBQ, mengarahkan tahapan seperti Dijkstra dan sebagainya. NS juga menempatkan multicasting dan sebagian dari lapisan protokol MAC untuk simulasi LAN. Proyek NS saat ini mer*up*akan bagian dari proyek VINT yang mengembangkan simulasi dari tampilan hasil, analisa, pengubah yang mengubah topologi jaringan yang dihasilkan oleh generator ternama untuk format-format NS. Saat NS (versi 2) yang ditulis dalam C++ dan OTcl (bahasa script Tcl dengan Perluasan objek-berorientasi yang dikembangkan oleh MIT) yang ada.

Komponen utama lainnya pada NS di samping *object* jaringan adalah penjadwal kejadian. Sebuah kejadian dalam NS mer*up*akan suatu paket ID yang unik untuk sebuah paket dengan waktu yang terjadwal dan penunjuk untuk suatu objek yang menangani masalah kejadian. Dalam NS, suatu penjadwal kejadian menjaga jalur dari waktu simulasi dan membawa seluruh kejadian dalam antrian kejadian terjadwal untuk saat ini dengan bantuan komponen jaringan yang sesuai, di mana biasanya hal tersebut mer*up*akan salah satu hal yang disebutkan dalam kejadian, dan membiarkan hal tersebut menjalankan tindakan yang sesuai yang diwakilkan dengan paket yang ditunjuk oleh kejadian. Komponen jaring berkomunikasi dengan salah satu dari berbagai paket yang berlalu, bagaimanapun komponen ini tidak memakan waktu simulasi actual.

Seluruh komponen jaringan yang dibutuhkan untuk mengisi beberapa waktu simulasi untuk mengatasi satu paket dengan catatan membutuhkan penundaan, memakai pejadwal kejadian dengan mengeluarkan sebuah kejadian untuk paket dan menuggu berhentinya kejadian pada bagiannya sendiri, sebelum melakukan tindakan yang lebih jauh dalam menghandel paket tersebut. Sebagai contoh, suatu komponen saklar jaringan yang mensimulasikan suatu saklar dalam 20 mikro sekon pada pensaklaran mengeluarkan suatu kejadian untuk disaklarkan

terhadap penjadwal sebagai suatu kejadian 20 mikro sekon untuk selanjutnya. Penjadwal setelah 20 mikro sekon mengurutkan ulang kejadian dan membawanya pada komponen saklar, di mana kemudian melewatkan paket tersebut pada suatu komponen jalur keluaran yang sesuai kegunaan lain dari pejadwal kejadian sebagai pewaktu.

NS yang tertulis tidak hanya dalam OTcl tetapi juga dalam C++. Untuk suatu alasan yang *efisien*, NS memisahkan penerapan jalur data dari penerapan jalur kontrol. Untuk mengurangi paket dan kejadian waktu pemrosesan, pejadwal kejadian dan objek komponen jaringan dasar pada saluran data ditulis dan dikompilasi dengan menggunakan C++. Objek yang telah dikompilasi tersebut disesuaikan untuk penerjemah OTcl melalui petalian OTcl yang menciptakan suatu objek OTcl yang cocok untuk setiap objek C++ dan membuat fungsi kontrol dan variable yang dapat dikonfigurasikan lalu dispesifikasikan oleh objek C++ yang bertindak sebagai fungsi anggota dan variable anggota pada penyesuaian objek OTcl.

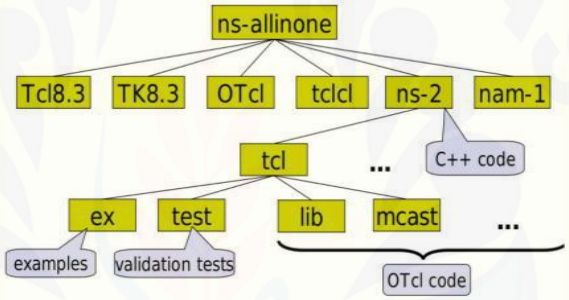
Dalam jalan ini, kontrol dari objek C++ ditujukan untuk OTcl. Hal ini juga mungkin dilakukan untuk menambahkan fungsi anggota dan *variable* objek OTcl C++ terjalur. Objek-objek pada C++ yang tidak perlu dikontrol dalam simulasi atau secara internal yang digunakan oleh objek lain tidaklah perlu dijalurkan pada OTcl. Demikian juga suatu objek (bukan dalam jalur data) secara keseluruhan dapat diterapkan dalam OTcl.

2.6.1 Struktur NS

Ns dibangun menggunakan metoda object oriented dengan bahasa C++ dan OTcl (*variant object oriented* dari Tcl). Seperti terlihat pada Gambar 4, ns-2 menginterpretasikan script simulasi yang ditulis dengan OTcl. Seorang user harus mengeset komponen-komponen (seperti objek penjadwalan *event*, library komponen jaringan, dan library modul set*up*) pada lingkungan simulasi.

User menuliskan simulasinya dengan script OTcl, dan menggunakan komponen jaringan untuk melengkapi simulasinya. Jika user memerlukan komponen jaringan baru, maka user dengan bebas untuk menambahkan dan mengintegrasikan pada simulasinya atau pada ns-2. penjadwalan *event* (*event*

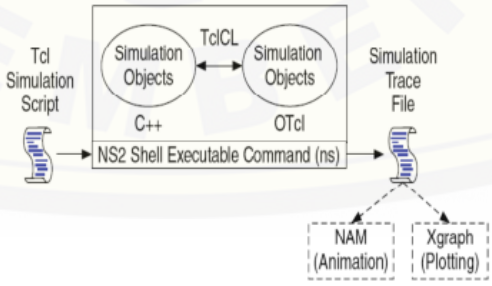
scheduler) berfungsi sebagai komponen utama selain pencetus (trigger) *event* komponen jaringan simulasi (seperti mengirim paket, memulai dan menghentikan tracing). Sebagian dari ns-2 ditulis dalam Bahasa C++ untuk alasan efisiensi. Jalur data (data path), ditulis dalam Bahasa C++, dipisahkan dari jalur kontrol (control path), ditulis dalam Bahasa OTcl. Objek jalur data dikompilasi dan kemudian interpreter OTcl melalui OTcl linkage (tclcl) yang memetakan metode dan variabel pada C++ menjadi objek dan variabel pada OTcl. Objek C++ dikontrol oleh objek OTcl. Hal ini memungkinkan untuk menambahkan metode dan variabel kepada C++ yang dihubungkan dengan objek OTcl. Hirarki linked class pada C++ memiliki korespondansi dengan OTcl, hal ini dapat dilihat pada Gambar dibawah ini:



Gambar 2.5 komponen pembangunan NS2

(sumber : Rozaki A. F (2015)

Hasil yang dikeluarkan oleh ns-2 ber*up*a file trace, harus diproses dengan menggunakan *tool* lain, seperti Network Animator (NAM), perl, awk, atau *gnuplot.*



Gambar 2.6 Arsitektur dasar dari NS2

(sumber : Rozaki A. F (2015)

2.7 IEEE 802.11ac

802.11ac adalah standar yang ditetapkan IEEE sebagai penerus teknologi Wi-Fi generasi kelima. Kemunculan standar ini dilandasi oleh semangat optimisme yang tinggi untuk meningkat produktivitas kerja disaat kapanpun dan dimanapun serta didorong oleh perubahan kebutuhan pasar yang tinggi untuk mendapatkan throughput yang sangat tinggi pada lingkungan multi-user.Kunci utama untuk memperoleh datarate yang sangat tinggi pada jaringan area lokal nirkabel adalah meningkatkan spesifikasi yang lebih baik daripada yang telah dimilik sebelumnya. Pada rancangan 802.11ac, sejumlah peningkatan dilakukan meliputi: pelebaran pita kanal (80 dan 160 MHz), peningkatan aliran data (spatial data streams) sampai dengan delapan streams, dan menambahkan metode 256 Quadrature Amplitude Modulation (256-QAM) untuk mengefisienkan teknik modulasi pada lapisan PHY. Untuk mendukung hal tersebut IEEE menetapkan bahwa 802.11ac beroperasi pada pita frekuensi dibawah 6 GHz dan teknik antena yang digunakan adalah Multiple-User MIMO (MU-MIMO). Standar ini juga menyederhanakan transmisi beamforming dan memperbesar agregasi frame-frame data pada lapisan MAC. (Afdhal, 2014)

2.8 IEEE 802.11p

IEEE 802.11p adalah amandemen yang disetujui untuk standar IEEE 802.11 yang bertujuan untuk menambahkan akses wireless pada sistem Transportasi. Teknologi ini merupakan peningkatan pada standar 802.11 (Lebih dikenal dengan teknologi Wi-Fi) untuk mendukung aplikasi Intelligent Transportation Systems (ITS). Peningkatan ini mencakup kecepatan pertukaran data antar kendaraan yang berkecepatan tinggi dan pertukaran data antara kendaraan dan infrastruktur jalan, yang disebut komunikasi V2X. Teknologi ini bekerja di frekuensi band ITS yang memiliki lisensi 5,9 GHz (5,85-5,925 GHz).

2.9 *Throughput*

Throughput adalah kemampuan sebenarnya suatu jaringan dalam melakukan pengiriman data. Biasanya *throughput* dikaitkan dengan *bandwidth*.

Karena *throughput* juga bisa disebut dengan *bandwidth* dalam kondisi yang sebenarnya. Sementara *throughput* sifatnya adalah dinamis tergantung trafik yang sedang terjadi. Semakin besar *bitrate* maka akan semakin besar pula *throughput* nya, Semakin besar nilai *throughput* nya akan menunjukkan semakin bagus pula kemampuan jaringan dalam mentransmisikan file. (Ahmad Afis Abror, M.Zen Samsono Hadi, Idris Winarno), *troughtput* dapat di peroleh dari persamaan berikut:

$$Throight\ put = \frac{\sum Received\ paket\ size}{\sum Delevery\ time} \qquad (2.1)$$

2.10 Packet Delivery Ratio (PDR)

Total paket yang diterima berbanding total paket yang dikirim. Rasio dari angka paket data yang berhasil terkirim ke tujuan yang di-*generate*oleh sumber CBR (*Constant Bit Rate)*. Rasio paket yang dikirim menjelaskan tingkat kehilangan (*loss rate*). Hal ini menunjukkan kelengkapan dan akurasi dari protocol *routing.* (Sachan P. & Khilar, P.M., 2011). *Packet delevery ratio* dapat diproleh dengan persamaan berilut:

$$PDR = \frac{Total\ packets\ received}{Total\ packets\ sent}\ X\ 100\% \qquad (2.2)$$

2.11 End-to-end delay

End-to-end delay dapat definisikan sebagai selisih waktu pengiriman sebuah paket saat dikirimkan dengan saat paket tersebut diterima pada node tujuan. Delay merupakan suatu indikator yang cukup penting untuk perbandingan protokol routing, karena besarnya sebuah *delay* dapat memperlambat kinerja dari routing protocol tersebut. (Endah Maya Megawati, 2016). Secara umum delay rata-rata dapat dinyatakan dengan persamaan berikut :

$$DELAY = \frac{T_{recived\ packet\ destinationn} - T_{packet\ sent\ source}}{packet\ received} \qquad (2.3)$$

BAB 3. METODOLOGI PENELITIAN

3.1 Tempat dan Waktu Penelitian

3.1.1 Tempat Penelitian

Penelitian ini akan dilaksanakan di Laboratorium Telekomunikasi Terapan Fakultas Teknik Universitas Jember Jl Slamet Riyadi No 26 Patrang.

3.1.2 Waktu Penelitian

Penelitian berjudul Analisis kinerja *Routing Protocol Optimized Link State Routing* (OLSR) dan Ad Hoc On Demand Distance Vektor Routing (AODV) pada *vehicular ad hoc network* (VANET), dengan jadwal pelaksanaan sebagai berikut :

Tabel 3.1 Tabel Waktu Penelitian

No	Kegiatan	Bulan / Minggu											
		Bulan ke-1				Bulan ke-2				Bulan Ke-3			
		1	2	3	4	1	2	3	4	1	2	3	4
1	Studi Literatur												
2	Perancangan Simulasi dan Pengambilan Data												
3	Analisa Data												
4	Penyusunan Laporan												

3.2 Alat dan Bahan

Alat yang digunakan pada penelitian ini berupa perangkat lunak yang digunakan untuk menjalankan software simulasi. Beberapa perangkat lunak yang digunakan yaitu :

1. *Laptop*
2. *Windows 7*

3. *Software Cygwin*
4. *Software NS2*

3.3 Parameter Simulasi

Sebelum membangun simulasi, kita tentukan parameter simulasi yang akan dibuat supaya simulasi berjalan sesuai yang diinginkan. Parameter-parameter tersebut akan dijelaskan pada Tabel 3.2 .

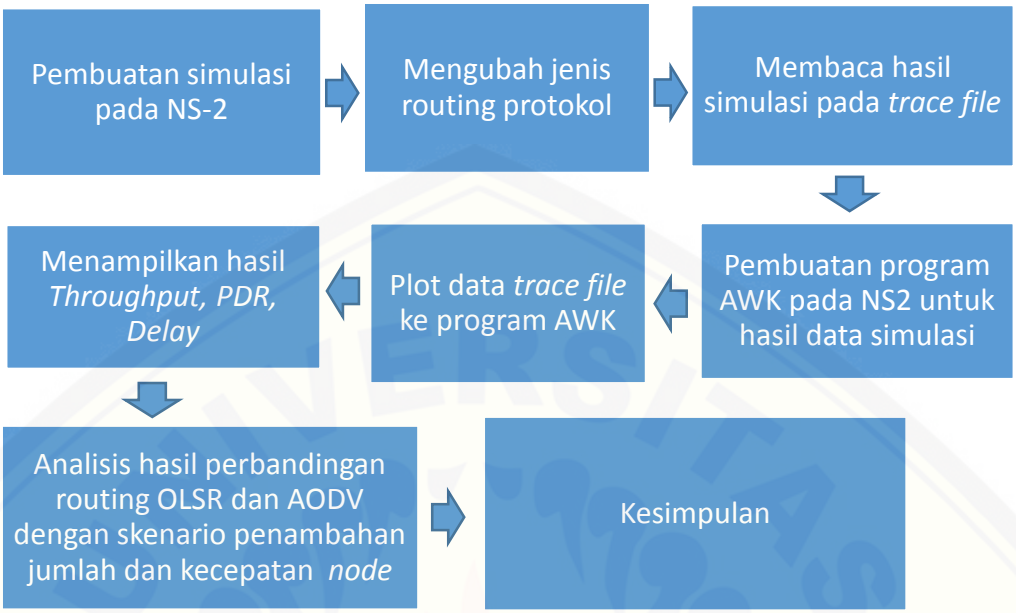
Tabel 3.2 Parameter Simulasi Jaringan

Parameter	Spesifikasi
Routing Protokol	AODV/OLSR
Waktu Simulasi	100 detik
Jumlah *node* maksimal	18 *node*
Area Simulasi	1000x 1000 meter
Tipe trafik	FTP
Network simulator	Ns.2.35

3.4 Tahapan Penelitian

3.4.1 Diagram Blok Sistem Penelitian

Pada penelitian ini terdapat blok diagram sistem sebagai gambar dari alur penelitian seperti gambar di bawah ini:



Gambar 3.1 Diagram Blok sistem penelitian

Dari Blok diagram diatas dapat dijelaskan sebagai berikut:

a. Pembuatan simulasi pada NS2
b. Mengubah jenis *routing* protokol

   Jadi pada simulasi pene;itian ini menggunakan *routing protocol* OLSR dan AODV yang akan dianalisis performansinya.
c. Membaca hasil simulasi pada trace file

   Setelah progam dijalankan, akan menghasilkan data hasil simulasi yang terdapat pada *trace file* untuk pengambilan data dari simulasi jaringan yang telah dijalankan.
d. Pembuatan program AWK pada NS2 untuk hasil data simulasi

   Jadi setelah data dari *trace file* didapatkan, selanjutnya pembuatan program AWK untuk diagram parameter yang digunakan.
e. *Plot* data *trace file* ke program AWK

   Kemudian data tersebut diplot pada program AWK yang telah dibuat untuk tiap parameter yang digunakan.
f. Menampilkan hasil *throughput, packet delivery ratio* dan *delay*.

Jadi setelah data diplot ke program AWK, maka akan ditampilkan hasil data *throughput*, *packet delivery ratio* dan *delay* .

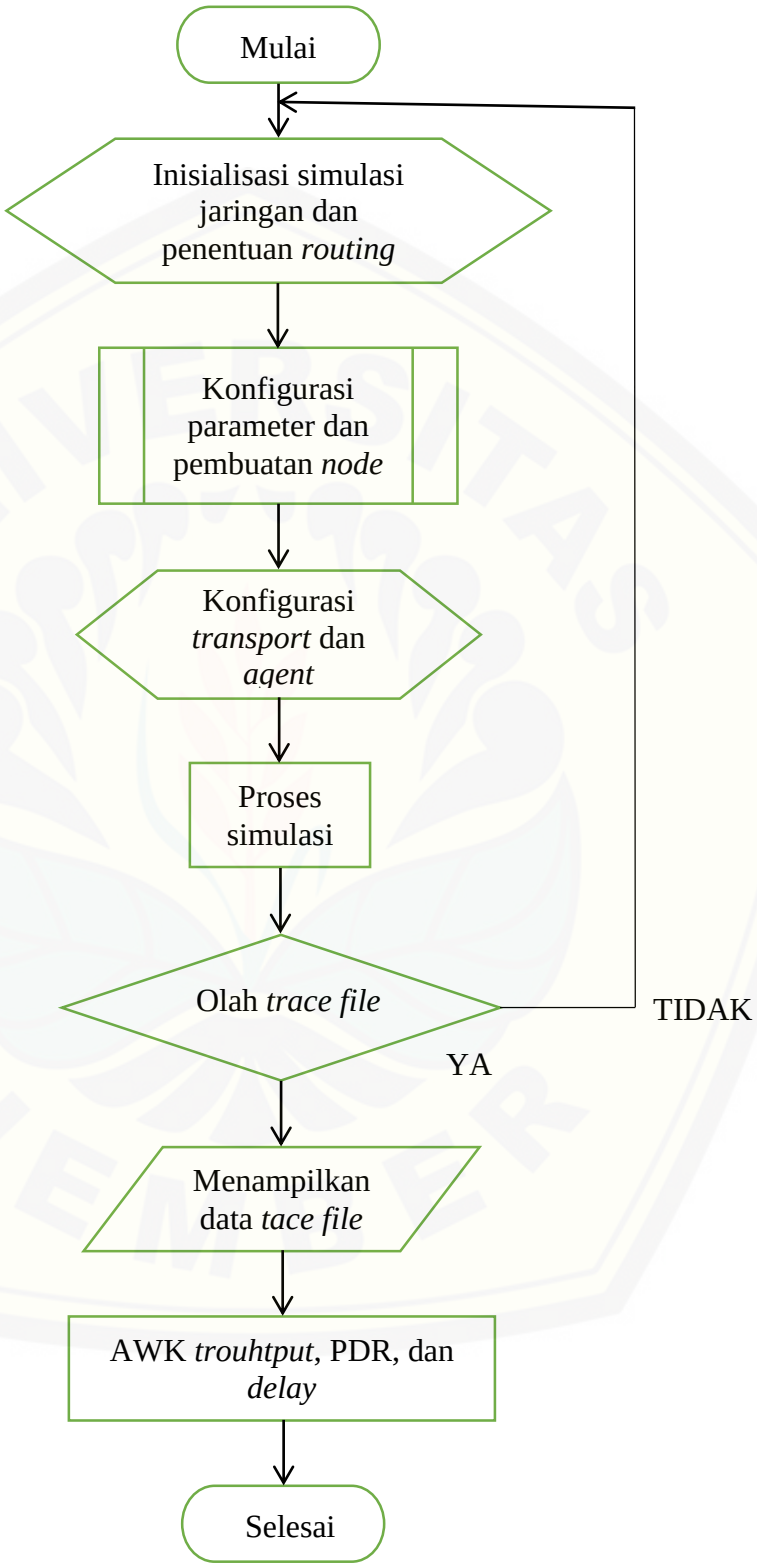
g. Analisis hasil kinerja routing OLSR dan AODV dengan skenario penambahan jumlah dan kecepatan *node.*

   Jadi setelah data parameter yang digunakan telah didapatkan, kemudian akan dilakukan analisis kinerja routing OLSR dan AODV dengan skenario penambahan jumlah dan kecepatan *node.*

h. Kesimpulan

   Setelah menganalisis data tiap parameter, akan ditarik kesimpulan manakah kinerja routing yang paling baik dengan skenario penambahan jumlah dan kecepatan *node.*

3.4.2 Diagram Alir Simulasi NS-2 Saat AWK



Gambar 3.2 Diagram Alir Simulasi pada NS-2 Saat AWK

3.5 Skenario Penelitian

Pada penelitian kali ini akan dilakukan skenario untuk mengetahui kinerja routing protokol OLSR dan AODV. Skenario yang dilakukan adalah sebagai beriku:

3.5.1 Skenario 1 Perubahan Jumlah *Node*

Dalam skenario ini dicoba untuk melihat bagaimana perubahan parameter berdasarkan jumlah *node* yang digunakan yaitu 6, 12, dan 18 *node* dengan kecepatan perpindahan *node* sebesar 10 m/s. Berikut tabel skenario 1:

Tabel 3.3 skenario 1 parameter berdasarkan perubahan jumlah node

Skenario	Perubahan jumlah *node*	Kecepatan (m/s)
Skenario 1	6 12 18	10

3.5.2 Skenario 2 Perubahan kecepata Pergerakan *Node*

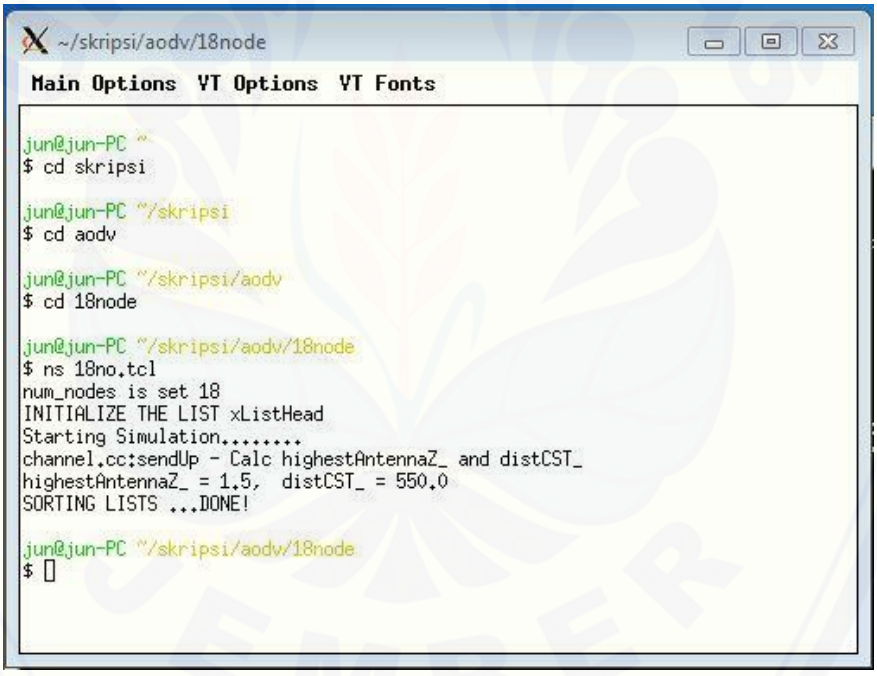
Dalam skenario ini dicoba untuk melihat bagaimana perubahan parameter berdasarkan pada kecepatan yang terdiri dari 10 m/s, 20 m/s, 30 m/s. Berikut tabel skenario 2 :

Tabel 3.4 skenario 2 parameter berdasarkan perubahan kecepatan node

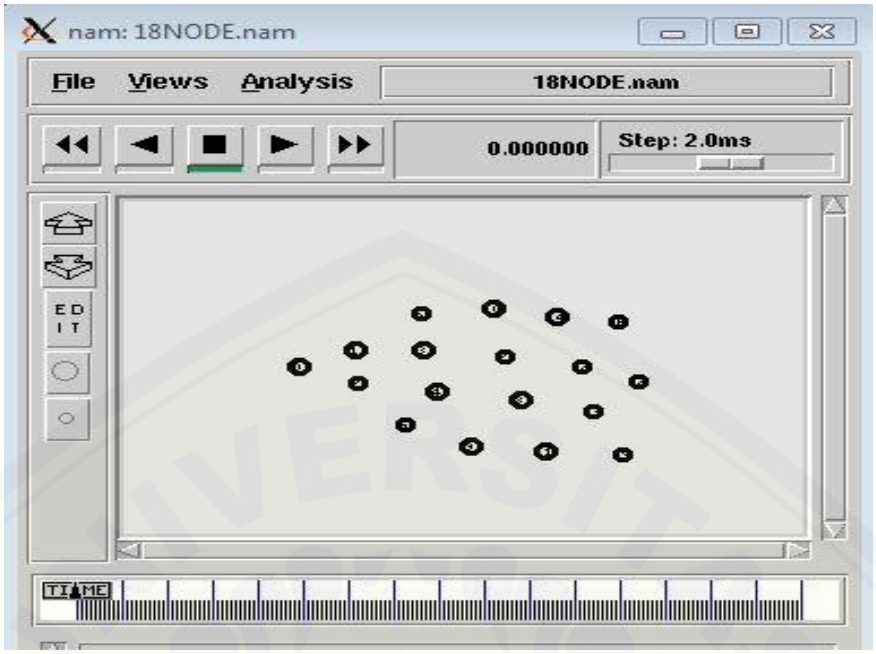
Skenario	Perubahan kecepatan *node* (m/s)	Jumlah *node*
Skenario 2	10 20 30	12

3.6 Proses menampilkan hasil AWK

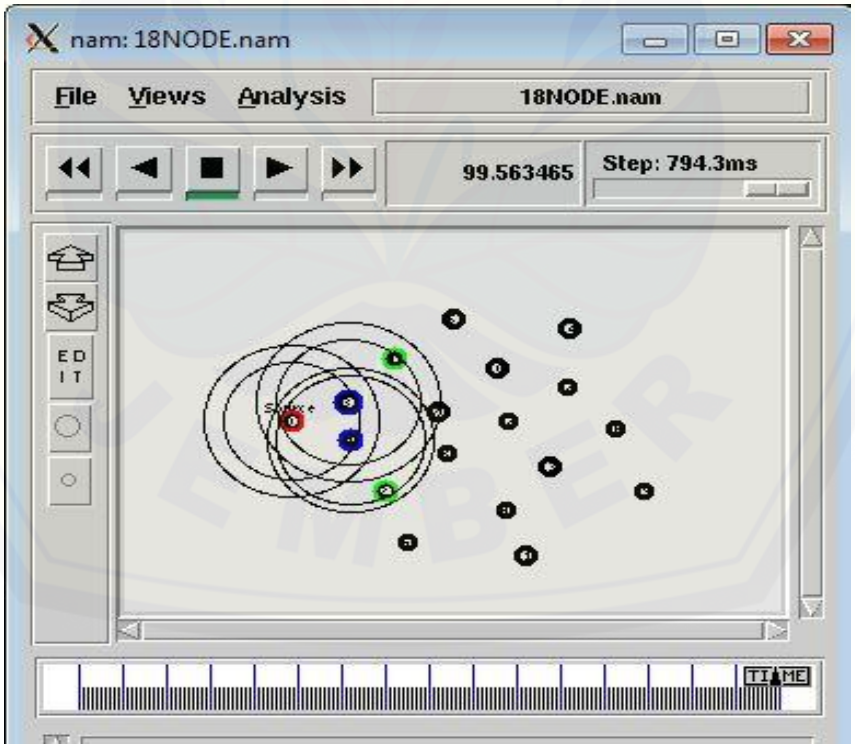
Simulasi pada aplikasi *network simulator* 2 akan menghasilkan keluaran berupa *file namtrace* yang berekstensi ".nam" dan *file trace* yang berekstensi ".tr", *file namtrace* digunakan untuk menginterpretasi hasil *output* secara grafis dan interaktif, digunakan sebuah *tool* seperti NAM (*Network Animator*), dan *file trace* mencatat semua kejadian selama simulasi terjadi dengan mengelolah *file trace* yang ada pada parameter kineja jaringan akan diperoleh dari hasil simulasi. Berikut merupakan tampilan perintah awal untuk menjalankan simulasi, *file nametrace* dan *file trace* yang diperoleh dari hasil simulasi pada skenario pertama dengan *routing protocol* AODV.



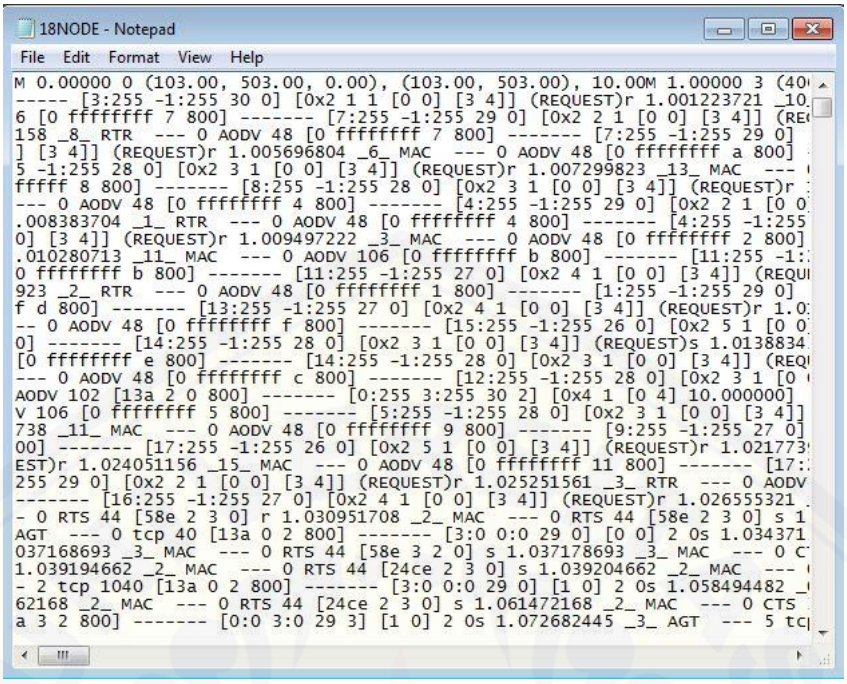
Gambar 3.3 Perintah untuk menjalankan simulasi untuk skenario 1 AODV



Gambar 3.4 Awal file *nametrace* skenario 1 untuk 18 *node* pada *routing* protokol AODV



Gambar 3.5 Akhir file *nametrace* skenario 1 untuk 18 *node* pada *routing* protokol AODV



Gambar 3.6 *File trace* skenario 1 untuk 18 *node* pada *routing* protokol AODV

Dari hasil *file trace* pada Gambar 3.6 dapat diperoleh hasil keseluruhan untuk parameter pengujian yang di antaranya PDR, *troughput*, dan *delay*. Pada skenario satu maupun skenario dua. Kemudian untuk mendapatkan nilai dari setiap parameter harus difilter terlebih dahulu dengan bantuan dari *file* awk, setelah mendapatkan *file* awk tersebut dapat diketikan pada cygwin dengan mengetikkan perintah untuk PDR perintah yang harus diketikkan yaitu:

Awk –f pdr.awk filename.tr > filename.tr

Perintah diatas akan otomatis memisahkan hasil PDR, sedangkan untuk *Throughput* perintah yang harus diketikkan yaitu:

Awk –f tr.awk filename.tr > filename.tr

Perintah diatas akan otomatis memisahkan hasil Throughput, dan untuk *delay* perintah yang harus diketikkan yaitu:

Awk –f delay.awk filename.tr > filename.tr

BAB 5. PENUTUP

5.1 Kesimpulan

Dari hasil penelitian yang telah dilakukan dan di analisis pada halaman sebelumnya maka dapat disimpulkan bahwa :

1. Pada skenario pertama dengan kondisi perubahan jumlah *node*, Pada *routing* protokol OLSR memperoleh hasil *packet delivery ratio* yang lebih besar daripada *routing* protokol AODV yaitu sebesar 99,33%; 99,41%; 99,11% dengan jumlah *node* sebanyak 6, 12, dan 18 *node*.
2. Nilai *throughput* pada skenario perubahan *node* dan perubahan kecepatan *node* memiliki hasil nilai yang besar. Semakin besar nilai *throughput pada* routing protocol menunjukkan semakin bagus pula kemampuan jaringan dalam mentransmisikan file. Hasil *throughput* yaitu sebesar 626,13kbps, 722,5kbps, 674,08kbps, dan 759,89kbps, 818,72kbps, 793,60kbps.

5.2 Saran

Dari hasil pengujian yang telah dilakukan, peneliti mempunyai beberapa saran agar penelitian ini dapat dikembangkan.

1. Membandingkan *routing protokol* yang agar dapat melihat performansi jaringan yang lebih jauh lagi.
2. Dapat menggunakan parameter yang lainnya seperti *jitter*, *paket loss*, dan lain sebagainya agar dapat melihat kualitas jaringan yang lebih jauh lagi..
3. Dapat dilakukan penelitian dalam hal pergerakan *node* secara random *node*.

DAFTAR PUSTAKA

Nutrihadi, Royyana Muslim. 2016. Studi Kerja VANET *Scenario Generators:*SUMO dan VanetMobisim untuk Implementasi *Routing Protocol* AODV menggunakan Network Simulator 2 (NS-2). Institut Teknologi Sepuluh November. Surabaya.

Singh, Kapang Lego. 2011. *Simulation Based Analysis of Adhoc Routing Protocol in Urban and Highway Scenario of VANET.* Central Institute of Technology, Kokrajhar, India.

Rozaki A. F. 2015. *Analisis Perbandingan Performansi Jaringan Wireless Sensor Network Saat Kondisi Node Diam Dengan Node Bergerak.*

Alfinanto Abednega, *Bambang Soelistijanto, Analisis Perbandingan Unjuk Kerja Protokol Routing Reaktif (DYMO) terhadap Routing Reaktif (AODV) pada Jaringan MANET.*

Endah Maya megawati. 2015. *Analisis Perbandingan Routing Protokol OLSR (Proaktif) dan AODV (Reaktif) pada MANET*.

Sharma. 2013. *Implementation of Dedicated Short Range Communication (DSRC) IEEE 802.11p in NS2 and Its Perfomance Analysis Over IEEE 802.11*. International Jjournal of Advanc Research, IJOAR.org

Arifin, M. Zen, dkk. 2011. *Analisis Performansi Routing AODV pada Jaringan VANet.*

Abduh I, Dahlia. N, dkk. 2017. *Perancangan Sistem Peringatan Antar Kendaraan Untuk Peningkatan Keselamatan Berkendara di Jalan*. Teknik Elektro. Politeknik Negeri Ujung Pandang.

Alamsyah, Eko. S, dkk.2018. *Analisis Kinerja Protokol Routing Reaktif Dan Proaktif Pada MANET Menggunakan NS2*.

Muhammad, Doan Perdana, dkk. 2016. Analisis Perbandingan Kinerja Protokol Routing DSDV dan OLSR untuk Perubahan Keccepatan mobilitas pada Standar IEEE 802.11ah. Fakultas Teknik Elektro. Universitas Jember.

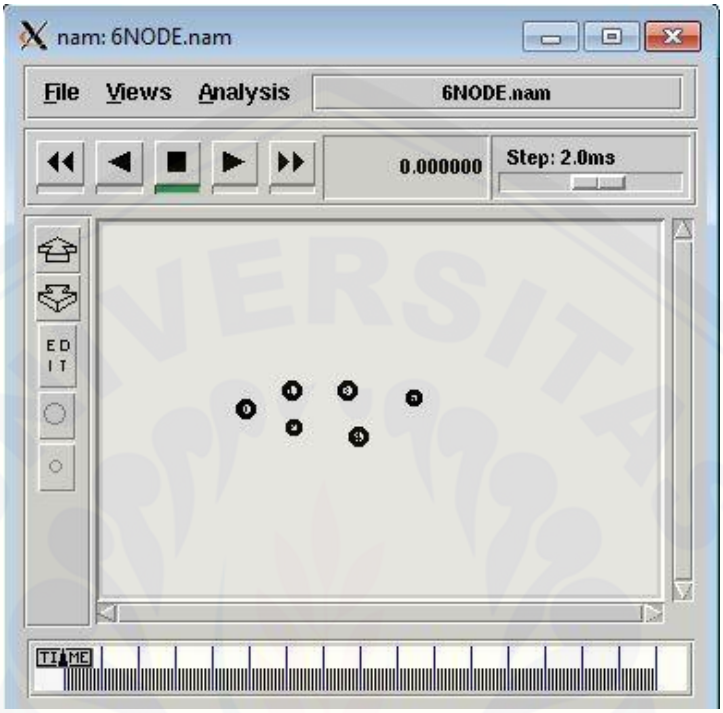
Hadi , Michele C. 2010. *Highway Mobility and Vehicular Ad-Hoc Networks in NS-3.* Old Dominion Unversity, Norvolk, USA.

Wei, Mei-Wen. *A Comparison of 802.11a and 802.11p for V-to-I communication : a measurement study.*

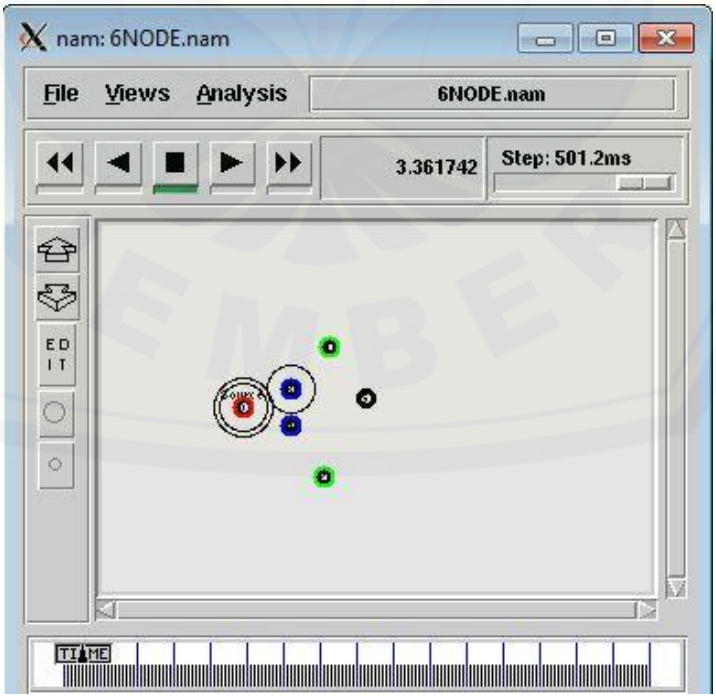
Hartenstein H, Kenneth L (Ed.). 2009. *Vehicular Ad Hoc Network* (VANET). New Jersey : Wiley.

LAMPIRAN

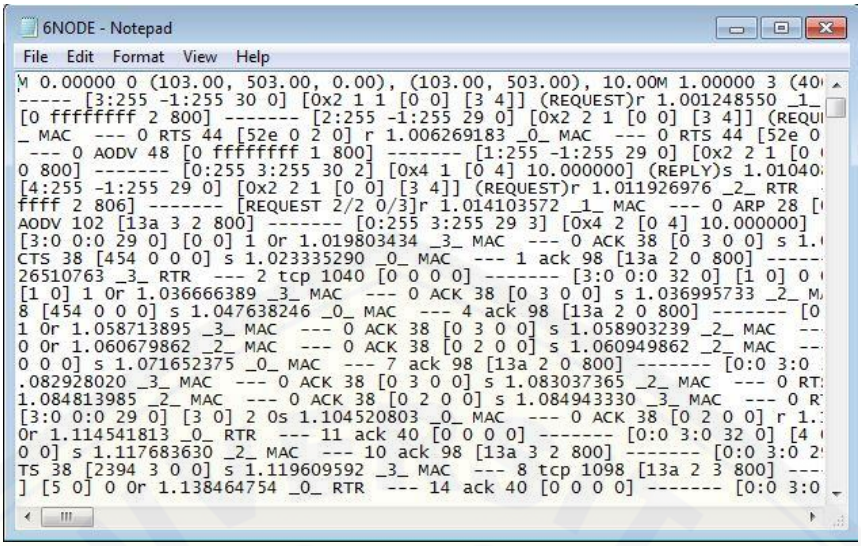
1. Lampiran Simulasi VANET Pada Beberapa Percobaan



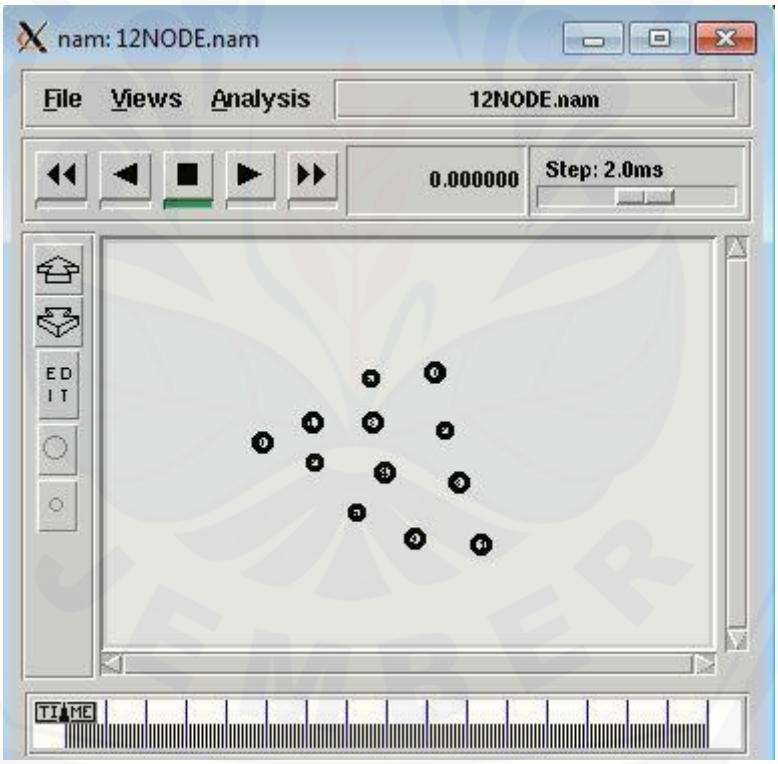
Gambar 1. Posisi *Node* Pada Jumlah 6 *Node*



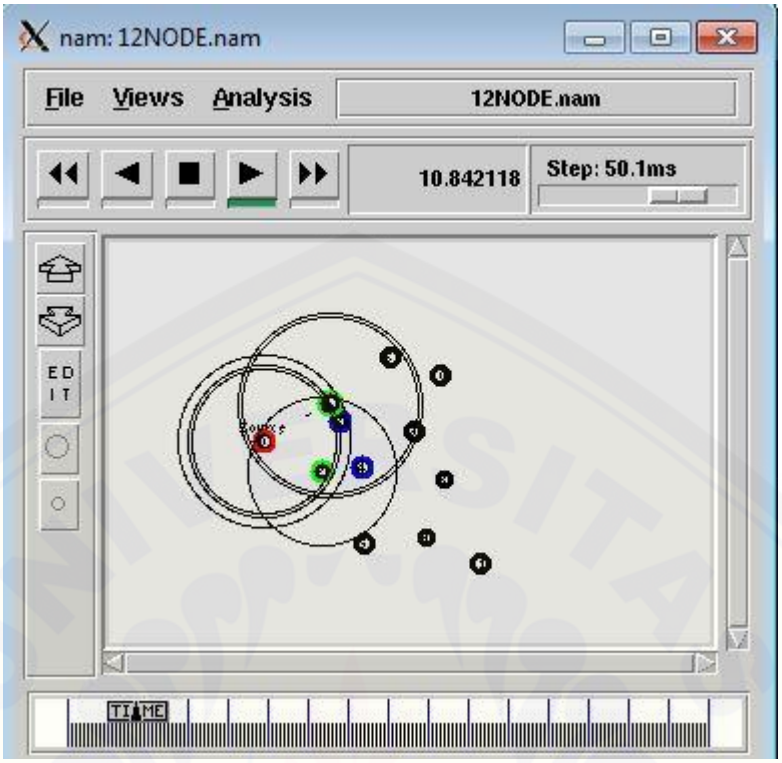
Gambar 2. Proses Pengiriman Paket Data Pada Jumlah 6 *Node*



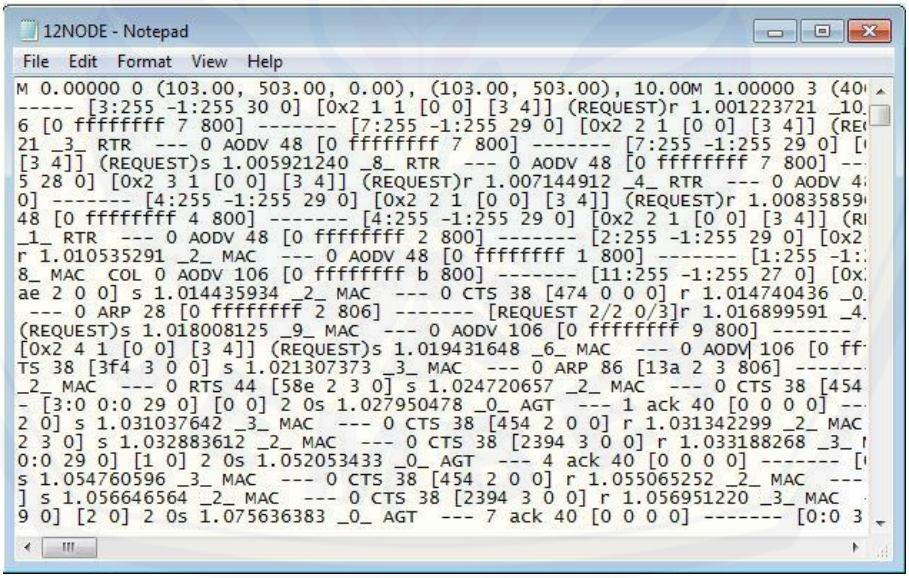
Gambar 3. Hasil *file trace Node* Pada Jumlah 6 *Node*



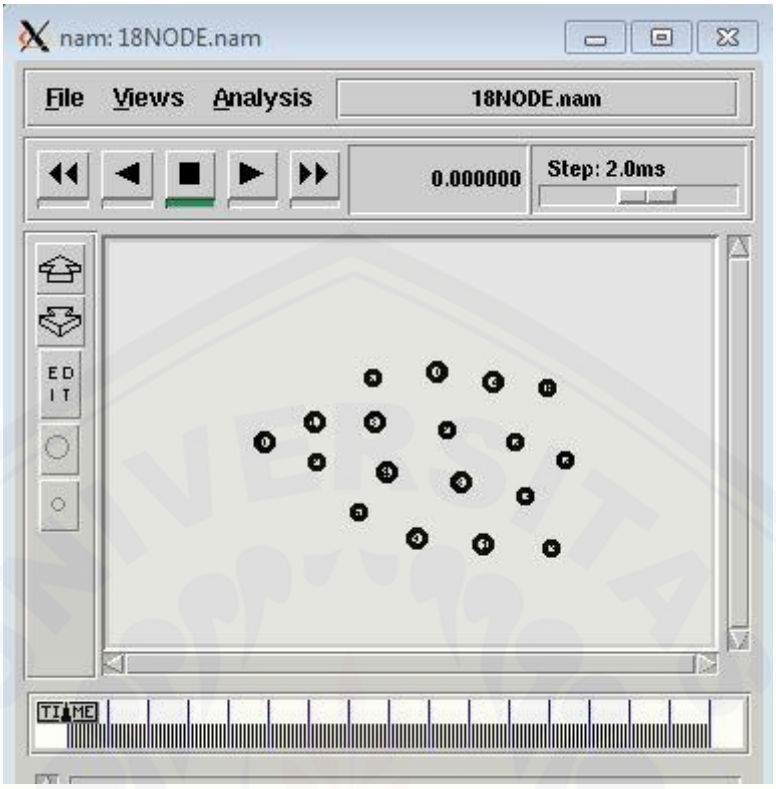
Gambar 4. Posisi *Node* Pada Jumlah 12 *Node*



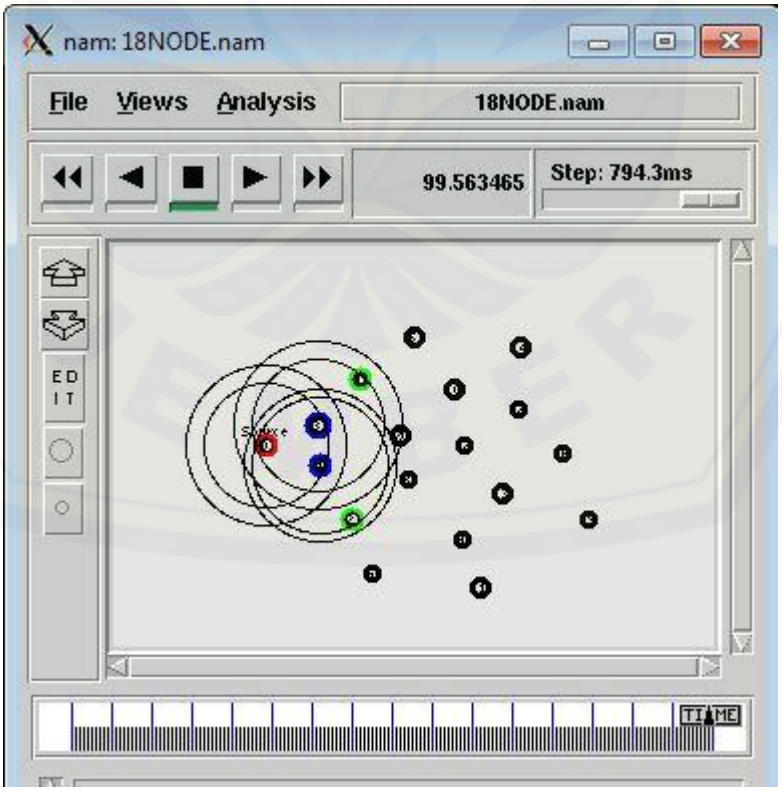
Gambar 5. Proses Pengiriman Paket Data Pada Jumlah 12 *Node*



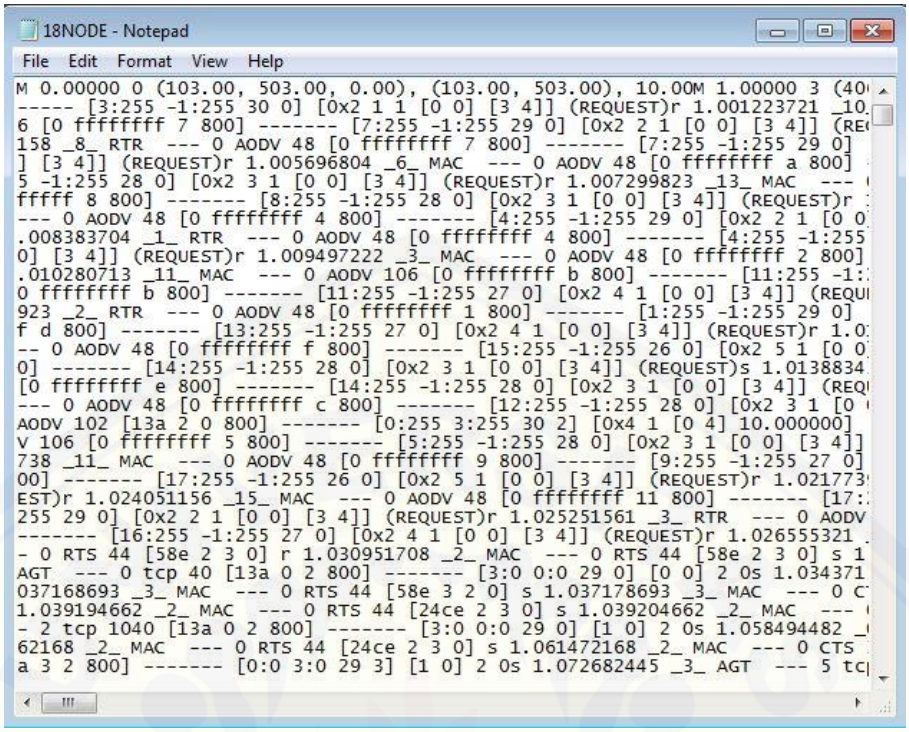
Gambar 6. Hasil *file trace Node* Pada Jumlah 12 *Node*



Gambar 7. Posisi *Node* Pada Jumlah 18 *Node*



Gambar 8. Proses Pengiriman Paket Data Pada Jumlah 18 *Node*



Gambar 9. Hasil *file trace Node* Pada Jumlah 18 *Node*

2. List Program tcl Pada Perubahan Jumlah Node

```
# Define options
#
=====================================================
=
set val(chan) Channel/WirelessChannel ;# Channel Type
set val(prop) Propagation/TwoRayGround ;# radio-propagation model
set val(netif) Phy/WirelessPhy ;# network interface type
set val(mac) Mac/802_11 ;# MAC type
set val(ifq) Queue/DropTail/PriQueue ;# interface queue type
set val(ll) LL ;# link layer type
set val(ant) Antenna/OmniAntenna ;# antenna model
set val(ifqlen) 50 ;# max packet in ifq
set val(nn) 18 ;# number of mobilenodes
set val(rp) AODV ;# routing protocol
set val(x) 1000
set val(y) 1000
#
=====================================================
=
# Main Program
#
=====================================================
=
# inisialisasi variabel global
set ns_ [new Simulator]
#create the nam and trace file:
set tracefd [open 18NODE.tr w]
$ns_ trace-all $tracefd
set namtrace [open 18NODE.nam w]
$ns_ namtrace-all-wireless $namtrace $val(x) $val(y)
```

```
set topo [new Topography]
$topo load_flatgrid $val(x) $val(y)
create-god $val(nn)
set chan_1_ [new $val(chan)]

# Defining Node Configuration
$ns_ node-config -adhocRouting $val(rp) \
-llType $val(ll) \
-macType $val(mac) \
-ifqType $val(ifq) \
-ifqLen $val(ifqlen) \
-antType $val(ant) \
-propType $val(prop) \
-phyType $val(netif) \
-topoInstance $topo \
-agentTrace ON \
-routerTrace ON \
-macTrace ON \
-movementTrace ON \
-channel $chan_1_

# create nodes
for {set i 0} {$i < $val(nn)} {incr i} {
set node_($i) [$ns_ node]
$node_($i) random-motion 0;
}
#Inisialisasi (X,Y, dan Z=0)
#Create 18 nodes
$node_(0) set X_ 103
$node_(0) set Y_ 503
$node_(0) set Z_ 0.0
```

```
$ns_ initial_node_pos $node_(0) 40
$node_(1) set X_ 241
$node_(1) set Y_ 556
$node_(1) set Z_ 0.0
$ns_ initial_node_pos $node_(1) 40
$node_(2) set X_ 243
$node_(2) set Y_ 447
$node_(2) set Z_ 0.0
$ns_ initial_node_pos $node_(2) 40
$node_(3) set X_ 406
$node_(3) set Y_ 558
$node_(3) set Z_ 0.0
$ns_ initial_node_pos $node_(3) 40
$node_(4) set X_ 437
$node_(4) set Y_ 422
$node_(4) set Z_ 0.0
$ns_ initial_node_pos $node_(4) 40
$node_(5) set X_ 358
$node_(5) set Y_ 308
$node_(5) set Z_ 0.0
$ns_ initial_node_pos $node_(5) 40
$node_(6) set X_ 396
$node_(6) set Y_ 681
$node_(6) set Z_ 0.0
$ns_ initial_node_pos $node_(6) 40
$node_(7) set X_ 600
$node_(7) set Y_ 534
$node_(7) set Z_ 0.0
$ns_ initial_node_pos $node_(7) 40
$node_(8) set X_ 640
$node_(8) set Y_ 393
```

```
$node_(8) set Z_ 0.0
$ns_ initial_node_pos $node_(8) 40
$node_(9) set X_ 520
$node_(9) set Y_ 241
$node_(9) set Z_ 0.0
$ns_ initial_node_pos $node_(9) 40
$node_(10) set X_ 575
$node_(10) set Y_ 693
$node_(10) set Z_ 0.0
$ns_ initial_node_pos $node_(10) 40
$node_(11) set X_ 703
$node_(11) set Y_ 223
$node_(11) set Z_ 0.0
$ns_ initial_node_pos $node_(11) 40
$node_(12) set X_ 787
$node_(12) set Y_ 505
$node_(12) set Z_ 0.0
$ns_ initial_node_pos $node_(12) 40
$node_(13) set X_ 814
$node_(13) set Y_ 355
$node_(13) set Z_ 0.0
$ns_ initial_node_pos $node_(13) 40
$node_(14) set X_ 729
$node_(14) set Y_ 666
$node_(14) set Z_ 0.0
$ns_ initial_node_pos $node_(14) 40
$node_(15) set X_ 885
$node_(15) set Y_ 209
$node_(15) set Z_ 0.0
$ns_ initial_node_pos $node_(15) 40
$node_(16) set X_ 875
```

```
$node_(16) set Y_ 652
$node_(16) set Z_ 0.0
$ns_ initial_node_pos $node_(16) 40
$node_(17) set X_ 925
$node_(17) set Y_ 455
$node_(17) set Z_ 0.0
$ns_ initial_node_pos $node_(17) 40

## Giving Mobility to nodes
$ns_ at 0.0 "$node_(0) setdest 103.0 503.0 10.0"
$ns_ at 4.25 "$node_(1) setdest 359.0 686.0 10.0"
$ns_ at 7.75 "$node_(2) setdest 342.0 294.0 10.0"
$ns_ at 1.0 "$node_(3) setdest 241.0 558.0 10.0"
$ns_ at 4.20 "$node_(4) setdest 243.0 447.0 10.0"
$ns_ at 2.25 "$node_(5) setdest 392.0 146.0 10.0"
$ns_ at 3.25 "$node_(6) setdest 517.0 807.0 10.0"
$ns_ at 2.25 "$node_(7) setdest 465.0 531.0 10.0"
$ns_ at 6.25 "$node_(8) setdest 485.0 409.0 10.0"
$ns_ at 8.25 "$node_(9) setdest 643.0 243.0 10.0"
$ns_ at 9.25 "$node_(10) setdest 621.0 657.0 10.0"
$ns_ at 5.25 "$node_(11) setdest 692.0 108.0 10.0"
$ns_ at 4.25 "$node_(12) setdest 639.0 505.0 10.0"
$ns_ at 6.25 "$node_(13) setdest 746.0 373.0 10.0"
$ns_ at 7.25 "$node_(14) setdest 804.0 777.0 10.0"
$ns_ at 8.25 "$node_(15) setdest 989.0 301.0 10.0"
$ns_ at 9.25 "$node_(16) setdest 916.0 477.0 10.0"
$ns_ at 10.25 "$node_(17) setdest 789.0 608.0 10.0"

#defining heads
$ns_ at 0.0 "$node_(0) label Source"
$ns_ at 0.0 "$node_(0) shape hexagon "
```

```
$ns_ at 0.0 "$node_(0) add-mark N0 red circle"
$ns_ at 0.0 "$node_(1) add-mark N0 green circle"
$ns_ at 0.0 "$node_(2) add-mark N0 green circle"
$ns_ at 0.0 "$node_(3) add-mark N0 blue circle"
$ns_ at 0.0 "$node_(4) add-mark N0 blue circle"

#Set a TCP connection between node 3 and node 0
set tcp [new Agent/TCP]
$tcp set class_ 2
set sink [new Agent/TCPSink]
$ns_ attach-agent $node_(3) $tcp
$ns_ attach-agent $node_(0) $sink
$ns_ connect $tcp $sink
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ns_ at 1.0 "$ftp start"

#Set a TCP connection between node 0 and node 1
set tcp [new Agent/TCP]
$tcp set class_ 2
set sink [new Agent/TCPSink]
$ns_ attach-agent $node_(0) $tcp
$ns_ attach-agent $node_(1) $sink
$ns_ connect $tcp $sink
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ns_ at 2.0 "$ftp start"

#Set a TCP connection between node 4 and node 0
set tcp [new Agent/TCP]
$tcp set class_ 2
```

```
set sink [new Agent/TCPSink]
$ns_ attach-agent $node_(4) $tcp
$ns_ attach-agent $node_(0) $sink
$ns_ connect $tcp $sink
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ns_ at 3.0 "$ftp start"

proc stop {} {
                global ns_ tracefd
                $ns_ flush-trace
                close $tracefd
                exec nam 18NODE.nam &
                exit 0
            }
    puts "Starting Simulation....."
    $ns_ at 100 "stop"
$ns_ run
```

3. List Program tcl Pada Perubahan kecepatan Node

```
# Define options
#=================================================
set val(chan) Channel/WirelessChannel ;# Channel Type
set val(prop) Propagation/TwoRayGround ;# radio-propagation model
set val(netif) Phy/WirelessPhy ;# network interface type
set val(mac) Mac/802_11 ;# MAC type
set val(ifq) Queue/DropTail/PriQueue ;# interface queue type
set val(ll) LL ;# link layer type
set val(ant) Antenna/OmniAntenna ;# antenna model
set val(ifqlen) 50 ;# max packet in ifq
set val(nn) 12 ;# number of mobilenodes
```

```
set val(rp) AODV ;# routing protocol
set val(x) 1000
set val(y) 1000
#===================================================
# Main Program
# ===================================================
# inisialisasi variabel global
set ns_ [new Simulator]
#create the nam and trace file:
set tracefd [open 12NODE.tr w]
$ns_ trace-all $tracefd
set namtrace [open 12NODE.nam w]
$ns_ namtrace-all-wireless $namtrace $val(x) $val(y)
set topo [new Topography]
$topo load_flatgrid $val(x) $val(y)
create-god $val(nn)
set chan_1_ [new $val(chan)]
# Defining Node Configuration
$ns_ node-config -adhocRouting $val(rp) \
-llType $val(ll) \
-macType $val(mac) \
-ifqType $val(ifq) \
-ifqLen $val(ifqlen) \
-antType $val(ant) \
-propType $val(prop) \
-phyType $val(netif) \
-topoInstance $topo \
-agentTrace ON \
-routerTrace ON \
-macTrace ON \
-movementTrace ON \
```

```
-channel $chan_1_
# create nodes
for {set i 0} {$i < $val(nn)} {incr i} {
set node_($i) [$ns_ node]
$node_($i) random-motion 0;
}
#Inisialisasi (X,Y, dan Z=0)
#Create 18 nodes
$node_(0) set X_ 103
$node_(0) set Y_ 503
$node_(0) set Z_ 0.0
$ns_ initial_node_pos $node_(0) 40
$node_(1) set X_ 241
$node_(1) set Y_ 556
$node_(1) set Z_ 0.0
$ns_ initial_node_pos $node_(1) 40
$node_(2) set X_ 243
$node_(2) set Y_ 447
$node_(2) set Z_ 0.0
$ns_ initial_node_pos $node_(2) 40
$node_(3) set X_ 406
$node_(3) set Y_ 558
$node_(3) set Z_ 0.0
$ns_ initial_node_pos $node_(3) 40
$node_(4) set X_ 437
$node_(4) set Y_ 422
$node_(4) set Z_ 0.0
$ns_ initial_node_pos $node_(4) 40
$node_(5) set X_ 358
$node_(5) set Y_ 308
$node_(5) set Z_ 0.0
```

```
$ns_ initial_node_pos $node_(5) 40
$node_(6) set X_ 396
$node_(6) set Y_ 681
$node_(6) set Z_ 0.0
$ns_ initial_node_pos $node_(6) 40
$node_(7) set X_ 600
$node_(7) set Y_ 534
$node_(7) set Z_ 0.0
$ns_ initial_node_pos $node_(7) 40
$node_(8) set X_ 640
$node_(8) set Y_ 393
$node_(8) set Z_ 0.0
$ns_ initial_node_pos $node_(8) 40
$node_(9) set X_ 520
$node_(9) set Y_ 241
$node_(9) set Z_ 0.0
$ns_ initial_node_pos $node_(9) 40
$node_(10) set X_ 575
$node_(10) set Y_ 693
$node_(10) set Z_ 0.0
$ns_ initial_node_pos $node_(10) 40
$node_(11) set X_ 703
$node_(11) set Y_ 223
$node_(11) set Z_ 0.0
$ns_ initial_node_pos $node_(11) 40

## Giving Mobility to nodes
$ns_ at 0.0 "$node_(0) setdest 103.0 503.0 30.0"
$ns_ at 4.25 "$node_(1) setdest 359.0 686.0 30.0"
$ns_ at 7.75 "$node_(2) setdest 342.0 294.0 30.0"
$ns_ at 1.0 "$node_(3) setdest 241.0 558.0 30.0"
```

```
$ns_ at 4.20 "$node_(4) setdest 243.0 447.0 30.0"
$ns_ at 2.25 "$node_(5) setdest 392.0 146.0 30.0"
$ns_ at 3.25 "$node_(6) setdest 517.0 807.0 30.0"
$ns_ at 2.25 "$node_(7) setdest 465.0 531.0 30.0"
$ns_ at 6.25 "$node_(8) setdest 485.0 409.0 30.0"
$ns_ at 8.25 "$node_(9) setdest 643.0 243.0 30.0"
$ns_ at 9.25 "$node_(10) setdest 621.0 657.0 30.0"
$ns_ at 5.25 "$node_(11) setdest 692.0 108.0 30.0"

#defining heads
$ns_ at 0.0 "$node_(0) label Source"
$ns_ at 0.0 "$node_(0) shape hexagon "
$ns_ at 0.0 "$node_(0) add-mark N0 red circle"
$ns_ at 0.0 "$node_(1) add-mark N0 green circle"
$ns_ at 0.0 "$node_(2) add-mark N0 green circle"
$ns_ at 0.0 "$node_(3) add-mark N0 blue circle"
$ns_ at 0.0 "$node_(4) add-mark N0 blue circle"
#Set a TCP connection between node 3 and node 0
set tcp [new Agent/TCP]
$tcp set class_ 2
set sink [new Agent/TCPSink]
$ns_ attach-agent $node_(3) $tcp
$ns_ attach-agent $node_(0) $sink
$ns_ connect $tcp $sink
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ns_ at 1.0 "$ftp start"

#Set a TCP connection between node 0 and node 1
set tcp [new Agent/TCP]
$tcp set class_ 2
```

```
set sink [new Agent/TCPSink]
$ns_ attach-agent $node_(0) $tcp
$ns_ attach-agent $node_(1) $sink
$ns_ connect $tcp $sink
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ns_ at 2.0 "$ftp start"

#Set a TCP connection between node 4 and node 0
set tcp [new Agent/TCP]
$tcp set class_ 2
set sink [new Agent/TCPSink]
$ns_ attach-agent $node_(4) $tcp
$ns_ attach-agent $node_(0) $sink
$ns_ connect $tcp $sink
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ns_ at 3.0 "$ftp start"

proc stop {} {
        global ns_ tracefd
        $ns_ flush-trace
        close $tracefd
        exec nam 12NODE.nam &
        exit 0
    }
puts "Starting Simulation....."
$ns_ at 100 "stop"
$ns_ run
```

4. List Program AWK *Packet Delivery Ratio* (PDR)

```
BEGIN {
        sendLine = 0;
        recvLine = 0;
        fowardLine = 0;
}

$0 ~/^s.* AGT/ {
        sendLine ++ ;
}

$0 ~/^r.* AGT/ {
        recvLine ++ ;
}

$0 ~/^f.* RTR/ {
        fowardLine ++ ;
}

END {
        printf "tcp s:%d r:%d, r/s Ratio:%.4f, f:%d \n", sendLine, recvLine, (recvLine/sendLine),fowardLine;
}
```

5. List Program AWK *Throughput*

```
BEGIN {
      recvdSize = 0
      startTime = 0
      stopTime = 120
 }
```

```
{
        event = $1
        time = $2
        node_id = $3
        pkt_size = $12
        level = $4

# Store start time
if (level == "AGT" && event == "s" && pkt_size >= 800) {
  if (time < startTime) {
        startTime = time
        }
    }

# Update total received packets' size and store packets arrival time
if (level == "AGT" && event == "r" && pkt_size >= 800) {
    if (time > stopTime) {
        stopTime = time
        }
    # Rip off the header
    hdr_size = pkt_size % 800
    pkt_size -= hdr_size
    # Store received packet's size
    recvdSize += pkt_size
    }
}
END {
    printf("Average Throughput[kbps] = %.2f\t\t
StartTime=%.2f\tStopTime=%.2f\n",(recvdSize/(stopTime-
startTime))*(8/1000),startTime,stopTime)
}
```

6. List Program AWK *Delay*

```
BEGIN{
sumdelay=0;
countdelay=0;
    }
    {
     sumdelay +=$10;
     countdelay++;
    }
END{
printf("Average   Delay[s%]   =   %f   \n",sumdelay/countdelay);
}
```