PENGARUH SERANGAN *RUSHING* TERHADAP *PROTOCOL AD HOC ON DEMAND DISTANCE VECTOR* (AODV) PADA JARINGAN *MOBILE AD HOC NETWORKS* (MANET)

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Eldyto Puspa Laksana
145150200111107

PROGRAM STUDI INFORMATIKA
JURUSAN INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2019

PENGESAHAN

PENGARUH SERANGAN *RUSHING* TERHADAP *PROTOCOL AD HOC ON DEMAND DISTANCE VECTOR* (AODV) PADA JARINGAN *MOBILE AD HOC NETWORKS* (MANET)

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :

Eldyto Puspa Laksana
NIM: 145150200111107

Skripsi ini telah diuji dan dinyatakan lulus pada
3 Desember 2019
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing 1	Dosen Pembimbing 2
Reza Andria Siregar, S.T., M.Kom.	Achmad Basuki, S.T., M.MG., Ph.D.
NIK: 19790621 200604 1 003	NIK: 19741118 200312 1 002

Mengetahui
Ketua Jurusan **Teknik Informatika**

Tri Astoto Kurniawan, S.T., M.T, Ph.D.
NIK: 19790621 200604 1 003

PERNYATAAN ORISINALITAS

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

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

Malang, 3 Desember 2019

Eldyto Puspa Laksana

NIM: 14515020011107

PRAKATA

Puji Syukur kehadirat Allah Bapa Yang Maha Kuasa yang mengaruniakan anak-Nya yang tunggal Tuhan Yesus Kristus yang telah memberikan berkat karunia-Nya sehingga laporan skripsi yang berjudul "Pengaruh Serangan *Rushing* Terhadap *Protocol Ad Hoc On Demand Distance Vector* (Aodv) Pada Jaringan *Mobile Ad Hoc Networks* (Manet)" ini dapat terselesaikan. Penulis ingin menyampaikan rasa hormat dan terimakasih kepada :

1. Reza Andria Siregar, S.T., M.Kom. selaku dosen pembimbing 1 yang telah meluangkan waktu untuk membimbing dan memberikan masukan ilmu, arahan serta nasihat dan masukan kepada penulis sehingga dapat menyelesaikan skripsi ini dengan baik.
2. Achmad Basuki, S.T., M.MG., Ph.D. selaku dosen pembimbing 2 yang telah meluangkan waktu untuk membimbing dan memberikan masukan ilmu, arahan serta nasihat dan masukan kepada penulis sehingga dapat menyelesaikan skripsi ini dengan baik.
3. Bayu Priyambadha, S.Kom., M.Kom. selaku dosen pembimbing akademik.
4. Tri Astoto Kurniawan, S.T., M.T, Ph.D. selaku Ketua Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya.
5. Wayan Firdaus Mahmudy, S.Si, M.T., P.hD. selaku Dekan Fakultas Ilmu Komputer Universitas Brawijaya.
6. Bapak Budyo Leksono dan Ibu Arie Utami Yanuarti selaku orang tua penulis atas segala doa, dukungan, dan kepercayaan dalam bentuk apapun dalam menyelesaikan skripsi ini
7. Semua pihak yang tidak dapat penulis sebutkan satu persatu. Untuk yang terlibat secara langsung maupun tidak langsung dalam penyelesaian skripsi ini.

Semoga Allah Bapa senantiasa memberikan berkat karunia-Nya untuk semua pihak yang telah membantu dan mendukung penyelesaian skripsi ini. Dengan segala kekurangan penulis mengharapkan untuk memberikan kritik dan saran sebagai acuan pada penelitian selanjutnya. Semoga skripsi ini dapat memberikan manfaat untuk semua pihak yang menggunakannya.

Malang, 3 Desember 2019

Eldyto Puspa Laksana
eldyto180395@gmail.com

ABSTRAK

Eldyto Puspa Laksana, Pengaruh Serangan *Rushing* Terhadap *Protocol Ad Hoc On Demand Distance Vector* (Aodv) Pada Jaringan *Mobile Ad Hoc Networks* (Manet).

Pembimbing: Reza Andria Siregar, S.T., M.Kom. dan Achmad Basuki, S.T, M.MG, Ph.D

Abstrak

Mobile Ad Hoc Networks (MANET) adalah salah satu jenis jaringan nirkabel yang ada saat ini, dimana *mobile node*s diasosiasikan dengan tidak terencana atau disebut *ad hoc.* Terdapat berbagai jenis protokol routing pada MANET dan salah satunya adalah *reactive routing.* Salah satu contoh algoritme *reactive routing* adalah Ad hoc On-Demand Distance Vector (AODV). Serangan *rushing* memanfaatkan RREQ pada *route discovery* untuk menjadi *node* penghubung antara *source* dan *destination* paket data pada jaringan. Jumlah *node* yang digunakan 30 *node*, 40 *node*, 50 *node*. Simulasi tersebut dilakukan menggunakan *Network Simulator* (NS-2.35). Serangan *rushing* memberikan dampak pada *routing* protokol AODV dengan menurunkan kinerjanya. Dengan dibuktikan dari nilai *packet delivery ratio, packet loss, throughput,* dan *end to end delay* yang turun kualitasnya. Penurunan nilai *packet delivery ratio* paling banyak terjadi pada implementasi 3 serangan *rushing* pada 30 *node* yaitu sebesar 20,35 %, untuk kenaikan nilai *packet loss* paling banyak pada implementasi 3 serangan *rushing* pada 30 *node* yaitu sebesar 20,35%, untuk penurunan nilai *throughput* paling banyak pada implementasi 3 serangan *rushing* pada 30 *node* yaitu sebesar 20,75 kbps, dan untuk kenaikan nilai *end to end delay* paling banyak pada implementasi 3 serangan *rushing* pada 40 *node* yaitu sebesar 770,8293 ms.

Kata kunci: *MANET, AODV, Rushing, NS-2.35*

ABSTRACT

Eldyto Puspa Laksana, Pengaruh Serangan *Rushing* Terhadap *Protocol Ad Hoc On Demand Distance Vector* (Aodv) Pada Jaringan *Mobile Ad Hoc Networks* (Manet).

Supervisors: Reza Andria Siregar, S.T., M.Kom. dan Achmad Basuki, S.T, M.MG, Ph.D

Mobile Ad Hoc Networks (MANET) is one type of wireless network that exists today, where mobile nodes are associated with unplanned or called ad hoc. There are various types of routing protocols on MANET and one of them is reactive routing. One example of reactive routing algorithms is Ad hoc On-Demand Distance Vector (AODV). Rush attacks use RREQ on route discovery to become a connecting node between the source and destination data packets on the network. The number of nodes used is 30 nodes, 40 nodes, 50 nodes. The simulation was carried out using Network Simulator (NS-2.35). Rush attacks have an impact on routing the AODV protocol by reducing its performance. It is proven by the value of the packet delivery ratio, packet loss, throughput, and end to end delay, which decreases in quality. The most decline in the value of the packet delivery ratio occurs in the implementation of 3 rushing attacks at 30 nodes, namely 20.35%, to increase the value of packet loss mostly in the implementation of 3 rushing attacks at 30 nodes, namely 20.35%, to decrease the most throughput value there are many on the implementation of 3 rushing attacks on 30 nodes, namely 20.75 kbps, and for increasing the end to end delay the most is the implementation of 3 rushing attacks on 40 nodes, namely 770,8293 ms.

Keywords : *MANET*, *AODV*, *Rushing*, *NS-2.35*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Mobile Ad Hoc Networks (MANET) adalah salah satu jenis jaringan nirkabel yang ada saat ini, dimana *mobile nodes* diasosiasikan dengan tidak terencana atau disebut *ad hoc.* MANET dapat melakukan *self-forming* dan *self-healing* dimana hal tersebut memungkinkan komunikasi antara *mobile nodes* tidak bergantung pada sumber daya yang terpusat atau infrastruktur tetap (Cisco. 2018). Kemampuan *self-forming* dan *self-healing* pada MANET dapat memberikan manfaat dalam memenuhi kebutuhan jaringan untuk pengguna atau platform yang memiliki mobilitas tinggi pada wilayah yang belum mempunyai infrastruktur jaringan. MANET dapat digunakan untuk pemulihan bencana, pertambangan, transportasi, dan pertahanan. MANET memungkinkan komunikasi tanpa adanya infrastruktur tetap. MANET dapat diterapkan sebagai salah satu alternatif infrastruktur jaringan dalam membantu komunikasi antar *node*. Terdapat berbagai jenis protokol routing pada MANET dan salah satunya adalah *reactive routing. Reactive routing* akan mengirimkan paket *Route Request* (RREQ) ke semua *node* yang ada di suatu jaringan untuk menemukan rute. Salah satu contoh algoritme *reactive routing* adalah Ad hoc On-Demand Distance Vector (AODV).

AODV merupakan *distance vector routing protocol* yang termasuk dalam kelompok reaktif *routing protocol*, yang me-*request* sebuah rute hanya pada saat dibutuhkan. AODV mempunyai ciri-ciri utama yaitu menjaga *timer-based state* setiap *node* sesuai dengan penggunaan tabel *routing*. AODV mempunyai *route discovery* dan *route maintenance*. *Route Discovery* berupa RREQ dan *Route Reply* (RREP). Sedangkan *Route Maintenance* berupa Data, *Route update* dan *Route Error* (RRER) (Sari, Syarif, & Budiardjo, 2010) .

Serangan-serangan yang ada saat ini mempunyai karakteristik yang berbeda, dari perbedaan karakteristik tersebut serangan dalam MANET dibedakan menjadi serangan lalu lintas data dan serangan lalu lintas *control*. Serangan lalu lintas data akan membuang atau menahan data melalui *node malicious* hal ini akan menurunkan kualitas layanan dan meningkatkan *end to end delay*, hingga dapat menghilangkan data yang penting, contohnya *blackhole, grayhole*. Pada serangan lalu lintas *control* akan mengganggu jaringan dengan memanipulasi *routing table* sehingga *node malicious* seolah-olah berada pada rute terpendek dan serangan lalu lintas control dapat mencuri data yang dikirimkan dari *source node* menuju *destination node,* contohnya *rushing, wormhole* (Bhattacharyya, 2011).

Serangan *rushing* memanfaatkan RREQ pada *route discovery* untuk menjadi *node* penghubung antara *source* dan *destination* paket data pada jaringan. Dalam kerjanya, *node* yang menjadi penyerang akan menempatkan dirinya pada jaringan dengan posisi yang paling dekat dengan *node* tujuan. Ketika *source node* mengirimkan RREQ ke jaringan, maka *node* penyerang akan mengirimkan RREQ

juga ke *destination node*. *Node* penyerang akan berusaha secepat mungkin untuk mengirimkan RREQ ke *destination node*. Ketika *destination node* menerima RREQ tercepat yang tiba kepadanya, maka *destination node* akan memberi jawaban RREP ke jaringan tanpa memperhatikan asal RREQ yang diterimanya. Dengan demikian, jalur paket data yang akan digunakan oleh jaringan adalah jalur dengan melewati *node* penyerang tersebut. (Seyyedtaj & Jamali, 2014).

Pada penelitian ini membahas serangan *rushing* pada AODV karena memiliki alasan tersendiri. Serangan *rushing* memanipulasi *control* dari *node malicious,* hal ini bertujuan agar rute yang terbentuk akan melewati *node malicious* sehingga paket yang dikirim dapat dicuri atau ditahan oleh *node malicious*. Efek dari serangan *rushing* sangat merugikan, hal ini menarik untuk diteliti.

Dengan media transmisi yang terbuka MANET, kesempatan untuk menyerang MANET akan selalu ada. Sebelum membangun komunikasi yang aman *node* yang saling terhubung harus dapat saling mengidentifikasi satu sama lain dengan cara masing-masing *node* diberikan identitas yang sudah diautentikasi dan dilindungi sehingga keasliannya dan integritas dari identitas masing-masing *node* (Bayu Aji, 2015).

1.2 Rumusan Masalah

Berdasarkan dari latar belakang yang telah dijelaskan dapat diambil rumusan masalah sebagai berikut :

1. Bagaimana kinerja dari *routing protocol* AODV pada MANET ketika terdapat serangan *rushing*?
2. Bagaimana dampak yang ditimbulkan dari serangan *rushing* terhadap *routing protocol* AODV pada MANET dilihat dari parameter ukur yang terpengaruh?

1.3 Tujuan

Tujuan yang ingin diperoleh pada penelitian ini adalah, diantaranya:

1. Dapat mengetahui karateristik dari serangan *rushing*.
2. Dapat mengetahui kinerja dari *routing protocol* AODV pada *Mobile Ad Hoc Networks* (MANET) jika mendapatkan serangan *rushing.*
3. Mengetahui dampak yang ditimbulkan dari serangan *rushing*.

1.4 Manfaat

Manfaat yang didapatkan dari penelitian ini adalah sebagai berikut.

- Memberikan kontribusi ilmiah pada penelitian mengenai serangan pada jaringan MANET terutama serangan *rushing*
- Memberikan referensi baru mengenai serangan pada MANET terutama serangan *rushing*
- Mendapatkan pengetahuan menegenai *routing* protokol AODV pada MANET.

- Mendapatkan wawasan mengenai serangan pada jaringan MANET

1.5 Batasan Masalah

Batasan masalah pada penelitian ini adalah sebagai berikut.

1. Pada pengerjaan hanya memilih satu algoritme *routing* yaitu AODV .
2. Banyaknya *node* yang digunakan sudah ditentukan sebelum pengujian dimulai.
3. Waktu simulasi yang akan dilakukan telah ditetapkan sebelum pengujian dilakukan.
4. Parameter pengujian yang dipakai pada penelitian antara lain *Packet Delivery Ratio* (PDR), *Throughput*, *End to end delay,* dan *Packet Loss*
5. Skenario disimulasikan dengan memakai *Network Simulator* 2.35.

1.6 Sistematika Pembahasan

Sistematika pembahasan adalah uraian dalam penyusunan penelitian ini yang dijelaskan sebagai berikut.

BAB 1 Pendahuluan

Isi dari bab ini diantaranya latar belakang dari masalah, rumusan masalah, tujuan dan manfaat, identifikasi dan pembatasan masalah, serta sistematika penulisaan.

BAB 2 Landasan pustaka

Bab ini membahas tentang teori-teori dari bermacam sumber pustaka yang berkaitan dengan penelitian ini sebagai referesensi dalam penelitian.

BAB 3 Metode Penelitian

Bab ini menjelaskan tahapan-tahapan dalam pengerjaan penelitian diantaranya studi literatur, analisis kebutuhan, perancangan sistem, implementasi, analsis, dan kesimpulan.

BAB 4 Perancangan Sistem

Bab ini menjelaskan perancangan sistem yang dipakai pada penelitian ini yang berisi perancangan simulasi, perancangan area simulasi, perancangan letak posisi *node*, perancangan serangan *rushing*, perancangan pergerakan *node*, dan perancangan parameter pengujian

BAB 5 Implementasi

Isi dari bab ini adalah proses implementasi dari penelitian terhadap *routing* protokol *Ad hoc On-Demand Distance Vector* (AODV) ketika mendaptakan serangan *rushing* dengan memakai *Network Simulator* 2.35.

BAB 6 Hasil dan Analisis

Bab ini merupakan hasil dan analisis dari implementasi serangan *rushing* pada *routing* protokol *Ad hoc On-Demand Distance Vector* (AODV)

BAB 7 Kesimpulan

Isi dari bab ini adalah kesimpulan yang didapat dari hasil implementsi. Pada bab ini juga berisi saran dari hasil pengujian yang sudah dilakukan serta perkembangan dari penelitian yang perlu ditingkatakan.

BAB 2 LANDASAN PUSTAKA

Bab ini berisi beberapa pustaka yang membantu penelitian ini. Landasan pustaka berisi beberapa penelitian terdahulu terpaut dengan *Simulation of Gray Hole Attack in Adhoc Network Using NS2* dan *Impact of* Serangan *rushing on Multicast in Mobile Ad Hoc Network*. Jurnal penelitian terdahulu, penulis jadikan rujukan, objek kajian, dan bagian dalam membantu penelitian ini.

2.1 Tinjauan Pustaka

Beberapa penelitian sebelumnya dijadikan oleh penulis sebagai pegangan untuk melakukan penelitian ini.

Tabel 2.1 Tinjauan Pustaka

Judul Penelitian	Persamaan	Pembeda	
		Penelitian Terdahulu	Rencana Penelitian
V. Palanisamy dan P. Annadurai, 2009. *Impact of* Serangan *rushing on Multicast in Mobile Ad Hoc Network*	Melakukan implementasi serangan *rushing* pada MANET	Jumlah *node* 50 – 100 *node* dan *malicious node* berjumlah 1 namun diletakkan dekat dengan *source node* dan *destination node.*	Jumlah *node* 30, 40, dan 50 dan *rushing attack node* berjumlah 3, 4, dan 5
	Implementasi menggunakan NS 2	Menggunakan perhitungan *delay* sebagai pengujian performa	Menggunakan perhitungan *Packet loss, Packet Delivery Ratio* (PDR), *End to End Delay*, *Throughput.*
	Menggunakan protokol *routing* AODV	Luas area yang digunakan 500 x 500	Luas area yang digunakan 1000 x 1000
Ms Meenakshi dan Mr. Kapil Kumar Kaswan, 2014. *Simulation of Gray Hole Attack in Adhoc Network Using NS2*	Implementasi menggunakan NS 2	Melakukan implementasi serangan *grey hole* pada MANET	
	Pengujian performa menggunakan *Throughput, End*	*Simulation time* 30 sec	*Simulation time* 500 sec

	to end delay, dan Packet delivery ratio		
	Menggunakan protokol *routing* AODV	Menggunakan perhitungan *packet loss* sebagai pengujian performa	Menggunakan perhitungan *Packet Delivery Ratio* (PDR), *End to End Delay*, *Trhoughput*.

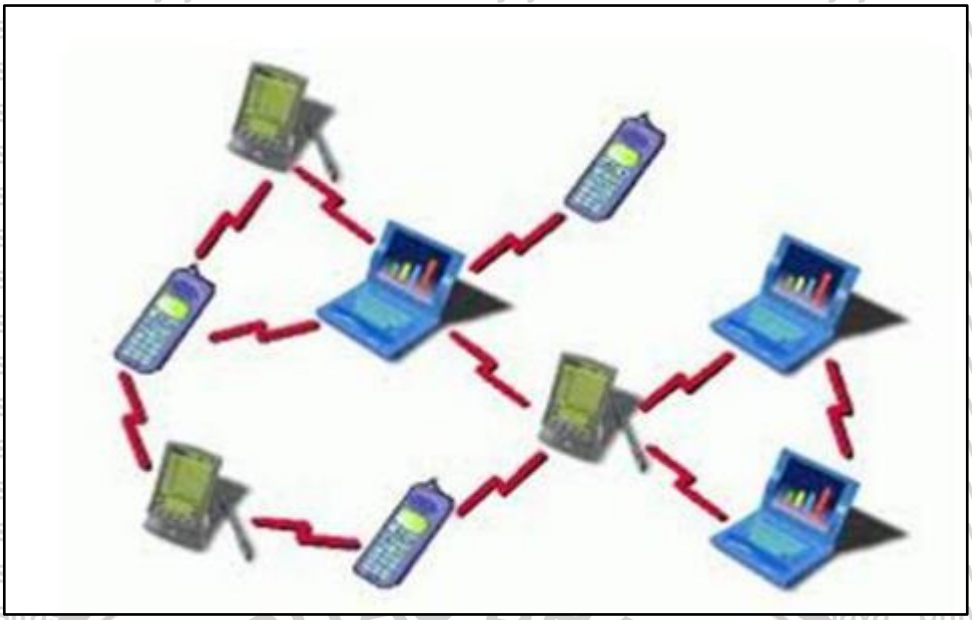
Diatas pada tabel 2.1 merupakan penelitian terdahulu yang penulis jadikan sebagai pegangan untuk melakukan penelitian ini dan terdapat faktor persamaan dan pembeda dari penelitian sebelumnya dengan penelitian yang oleh penulis lakukan.

2.2 Dasar Teori

Dasar teori menjelaskan beberapa teori dan beberapa faktor yang diperlukan dan aplikasi yang membantu pada penelitian ini.

2.2.1 Mobile Ad Hoc Network

Mobile ad-hoc network (MANET) adalah suatu jaringan yang terdiri dari beberapa perangkat bergerak atau yang disebut *node* tanpa memerlukan prasarana jaringan yang tetap, sehingga membuat jaringan yang bersifat temporer (Saputra, 2011). Setiap *node* dapat berfungsi sebagai *host* maupun *routers* sehingga jaringan dapat terbentuk dimana saja dan kapan saja selama dua atau lebih *node* terhubung dan saling berkomunikasi satu sama lain baik secara langsung antara *node* yang saling berhubungan maupun melalui simpul yang terbentuk dari *node* yang saling terhubung sesuai dengan topologinya dikarenakan fleksibilitas dari MANET (Roy, 2011). *Node* yang berperan sebagai router bisa bergerak dengan bebas dan dapat mengatur diri sendiri sehingga topologi jaringan dari MANET dapat berubah dengan cepat dan tidak terduga. Tiap *node* pada MANET dilengkapi dengan antena yang dapat menerima dan mengirimkan sinyal dapat secara terarah (*point to point*), *omni directional* (*broadcast*), atau kombinasi dari keduanya (Jadhav, Kulkarni, & Menon, 2014).



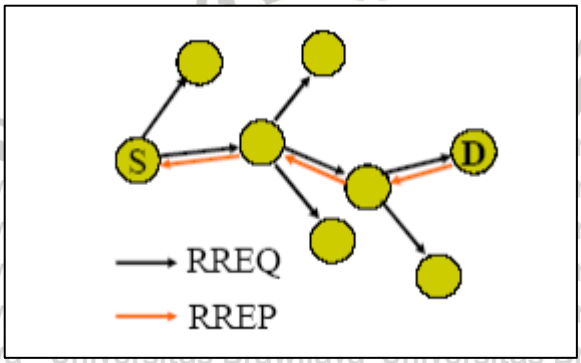
Gambar 2.1 Jaringan MANET

Sumber : Jayanti (2014)

Dalam gambar 2.1 menunjukkan MANET terbentuk dari beragam *mobile platform* yang simple sebagai *node* yang bergerak secara bebas dan acak.

2.2.2 *Ad Hoc On-Demand Distance Vector* (AODV)

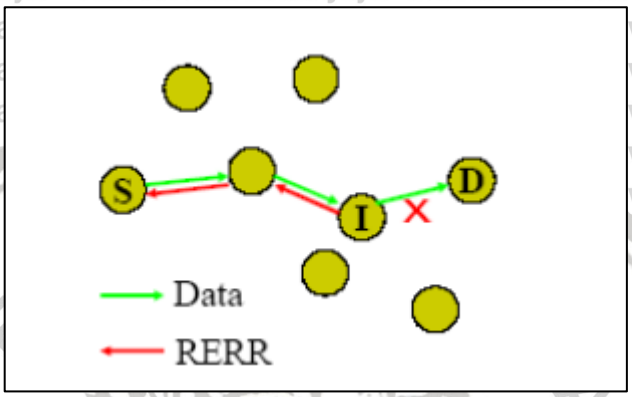
AODV merupakan salah satu protokol *routing reactive* artinya protokol ini akan bekerja ketika ada permintaan untuk mengirimkan data. Pada protokol ini terdapat tiga pesan utama yang digunakan untuk proses pembentukan jalur dan pemeliharaan jalur yaitu: *route request* (RREQ), *route replay* (RREP), dan *route error* (RRER). AODV berjalan secara dinamis, *self-starting*, dan *routing multihop* (Sari, Syarif, & Budiardjo, 2010) .



Gambar 2.2 Mekanisme Pencarian Rute

Sumber : (Sari, Syarif, & Budiardjo, 2010)

Pencarian jalur (*Path discovery*) atau *Route discovery* diawali dengan menyebarkan RREP, seperti dalam Gambar 2.2. Saat RREP menjelajahi *node* RREP dapat otomatis membentuk jalur. Jika *node* menerima RREP, maka *node* tersebut akan mengirimkan RREP lagi ke *node* atau *destination sequence number.* Pada proses ini, *node* pertama kali akan mengecek *destination sequence number.* Pada proses ini, *node* pertama kali memerika *destination sequence number* pada tabel *routing*, apakah lebih besar dari 1 (satu) pada RREQ, jika benar, maka *node* akan mengirim RREP. Ketika RREP berjalan kembali ke source melalui *path* yang telah di-setup, ia akan men-setup jalur kedepan dan meng-*update timeout*.



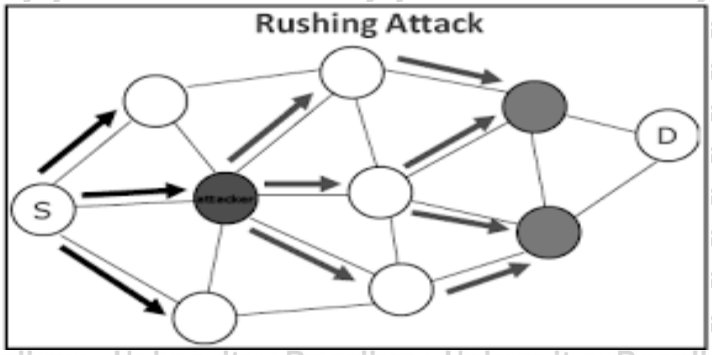
Gambar 2.3 Mekanisme Pemeliharaan Jalur

Sumber : (Sari, Syarif, & Budiardjo, 2010)

Jika sebuah *link* ke *hop* berikutnya tidak terdeteksi dengan metode penemuan rute, maka *link* tersebut dinyatakan terputus dan RERR akan disebarkan ke *node* tetangganya seperti dalam Gambar 2.3. Sebuah *node* dapat menghentikan pengiriman data melalui rute ini dan meminta rute baru dengan menyebarkan RREQ kembali.

2.2.3 Rushing

Serangan *rushing* memanfaatkan RREQ pada *route discovery* untuk menjadi *node* penghubung antara *source* dan *destination* paket data pada jaringan. Dalam kerjanya, *node* yang akan menjadi penyerang akan menempatkan dirinya pada jaringan dengan posisi yang paling dekat dengan *node* tujuan. Ketika *source node* mengirimkan RREQ ke jaringan, maka *node* penyerang akan mengirimkan RREQ juga ke *destination node*. *Node* penyerang akan berusaha secepat mungkin untuk mengirimkan RREQ ke *destination node*. Ketika *destination node* menerima RREQ tercepat yang tiba kepadanya, maka *destination node* akan memberi jawaban RREP ke jaringan tanpa memperhatikan asal RREQ yang diterimanya. Dengan demikian, jalur paket data yang akan digunakan oleh jaringan adalah jalur dengan melewati *node* penyerang tersebut. (Seyyedtaj & Jamali, 2014).



Gambar 2.4 Serangan *Rushing*

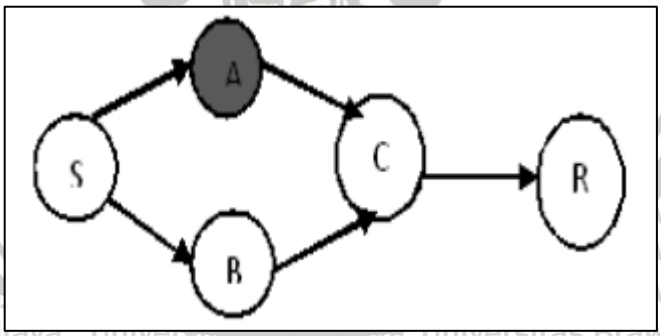
Sumber : (Jain & Sharma, 2014)

Pada gambar 2.4 menunjukkan mekanisme serangan *rushing. Attacker node* seolah-olah menjadi jalur terpendek sehingga data dikirim melalui *attacker node* dan data tersebut akan diduplikasi oleh *attacker node*.

Paket RREQ dalam protokol *On Demand Routing* diteruskan untuk menemukan rute mencapai tujuan dan meminimalkan *overhead* dalam jaringan dengan mengirimkan paket permintaan rute pertama ke tujuan untuk menemukan rute. Penyerang mendapatkan paket RREQ dari *source node* atau di sekitar *node* lainnya lalu meneruskannya untuk mencapai tujuan lebih cepat daripada *node* lainnya. Sehingga *node receiver* menganggapnya sebagai paket RREQ yang dikirim oleh *source node* dengan mengotentikasi *node*. Sehingga *destination node* akan membuang paket RREQ yang baru tiba dan menetapkan rute antara *source node* dan *destination node* dengan penyerang di tengah untuk mendapatkan akses antara dua *node* (Thilagarasi & Geetha, 2015).

Serangan *rushing* dapat berada dimana saja dalam jaringan, seperti berikut :

- Serangan berada di dekat *sender*

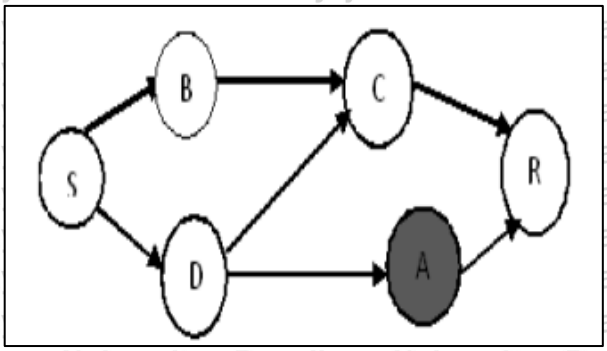


Gambar 2.5 *Node Attacker* Dekat *Source Node*

Sumber : (Thilagarasi & Geetha, 2015)

Pada gambar 2.5 *node attacker* A berada didekat *source node*. Paket RREQ berasal dari *source node* S meneruskan paket *request* ke *node* A dan *node* B. *Node* A atau *node attacker* dengan cepat meneruskan paket ke *node* C daripada *node* B, maka paket dari *node A* yang melalui *node* C tiba lebih dulu di *node R* dari pada paket dari *node* B.

- Serangan berada di dekat *receiver*

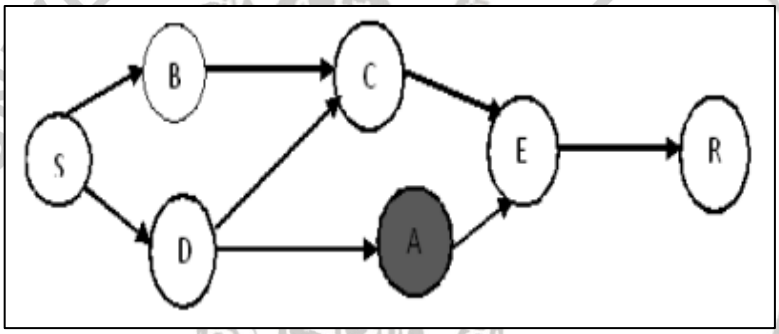


Gambar 2.6 *Node Attacker* Dekat *Destination Node*

Sumber : (Thilagarasi & Geetha, 2015)

Pada gambar 2.6 *node attacker* A berada di dekat *destination node*. Di sini paket RREQ dimulai dari S *source node* diteruskan ke B & D, kemudian B meneruskannya ke C dan D meneruskan paket ke A & C, *node attacker* A kemudian meneruskannya dengan cepat ke *node* R daripada *node* C. Akhirnya R membuang paket yang terakhir tiba dari *node* lainnya.

- Serangan berada dimana saja



Gambar 2.7 *Node Attacker* Dimana Saja

Sumber : (Thilagarasi & Geetha, 2015)

Pada gambar 2.7 *node attacker* A berada di tengah-tengah jaringan. Paket *request route* dimulai oleh S *source node*, lalu meneruskannya ke B & D, *node* B meneruskannya ke C, *node* D meneruskan paket ke C & A, E mendapat paket permintaan yang dikirim oleh *node attacker* A dari C. Akhirnya R *destination node* menerima paket permintaan yang dikirim oleh *node attacker* daripada *node* lainnya.

2.2.4 Quality of Service (QoS)

Quality of Service (QoS) adalah kemampuan sebuah jaringan untuk menyediakan layanan yang lebih baik bagi layanan trafik yang melewatinya. Baik atau buruknya kualitas dan kemampuan suatu jaringan dapat diukur dari kinerja jaringan tersebut. Parameter *Quality of Service* terdiri dari :

a. *Throughput*

Throughput adalah banyaknya *byte* yang diterima pada waktu tertentu dengan satuan *byte/second* atau dengan kata lain kondisi data *rate* sebenarnya

pada sebuah jaringan. *Throughput* dapat dinyatakan dengan rumus sebagai berikut:

$$Throughput = \frac{jumlah\ paket\ yang\ diterima}{total\ waktu\ pengamatan}\ x\ ukuran\ paket$$

Throughput dapat mengalami penurunan yang diakibatkan oleh adanya jarak yang jauh antara *source node* dengan *destination node* sehingga bertambahnya jumlah *hop routing*. Semakin besar nilai *throughput* maka semakin baik pula kinerja sebuah jaringan.

b. *End to End Delay*

End to end delay adalah waktu yang terjadi pada saat proses pengiriman suatu paket dari titik awal menuju titik tujuan. *end to end delay* terdiri dari waktu pengiriman, propagasi, proses, dan antrian dari suatu paket pada setiap *node* dalam jaringan. *end to end delay* dapat dirumuskan sebagai berikut :

$$End\ to\ end\ delay = \frac{waktu\ paket\ diterima - waktu\ paket\ dikirimkan}{jumlah\ paket\ yang\ dikirim}$$

Faktor utama yang memberikan pengaruh pada *end to end delay* adalah waktu yang digunakan untuk menemukan rute, hal ini dikarenakan sebelum mengirimkan pesan *node* harus mengenal jalur yang akan dilalui untuk mencapai *destination*. Nilai *end to end delay* yang rendah menunjukkan performansi yang baik.

c. *Packet Loss*

Packet loss adalah suatu parameter yang menggambarkan suatu kondisi yang menunjukkan jumlah total paket yang hilang. Beberapa penyebab terjadinya *packet loss* yaitu: tabrakan (*congestion*), memory yang terbatas pada *node, node* yang bekerja melebihi kapasitas *buffer.* Hal ini mempengaruhi kinerja sebuah jaringan secara keseluruhan, walaupun *bandwidth* yang tersedia sudah mencukupi. Secara umum terdapat empat kategori penurunan performansi jaringan berdasarkan nilai *packet loss* yaitu seperti tampak pada tabel.

Tabel 2.2 Kategori *Packet Loss*

Kategori Degredasi	*Packet Loss*
Sangat Bagus	0%-2%
Bagus	3%-14%
Sedang	15%-24%
Buruk	>25%

Sumber : (Iskandar & Hidayat, 2015)

Pada tabel 2.2 dijelaskan kategori dalam *Packet Loss,* terdapat beberapa kategori dianataranya Sangat bagus, Bagus, Sedang, dan Buruk. Untuk mendapatkan nilai *packet loss*, dapat ditentukan dengan rumus :

$$packet\ loss = \frac{packet\ send - packet\ received}{packet\ send}\ x\ 100\%$$

d. *Packet Delivery Ratio* (PDR)

Packet delivery ratio (PDR) merupakan rasio antara banyaknya paket yang diterima oleh tujuan dengan banyaknya paket yang dikirimkan oleh sumber. *packet delivery ratio* dapat dijadikan sebagai acuan sebagai tingkat kesuksesan suatu protokol *routing* dalam pencarian dan pemeliharaan rutenya. Semakin tinggi nilai *packet delivery ratio* maka akan semakin baik pula performa jaringan. Untuk mendapatkan nilai *packet delivery ratio*, dapat ditentukan dengan rumus :

$$PDR = \frac{Pr}{Ps} x100\%$$

0 ≤ t ≤ T dimana : Pr = Paket yang diterima
Ps = Paket yang dikirim
T = Waktu simulasi (detik)
t = Waktu pengambilan sampel (detik)

2.2.5 *Random Way Point*

Random way point merupakan model pergerakan secara acak yang digunakan untuk simulasi. Dalam *random way point node* bergerak secara acak dan bebas tanpa adannya batasan dengan arah dan kecepatan yang sudah ditentukan. Untuk membentuk skenario *random way point* digunakan sebuah *tool* yang terdapat pada NS2, *tool* tersebut bernama setdest. Pada *tool* setdest *node* dapat diatur pergerekannnya dan tujuan pergerakannya. Untuk menjalankan setdest, diperlukan perintah sebagai berikut :

"setdest [-v num of version] [-n num of *nodes*] [-p pausetime] [-t simtime] [-x max x] [-y max y] > [name file]

Parameter yang digunakan pada perintah untuk menjalankan *tool* setdest dijelaskan pada Tabel 2.3.

Tabel 2.3 Parameter Baris Perintah untuk Menjalankan Setdest

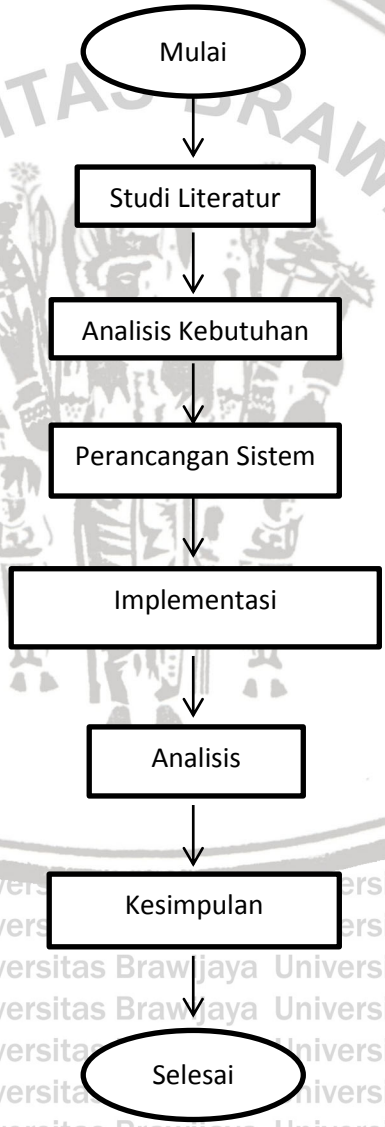
No	Parameter	Keterangan
1	-v	Versi yang digunakan
2	-n	Jumlah *node*
3	-p	Lama waktu *node* diam ketika tiba di lokasi pergerakan
4	-M	Kecepatan *node* bergerak
5	-t	Waktu simulasi
6	-x	Panjang maksimal area simulasi
7	-y	Lebar maksimal area simulasi

BAB 3 METODOLOGI PENELITIAN

Bab ini berisi mengenai metodologi penelitian yang digunakan pada penelitian ini. Metodologi penelitian membahas tahapan-tahapan dalam melakukan penelitian serta perangkat-perangkat yang digunakan dalam penelitian.

3.1 Alur Metode Penelitian

Alur metode penelitian merupakan suatu tingkatan yang digunakan dalam menyelesaikan penelitian atau dalam penulisan skripsi. Berikut merupakan alur metode penelitian yang dilakukan :



Gambar 3.1 Diagram Alir Penelitian

Pada gambar 3.1 adalah tahapan-tahapan yang digunakan pada penelitian ini dan digambarkan pada bentuk diagram alir.

3.2 Studi Literatur

Studi literatur menggambarkan teori yang dipakai sebagai penunjang dalam perancangan dan pengimplementasian sistem. Untuk dapat merancang sistem tersebut, berikut adalah dasar teori penunjang yang digunakan :

1. Protokol *routing* AODV
2. Serangan Rushing
3. *Network Simulator* 2.35
4. Parameter uji : - *packet delivery ratio*
   - *packet loss*
   - *throughput*
   - *end to end delay*
5. Jenis *node mobility* : *Random way point*

3.3 Analisis Kebutuhan

Analisis kebutuhan dilakukan untuk memahami kebutuhan apa saja yang digunakan oleh sistem untuk melakukan simulasi, sehingga tujuan dari penelitian ini dapat tercapai. Analisis kebutuhan terdapat dua jenis, yaitu analisis kebutuhan perangakat keras dan analisis kebutuhan perangkat lunak. Kebutuhan perangkat keras merupakan suatu perangkat yang akan digunakan untuk menunjang penelitian, seperti laptop. Kebutuhan perangkat lunak merupakan aplikasi atau *software* yang akan digunakan dalam penelitian, misalnya linux.

3.3.1 Kebutuhan Perangkat Keras

Kebutuhan perangkat keras yang dibutuhkan dalam penelelitian ini adalah :

a. Laptop
b. Prosesor Intel Core i5 @ 1,60 GHz
c. RAM 4 GB

3.3.2 Kebutuhan Perangkat Lunak

Kebutuhan perangkat lunak yang dibutuhkan untuk mewujudkan sistem dalam penelitian ini, diantaranya adalah :

a. VirtualBox
b. Ubuntu 16.04
c. Network Simulator 2.35

3.4 Perancangan Sistem

Pada penelitian ini apabila dilakukan secara langsung akan membutuhkan biaya yang sangat besar. Sehingga pada penelitian ini dilakukan melalui simulasi dengan menggunakan Network Simulator 2.35. Simulasi ini dapat dijadikan sebagai gambaran apabila penelitian ini dilakukan secara langsung.

3.4.1 Perancangan Simulasi

Perancangan simulasi adalah perancangan parameter-parameter yang digunakan pada saat melakukan simulasi dengan tujuan agar simulasi dapat berjalan sesuai dengan skenario yang sudah ditentukan. Parameter - parameter yang digunakan misalnya protokol *routing*, model pergerakan, trafik yang digunakan, serta lamanya simulasi dilakukan.

3.4.2 Perancangan Area Simulasi

Perancangan area simulasi dilakukan untuk menentukan seberapa luas area yang akan digunakan untuk melakukan simulasi. Area simulasi ini nantinya menjadi tempat bergeraknya *node – node* yang akan digunakan pada simulasi. Area simulasi ditentukan melalui sumbu kartesius (X, Y) yang nantinya membentuk luas area simulasi.

3.4.3 Perancangan Letak *Node*

Perancangan letak *node* bertujuan untuk menentukan posisi *node – node* pada area simulasi yang sudah dibentuk. *Node – node* ini nantinya sebagai penunjang parameter simulasi yang sudah ditentukan. *Node – node* ini akan bervariasi jumlahnya, hal ini bertujuan untuk mengetahui mekanisme serangan rushing. *Node – node* ini akan menempati sumbu kartesius (X, Y).

3.4.4 Perancangan Serangan

Perancangan serangan dilakukan dengan memodifikasi algoritma protokol *routing* AODV supaya dapat mengimplementasikan serangan *rushing* pada protokol *routing* AODV. Hal-hal yang ditambahkan pada algoritma protokol *routing* AODV diantaranya dengan menambahkan beberapa fungsi pada mekanisme *route request* dan *foward packet* pada *routing table* agar sifat dari serangan *rushing* dapat diimplementasikan pada simulasi, selain itu juga ditambahkan fungsi dari serangan *rushing* pada algoritma protokol *routing* AODV. Setelah melakukan modifikasi pada algoritma protokol *routing* AODV juga harus ditentukan posisi atau letak dari serangan *rushing* pada simulasi, dimana pada penelitian ini serangan *rushing* dimasukkan dalam beberapa *node* yang sudah dibentuk.

3.4.5 Perancangan Pergerakkan *Node*

Perancangan pergerakkan *node* bertujuan untuk menentukan pergerakan *node – node* pada saat simulasi dilakukan. Perancangan pergerakan *node* juga menentukan mekanisme pergerakan *node* yang digunakan karena setiap mekanisme pergerakan *node* memiliki sifat yang berbeda pula. Pada pergerakan *node* beberepa hal yang harus diperhatikan, antara lain waktu *node* bergerak, sumbu kartsius (X, Y) sebagai tujuan untuk bergerak, dan kecepatan bergerak.

3.4.6 Perancangan Parameter Pengujian

Perancangan parameter pengujian bertujuan untuk mendapatkan hasil perhitungan dari simulasi yang digunakan untuk menganalisis data. Parameter pengujian yang digunakan antara lain *packet delivery ratio, packet loss,*

throughput, dan *end to end delay*. Setelah selesai menjalankan parameter pengujian selanjutnya dilakukan analisis yang diperoleh dari hasil pengujian.

3.5 Implementasi

Software yang dipakai untuk melakukan simulasi dari perancangan yang sudah dibuat adalah *Network Simulator* 2.35 dan *nam* yang digunakan untuk menampilkan simulasi dari *script* yang sudah dibuat. Pada simulator akan di implementasikan protokol *routing* AODV dan model pergerakan *Random Way Point* dalam bentuk *script* yang dijalankan pada simulator. Berikut faktor pendukung yang dipakai :

1. Algoritme *routing* AODV.
2. Algoritme *routing* AODV yang telah dimodifikasi untuk melakukan serangan *rushing.*
3. Script protokol *routing* AODV dengan parameter–parameter simulasi yang sudah ditentukan, seperti model pergerakan, luas area, jumlah *node*, jumlah *node* serangan, dan waktu simulasi
4. Bahasa pemrograman menggunakan C++

3.6 Analisis Hasil

Analisis hasil dilakukan untuk mengetahui efek yang ditimbulkan dari serangan *rushing*. Hasil analisis dapat diketahui dari hasil perhitungan yang sudah dilakukan dalam parameter pengujian. Dari analisis hasil ini dapat dijadikan dasar untuk menarik kesimpulan.

3.7 Kesimpulan

Pada bagian ini berisi kesimpulan dan saran. Kesimpulan akan dilakukan berdasarkan hasil pengujian yang sudah dianalisa yang kemudian akan menjawab rumusan masalah yang ditentukan di awal. Saran akan berisi tentang usulan dan harapan dari penulis untuk membantu pengembangan jaringan *Mobile Ad Hoc Network* (MANET) terlebih dalam sistem keamanan.

BAB 4 PERANCANGAN SISTEM

Perancangan Sistem mencakup perancangan simulasi, perancangan area simulasi, perancangan letak posisi *node*, perancangan serangan *rushing*, perancangan pergerakan *node*, dan perancangan parameter pengujian

4.1. Perancangan Simulasi

Perancangan simulasi menjelaskan parameter-parameter yang digunakan pada simulasi serangan. Parameter-parameter ini dipakai sebagai dasar pada proses simulasi. Parameter yang berbeda dalam setiap skenario akan menimbulkan perbedaan hasil nantinya.

Tabel 1.1 Parameter Simulasi

Parameter	Nilai
Jumlah *Node*	30, 40, 50
Jumlah *Node Malicious*	3,4,5
Jenis Protokol	AODV
Jenis Serangan	*Rushing*
Simulator	*Network Simulator* 2.35
Chanel Type	*Wireless Channel*
Mobility Model	*Random Way Point*
Model Propagasi	*Two Ray Ground*
Network Interface	*WirelessPhy*
Antena Mode	*OmniAntenna*
Ukuran Paket Data	1024 *bytes*
Jenis Trafik	CBR (*Constant Bit Rate*)
Luas Area Jaringan	1000 m x 1000 m
Kecepatan *Node*	10 m/s
Waktu Simulasi	1000 s

Pada tabel 4.1 menunjukkan parameter-prameter yang digunakan pada simulasi.

Berikut merupakan spesifikasi sistem yang digunakan untuk melakukan simulasi pada penelitian ini :

1. Jumlah *node* yang dipakai dalam simulasi
2. Jumlah *node malicious* yang akan dipakai dalam simulasi
3. Protokol *routing* yang dipakai pada proses simulasi adalah *Ad Hoc on Demand Distnce Vector* (AODV).
4. Jenis serangan yang dipakai adalah serangan *rushing*
5. Simulator yang dipakai untuk melakukan simulasi adalah *Network Simulator* versi 2.35
6. *Channel Type* yang pakai adalah *Wireless Channel*
7. Model mobilitas yang dipakai *Random Way Point*
8. Model propagasi yang dipakai adalah *Two Ray Ground*

9. Menggunakan *WirelessPhy* sebagai *Network Interface*
10. Menggunakan *Antena Mode Omni Antena*
11. Ukuran paket yang dipakai sebesar 1024 *bytes*.
12. Jenis trafik yang dipakai CBR (*Constant Bit Rate*).
13. Luas wilayah yang dipakai 1000m x 1000m.
14. Waktu simulasi 1000 sec.

4.2 Perancangan Area Simulasi

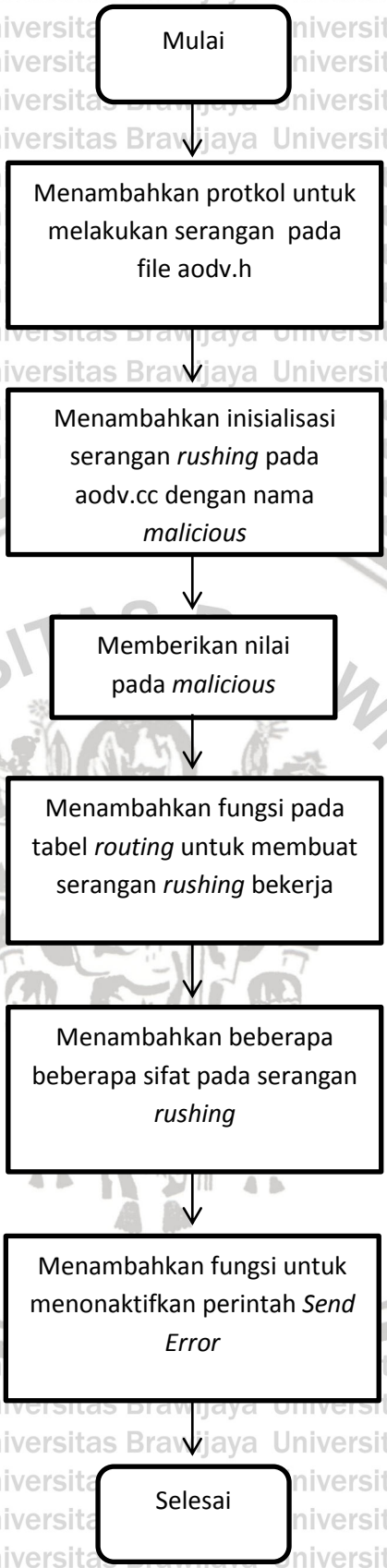
Pada penelitian ini memakai area simulasi dengan luas 1000x1000m dengan berpatokan pada sumbu kartesius (X, Y). Pada area seluas 1000x1000m ini akan di tempatkan *node* dengan variasi jumlah 30, 40, dan 50. Pada luas 1000x1000m ini juga menjadi batas pergerakan *node* pada simulasi.

4.3 Perancangan Serangan Rushing

Pada bagian perancangan serangan menjelaskan mengenai skenario penyerangan pada protokol *routing* AODV dalam jaringan *Mobile Ad Hoc Network* (MANET) dengan menambahkan beberapa fungsi untuk melakukkan serangan *rushing* pada protokol *routing* AODV.

1. Pada aodv.h ditambahkan sebuah protokol untuk melakukan serangan
2. Pada aodv.cc dilakukan inisialisasi untuk menambahkan serangan *rushing* dengan nama "rushingattack" dan diberikan nilai sebesar 1000
3. Penambahan fungsi pada tabel *routing*, dimana fungsi ini bertujuan untuk membuat serangan *rushing* dapat berperilaku seperti *node* lainnya yaitu melakukan *route request* namun serangan *rushing* akan melakukan *forward* packet lebih cepat daripada *node* lainnya sehingga rute yang terbentuk akan melalui serangan *rushing*. Fungsi yang ditambahkan inilah merupakan karakteristik dari serangan *rushing.*
4. Serangan *rushing* dapat membuang semua paket data atau beberapa paket data dan juga menahan paket data atau melakukan permintaan ulang (*re-ordering*).
5. Serangan *rushing* akan menjatuhkan paket karena tidak ada rute tujuan. *Node* yang terinfeksi serangan *rushing* harus menonaktifkan pengiriman *error*. Hal ini dimaksudkan agar serangan *rushing* tidak mudah terdeteksi.

Berikut merupakan diagram alir perancangan serangan *rushing* yang ditunjukkan pada gambar 4.1



Gambar 4.1 Diagram Alir Perancagan Serangan *Rushing*

Pada file .tcl ditambahkan fungsi untuk menjalankan serangan *rushing*. Fungsi yang ditambahkan ini mengambil dari fungsi yang telah ditambahkan pada file aodv.cc. Serangan *rushing* masuk ke dalam *node* yang sudah dibentuk. Serangan *rushing* memanfaatkan *Route Request* (RREQ) pada *route discovery* untuk menjadi *node* penghubung antara *source* dan *destination* paket data pada jaringan. Dalam kerjanya, *node* yang akan menjadi penyerang akan menempatkan dirinya pada jaringan dengan posisi yang paling dekat dengan *node* tujuan. Ketika *source node* mengirimkan RREQ ke jaringan, maka *node* penyerang akan mengirimkan RREQ juga ke *destination node*. *Node* penyerang akan berusaha secepat mungkin untuk mengirimkan RREQ ke *destination node*. Ketika *destination node* menerima RREQ tercepat yang tiba kepadanya, maka *destination node* akan memberi jawaban *Route Reply* (RREP) ke jaringan tanpa memperhatikan asal RREQ yang diterimanya. Dengan demikian, jalur paket data yang akan digunakan oleh jaringan adalah jalur dengan melewati *node* penyerang tersebut. Dibawah ini digambarkan *flowchart* mengenai mekanis *routing prtocol* AODV dan mekanisme serangan *rushing* pada AODV pada tabel 4.2 dan 4.3.

Tabel 4.2 Pseudocode AODV

```
    //method broadcast RREQ
1   SendRREQ (nodeX)
2   BROADCAST RREQ to Neighbors
    //method handling RREQ
3   ReceiveRREQ (RREQ, nodeX){
4       If {
5         (nodeX == Destination) UPDATE Route, Send
    RREP (nodeX,   RREQ)
6       }
7       Else {
8         FOWARD RREQ, UPDATE RREQ
9       }
10  UPDATE Route
11  }
    //method broadcast RREP
12  SendRREP (nodeX, RREQ)
13  BROADCAST RREP to Neighbors
    //method handling RREP
14  ReceiveRREP (RREP, nodex){
15      If {
16        (nodeX == Source) UPDATE Route, Send DATA
17      }
18      Else {
19        FOWARD RREP, UPDATE RREP
20      }
21  }
```

Tabel 4.3 Pseudocode AODV dengan Serangan *Rushing*

```
1   //method broadcast RREQ
    SendRREQ (nodeX)
2   BROADCAST RREQ to Neighbors
3   Rushing receive RREQ
4   Rushing BROADCAST RREQ to Neighbors
    //method handling RREQ
5   ReceiveRREQ (RREQ, nodeX, rushing){
6       If (nodeX == Destination){
7           UPDATE Route, Send RREP (nodeX, RREQ,
8   rushing)
9       }
10      Else {
11        FOWARD RREQ, UPDATE RREQ
12      }
13  UPDATE Route
    }
14  //method broadcast RREP
15  SendRREP (nodeX, RREQ, rushing)
    BROADCAST RREP to Neighbors
16  //method handling RREP
17  ReceiveRREP (RREP, nodex, rushing){
18      If (nodeX == Source){
19        UPDATE Route, Send DATA
20      }
21      Else {
22        FOWARD RREP, UPDATE RREP
23      }
24  }
25  rushing receive DATA (DATA) {
26      If (t < CURRENT TIME){
27        DATA Hold
28      }
29      Else {
30        DATA Delete
31      }
    }
```

4.4 Perancangan Pergerakan *Node*

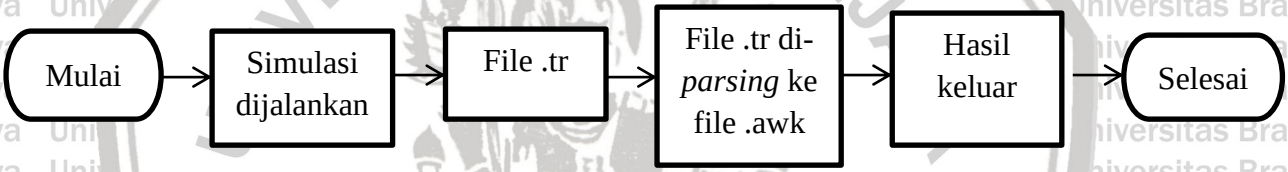
Perancangan pergerakan *node* dibentuk menurut luas area simulasi yang sudah di rancang. Terdapat 3 variasi jumlah *node* yang disimulasikan pada luas area simulasi yaitu 30, 40, dan 50 *node*. Pergerakan *node* menggunakan mekanisme pergerakan *random way point* dan juga diatur waktu dan kecepatannya, dimana pada perancangan simulasi telah ditetapkan waktu

simulasi selama 100s dan kecepatan 50 m/s. Untuk menghasilkan pergerakan *random way point* digunakan *tool* yang bernama setdest. Berikut baris perintah untuk membentuk pergerakan pada *node* 30, 40, dan 50 dengan kecepatan maksimal 50 m/s dan waktu simulasi selama 100s

- "setdest -v 1 -n 30 -p 0 -M 50 -t 100 -x 1000 -y 1000 > scenario_30"
- "setdest -v 1 -n 40 -p 0 -M 50 -t 100 -x 1000 -y 1000 > scenario_40"
- "setdest -v 1 -n 50 -p 0 -M 50 -t 100 -x 1000 -y 1000 > scenario_50"

4.5 Perancangan Parameter Pengujian

Pada bagian perancangan parameter pengujian ini menjelaskan paramater apa saja yang akan dipakai pada penelitian ini untuk dapat mengetahui dampak dari serangan *rushing.* Pada penelitian ini digunakan file .awk sebagai perhitungan dalam parameter pengujian. Untuk mendapatkan pengujian akan ditampilkan alur dalam bentuk diagram alir pada Gambar 4.2



Gambar 4.2 Diagram Alir untuk Mendapatkan Hasil Perhitungan

Gambar 4.2 menunjukkan tahapan-tahapan untuk mendapatkan hasil perhitungan, dimana untuk mendapatkan hasil perhitungan file .tr yang berisi hasil implentasi di *parsing* ke file .awk yang berisi rumus perhitungan sehingga mengeluarkan hasil perhitungan. File awk akan mengambil beberapa variabel yang ada pada *trace* file untuk dijadikan sebagai bahan dalam melakukan perhitungan. Berikut merupakan contoh variabel pada file awk beserta kodenya yang ditunjukkan pada tabel 4.4.

Tabel 4.4 Tabel Variabel File AWK

kode	variabel
$1	*Event*
$2	*Time*
$3	*Node id*
$4	*Layer*
$5	*Flags*
$6	*Sequence Number*
$7	*Packet Type*
$8	*Packet Size*

BAB 5 IMPLEMENTASI

Dalam bab ini membahas implementasi serangan *rushing* pada protokol AODV pada jaringan MANET. Implementasi dilakukan dengan mengunakan *Network Simulator* 2.35 (NS-2.35) dan animasi untuk simulasi menggunakan NAM. Implementasi dilakukan dengan menggunakan jumlah *node* yang berbeda, yaitu 30, 40, dan 50 *node* dan jumlah *node attack* yang berbeda pula, yaitu 3 *node*, 4 *node*, dan 5 *node* dan juga menggunakan perbedaan pergerekan *node* dimana *node attack* diam dan bergerak. Sebelum menjalankan simulasi ditambahkan beberapa fungsi dan protokol pada file aodv.h, aodv.cc dan file .tcl agar serangan *rushing* dapat berjalan. Berikut beberapa fungsi yang ditambahkan.

1. Pada aodv.h ditambahkan fungsi *nsaddr_t malicious* untuk menentukan serangan *rushing* seperti yang ditunjukkan pada tabel 5.1 berikut. Penambahan fungsi bertujuan untuk menetapkan serangan *rushing* pada algoritma protokol *routing* AODV.

Tabel 5.1 Penambahan fungsi nsaddr_t malicious pada aodv.h

```
1  nsaddr_t malicious;
```

2. Pada aodv.cc dilakukan inisialisasi untuk menambahkan serangan *rushing* dengan nama "rushingattack" dan nilai dari malicious = 1000 seperti yang terlihat Tabel 5.2, dimana nama ini digunakan pada file .tcl untuk melakukan serangan *rushing.* Serta pada *constructor* juga ditambahkan fungsi *malicious* seperti yang terlihat pada Tabel 5.3

Tabel 5.2 Inisialisasi Serangan Pada aodv.cc

```
1  if(strcmp(argv[1], "rushingattack") ==
2  0) {
3       malicious= 1000;
4                  return TCL_OK;
       }
```

Tabel 5.3 Penambahan Nilai Malicious Pada Constructor di AODV.cc

```
1  malicious = 900
```

3. Penambahan fungsi pada tabel *routing*, seperti pada tabel 5.4 dimana fungsi ini bertujuan untuk membuat serangan *rushing* dapat berperilaku seperti *node* lainnya yaitu melakukan *route request* namun serangan *rushing* akan

melakukan *forward* packet lebih cepat daripada *node* lainnya sehingga rute yang terbentuk akan melalui serangan *rushing*. Fungsi yang ditambahkan inilah merupakan karakteristik dari serangan *rushing*.

Tabel 5.4 Penambahan Fungsi pada *Routing Table*

```
1   if (malicious!=1000)
2       rq->rq_hop_count += 1;
3       // Maximum sequence number seen en
4   route
5       if (rt) rq->rq_dst_seqno = max(rt-
6   >rt_seqno, rq->rq_dst_seqno);
7       if (malicious==1000)
8       forward((aodv_rt_entry*) 0, p, 0);
9       else
10      forward((aodv_rt_entry*)     0,     p,
    DELAY);
```

4. Serangan *rushing* dapat membuang semua paket data atau beberapa paket data dan juga menahan paket data atau melakukan permintaan ulang (*re-ordering*). Seperti pada Tabel 5.5 terdapat fungsi *drop* hal inilah yang akan membuang paket data yang masuk ke dalam serangan *rushing* dan terdapat fungsi *forward* hal inilah yang menyebabkan *delay* pada paket data yang melalui serangan *rushing*

Tabel 5.5 Penambahan Fungsi untuk Serangan *Rushing*

```
1   if((ch->ptype()!=PT_AODV)                    &&
2   (malicious==1000))
3               {
4                if(t<CURRENT_TIME)
5                {
6                t=t+1;
7               drop(p, DROP_RTR_NO_ROUTE);
8           }
9       else
10      forward(rt, p, 0.8);
        }
```

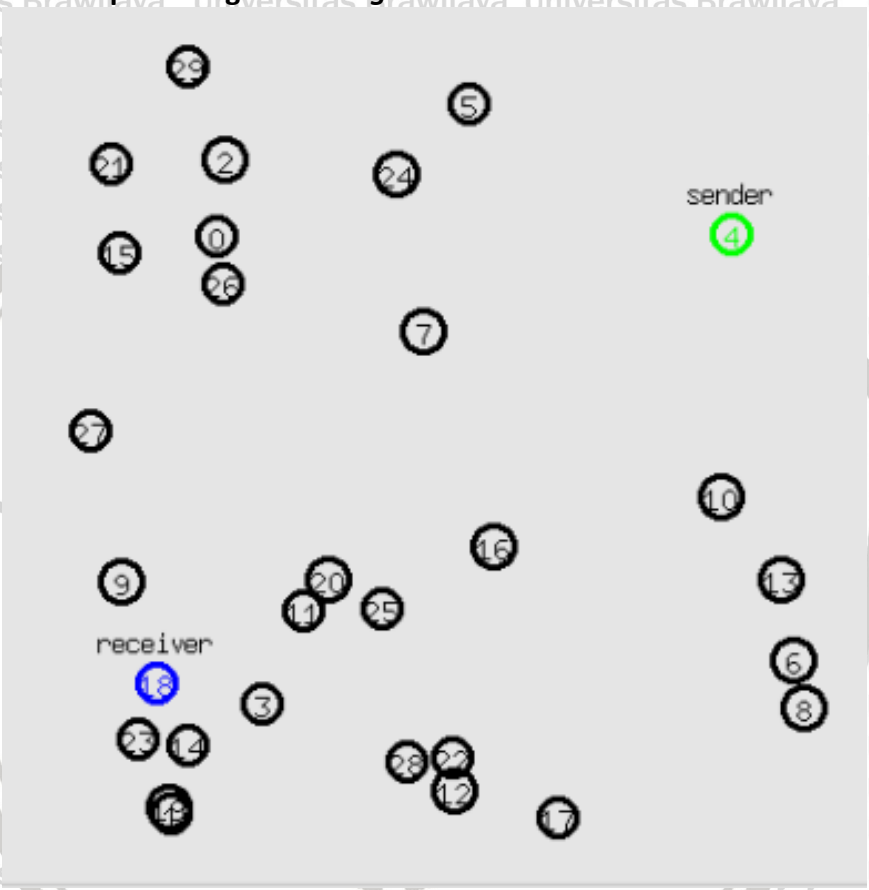
5. Serangan *rushing* akan menjatuhkan paket karena tidak ada rute tujuan *node* yang terinfeksi serangan *rushing* harus menonaktifkan pengiriman *error*. Hal ini dimaksudkan agar serangan *rushing* tidak mudah terdeteksi. Dapat dilihat pada Tabel 5.6 terdapat fungsi *sendError* pada baris 14 yang berfungsi untuk menonaktifkan pengeriman *error*.

Tabel 5.6 Penambahan Fungsi untuk Menonaktifkan *Send Error*

```
1  if     (malicious==1000)          drop(p,
2  DROP_RTR_NO_ROUTE);
3  else
```

5.1 Implementasi Tanpa Serangan *Rushing*

1. 30 *Node* Tanpa Serangan *Rushing*



Gambar 5.1 30 *Node* Tanpa Serangan *Rushing*

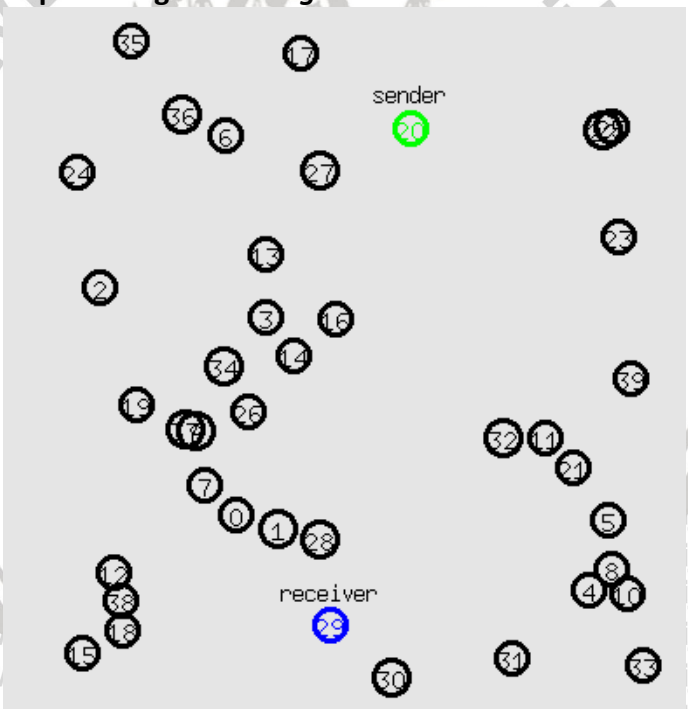
Pada gambar 5.1 adalah implementasi AODV dengan 30 *node* tanpa serangan *rushing* dengan pergerakan *Random Way Point*. Bisa dilihat dalam gambar diatas *source node* (*node* 4) yang berwarna hijau dan label *sender* dan *node receiver* (*node* 18) yang berwarna biru dan label *receiver.*

```
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$ ns rushingattacks_30.tcl
num_nodes is set 30
INITIALIZE THE LIST xListHead
Loading scenario file...
channel.cc:sendUp - Calc highestAntennaZ_ and distCST_
highestAntennaZ_ = 1.5,  distCST_ = 550.0
SORTING LISTS ...DONE!
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$ awk -f parameter.awk out30.tr
Packet Sent :1209
Packet Received :1207
Packet Loss :2
Packet Loss :0.165426 %
Packet Delivery Ratio :99.8346 %
Average Throughput[kbps] = 101.86       StartTime=1.00  StopTime = 99.97
End-to-End Delay = 31.9341 ms
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$
```

Gambar 5.2 Hasil Implementasi 30 *Node* Tanpa Serangan *Rushing*

Pada gambar 5.2 merupakan hasil dari implementasi dari protokol AODV tanpa serangan *rushing* dengan 30 *node* yang menunjukkan dalam proses pengiriman paket dari *source node* (*node* 4) menuju ke *node receiver* (*node* 18) *Packet loss* 8,35401 %, *Packet Delivery Ratio* 91,646 %, *Throughput* 93,48 kbps, dan *End to end delay* 417,453 ms .

2. 40 *Node* Tanpa Serangan *Rushing*



Gambar 5.3 40 *Node* Tanpa Serangan *Rushing*

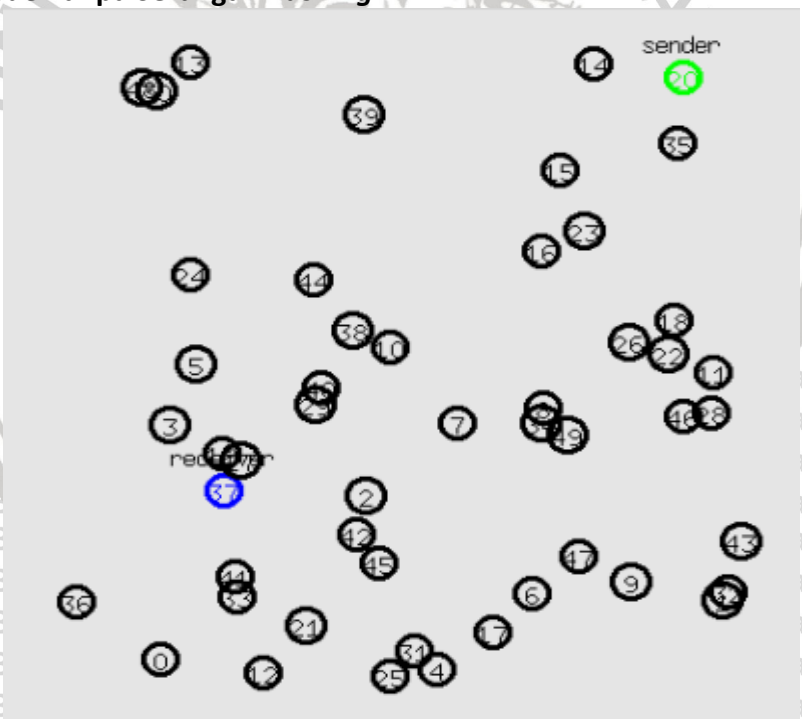
Pada gambar 5.3 merupakan implementasi AODV dengan 40 *node* tanpa serangan *rushing* dengan pergerakan *Random Way Point*. Bisa dilihat pada gambar diatas *source node* (*node* 20) yang berwarna hijau dan label *sender* dan *node receiver* (*node* 29) yang berwarna biru dan label *receiver.*

```
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$ ns rushingattacks_40.tcl
num_nodes is set 40
INITIALIZE THE LIST xListHead
Loading scenario file...
channel.cc:sendUp - Calc highestAntennaZ_ and distCST_
highestAntennaZ_ = 1.5,  distCST_ = 550.0
SORTING LISTS ...DONE!
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$ awk -f parameter.awk out40.tr
Packet Sent :1209
Packet Received :867
Packet Loss :342
Packet Loss :28.2878 %
Packet Delivery Ratio :71.7122 %
Average Throughput[kbps] = 101.56       StartTime=1.00  StopTime = 72.30
End-to-End Delay = 19.5681 ms
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$
```

Gambar 5.4 Hasil Implementasi 40 *Node* Tanpa Serangan *Rushing*

Pada gambar 5.4 merupakan hasil implementasi dengan menggunakan protokol AODV tanpa serangan *rushing* dengan 40 *node* yang menunjukkan dalam proses pengiriman paket dari *source node* (*node* 20) menuju ke *node receiver* (*node* 29) *Packet loss* 2,97767 %, *Packet Delivery Ratio* 97,0223 %, *Throughput* 98,96 kbps, dan *End to end delay* 60,8227 ms .

3. 50 *Node* Tanpa Serangan *Rushing*



Gambar 5.5 50 *Node* Tanpa Serangan *Rushing*

Pada gambar 5.5 merupakan implementasi AODV dengan 50 *node* tanpa serangan *rushing* dengan pergerakan *Random Way Point*. Bisa dilihat pada gambar diatas *source node* (*node* 20) yang berwarna hijau dan label *sender* dan *node receiver* (*node* 37) yang berwarna biru dan label *receiver.*

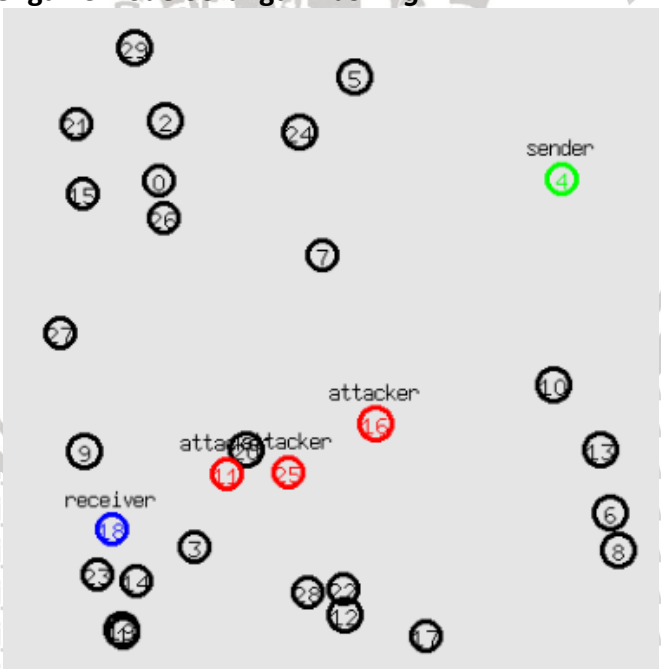
```
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$ ns rushingattacks_50.tcl
num_nodes is set 50
INITIALIZE THE LIST xListHead
Loading scenario file...
channel.cc:sendUp - Calc highestAntennaZ_ and distCST_
highestAntennaZ_ = 1.5,  distCST_ = 550.0
SORTING LISTS ...DONE!
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$ awk -f parameter.awk out50.tr
Packet Sent :1209
Packet Received :1141
Packet Loss :68
Packet Loss :5.62448 %
Packet Delivery Ratio :94.3755 %
Average Throughput[kbps] = 96.17        StartTime=1.00  StopTime = 100.09
End-to-End Delay = 356.205 ms
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$
```

Gambar 5.6 Hasil Implementasi 50 *Node* Tanpa Serangan *Rushing*

Pada gambar 5.6 merupakan hasil implementasi dengan menggunakan protokol AODV tanpa serangan *rushing* dengan 50 *node* yang menunjukkan dalam proses pengiriman paket dari *source node* (*node* 20) menuju ke *node receiver* (*node* 37) *Packet loss* 2,89495 %, *Packet Delivery Ratio* 97,105 %, *Throughput* 99,06 kbps, dan *End to end delay* 57,8196 ms.

5.2 Implementasi Dengan Serangan *Rushing*

1. 30 *Node* dengan 3 *Node* Serangan *Rushing*



Gambar 5.7 30 *Node* dengan Jumlah Serangan *Rushing* 3 *Node*

Pada gambar 5.7 merupakan implementasi dari AODV dengan 30 *node* dan jumlah serangan *rushing* 3 *node* dengan pergerakan *Random Way Point*. Bisa dilihat pada gambar diatas *source node* (*node* 4) yang berwarna hijau dan label

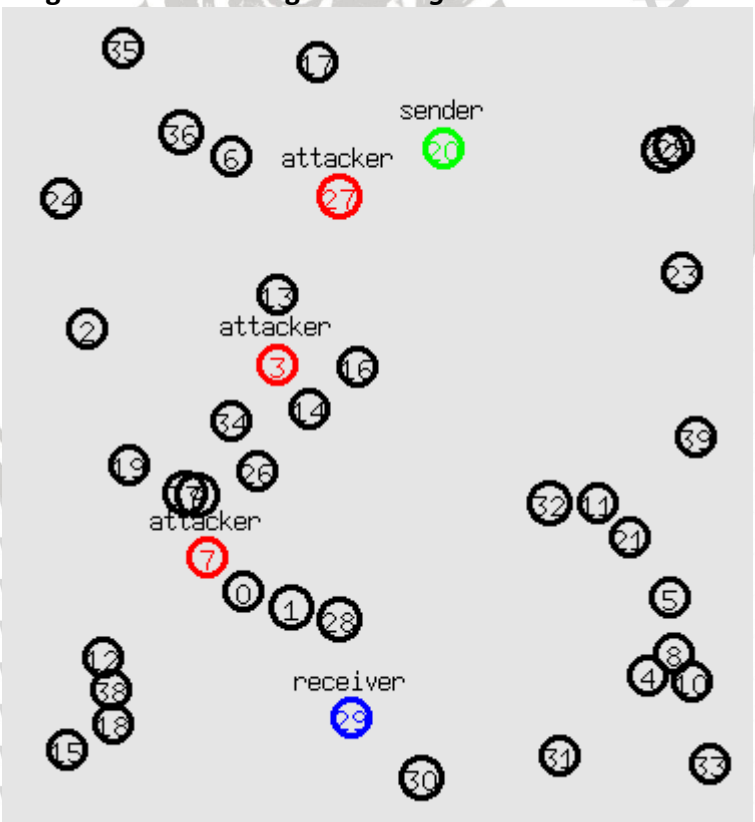
sender dan *node receiver* (*node* 18) yang berwarna biru dan label *receiver* dan juga *node* serangan *rushing* (*node* 11, *node* 25, *node* 16) yang berwarna merah dan label *attacker*.

```
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$ ns rushingattacks_30.tcl
num_nodes is set 30
INITIALIZE THE LIST xListHead
Loading scenario file...
channel.cc:sendUp - Calc highestAntennaZ_ and distCST_
highestAntennaZ_ = 1.5,  distCST_ = 550.0
SORTING LISTS ...DONE!
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$ awk -f parameter.awk out30.tr
Packet Sent :1209
Packet Received :1195
Packet Loss :14
Packet Loss :1.15798 %
Packet Delivery Ratio :98.842 %
Average Throughput[kbps] = 100.85       StartTime=1.00  StopTime = 99.97
End-to-End Delay = 790.375 ms
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$
```

Gambar 5.8 Hasil Implementasi 3 Serangan *Rushing* pada 30 *Node*

Pada gambar 5.8 merupakan hasil implementasi protokol AODV dengan jumlah 30 *node* dan jumlah serangan *rushing* 3 *node* yang menunjukkan dalam proses pengiriman paket dari *source node* (*node* 4) menuju ke *node receiver* (*node* 18) *Packet loss* 28,7014 %, *Packet Delivery Ratio* 71,2986 %, *Throughput* 72,73 kbps, dan *End to end delay* 995,407 ms.

2. 40 *Node* dengan 3 *Node* Serangan *Rushing*



Gambar 5.9 40 *Node* dengan Jumlah Serangan *Rushing* 3 *Node*

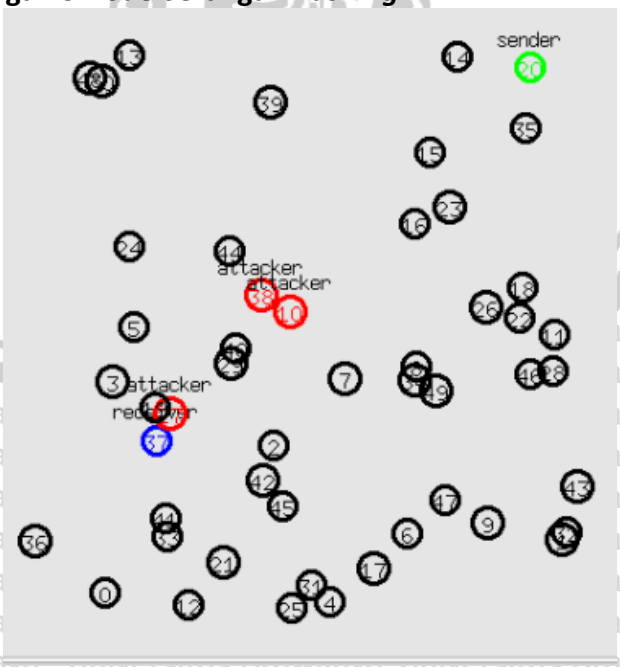
Pada gambar 5.9 merupakan implementasi AODV dengan 40 *node* dengan jumlah serangan *rushing* 3 *node* dengan pergerakan *Random Way Point*. Bisa dilihat pada gambar diatas *source node* (*node* 20) yang berwarna hijau dan label *sender* dan *node receiver* (*node* 29) yang berwarna biru dan label *receiver* dan juga *node* serangan *rushing* (*node* 3, *node* 7, *node* 27) yang berwarna merah dan label *attacker*.

```
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$ ns rushingattacks_40.tcl
num_nodes is set 40
INITIALIZE THE LIST xListHead
Loading scenario file...
channel.cc:sendUp - Calc highestAntennaZ_ and distCST_
highestAntennaZ_ = 1.5,  distCST_ = 550.0
SORTING LISTS ...DONE!
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$ awk -f parameter.awk out40.tr
Packet Sent :1209
Packet Received :867
Packet Loss :342
Packet Loss :28.2878 %
Packet Delivery Ratio :71.7122 %
Average Throughput[kbps] = 101.56       StartTime=1.00  StopTime = 72.30
End-to-End Delay = 52.1334 ms
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$
```

Gambar 5.10 Hasil Implementasi 3 Serangan *Rushing* pada 40 *Node*

Pada gambar 5.10 merupakan hasil implementasi protokol AODV dengan jumlah 40 *node* dan jumlah serangan *rushing* 3 *node* yang menunjukkan dalam proses pengiriman paket dari *source node* (*node* 20) menuju ke *node receiver* (*node* 29) *Packet loss* 10,67 %, *Packet Delivery Ratio* 89,33 %, *Throughput* 91,18 kbps, dan *End to end delay* 831,715 ms.

3. 50 *Node* dengan 3 *Node* Serangan *Rushing*



Gambar 5.11 50 *Node* dengan Jumlah Serangan *Rushing* 3 *Node*

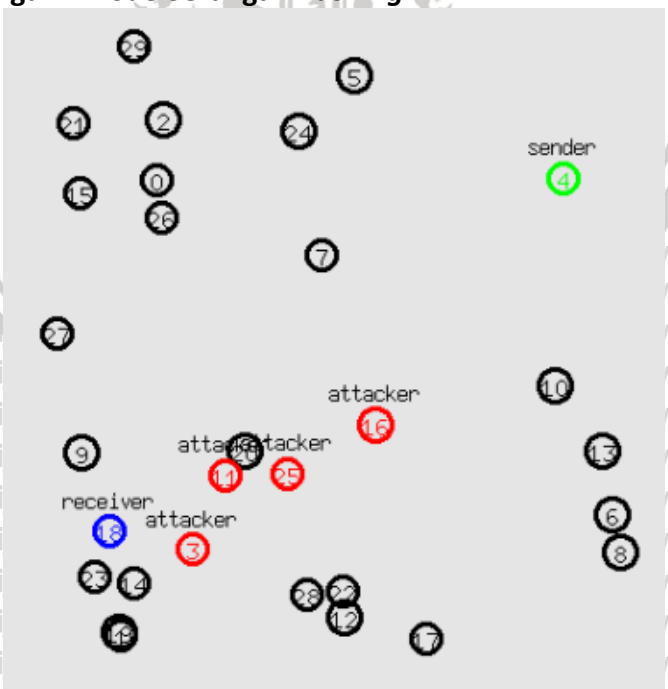
Pada gambar 5.11 merupakan implementasi AODV dengan 50 *node* dengan jumlah serangan *rushing* 3 *node* dengan pergerakan *Random Way Point*. Bisa dilihat pada gambar dibawah *source node* (*node* 20) yang berwarna hijau dan label *sender* dan *node receiver* (*node* 37) yang berwarna biru dan label *receiver* dan juga *node* serangan *rushing* (*node* 10, *node* 27, *node* 38) yang berwarna merah dan label *attacker*.

```
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$ ns rushingattacks_50.tcl
num_nodes is set 50
INITIALIZE THE LIST xListHead
Loading scenario file...
channel.cc:sendUp - Calc highestAntennaZ_ and distCST_
highestAntennaZ_ = 1.5,  distCST_ = 550.0
SORTING LISTS ...DONE!
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$ awk -f parameter.awk out50.tr
Packet Sent :1209
Packet Received :1120
Packet Loss :89
Packet Loss :7.36146 %
Packet Delivery Ratio :92.6385 %
Average Throughput[kbps] = 94.51        StartTime=1.00  StopTime = 99.98
End-to-End Delay = 135.305 ms
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$
```

Gambar 5.12 Hasil Implementasi 3 Serangan *Rushing* pada 50 *Node*

Pada gambar 5.12 merupakan hasil implementasi protokol AODV dengan jumlah 50 *node* dan jumlah serangan *rushing* 3 *node* yang menunjukkan dalam proses pengiriman paket dari *source node* (*node* 20) menuju ke *node receiver* (*node* 37) *Packet loss* 7,36146 %, *Packet Delivery Ratio* 92,6385 %, *Throughput* 94,51 kbps, dan *End to end delay* 135,305 ms.

4. 30 *Node* dengan 4 *Node* Serangan *Rushing*



Gambar 1.13 30 *Node* dengan Jumlah Serangan *Rushing* 4 *Node*

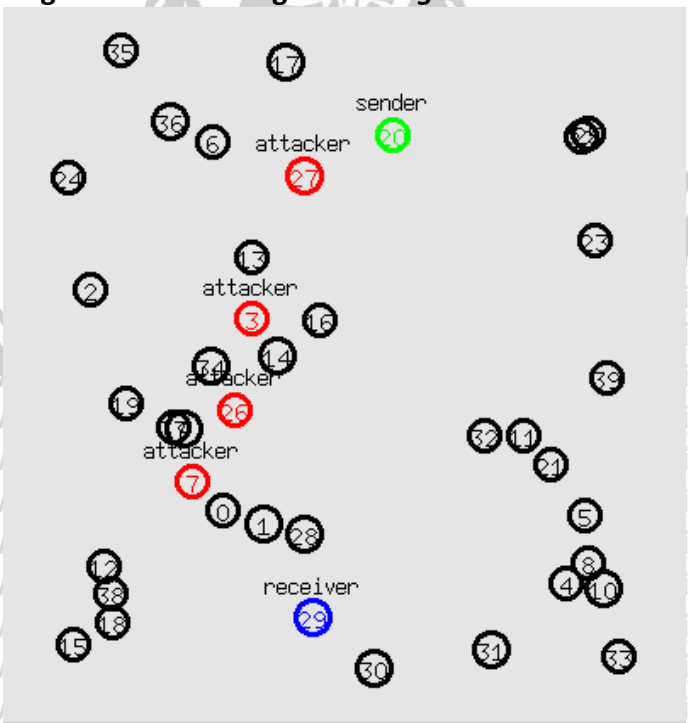
Pada gambar 5.13 merupakan implementasi AODV dengan 30 *node* dengan jumlah serangan *rushing* 4 *node* dengan pergerakan *Random Way Point*. Bisa dilihat pada gambar dibawah *source node* (*node* 4) yang berwarna hijau dan label *sender* dan *node receiver* (*node* 18) yang berwarna biru dan label *receiver* dan juga *node* serangan *rushing* (*node* 3, *node* 11, *node* 16, *node* 25) yang berwarna merah dan label *attacker*.

```
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$ ns rushingattacks_30.tcl
num_nodes is set 30
INITIALIZE THE LIST xListHead
Loading scenario file...
channel.cc:sendUp - Calc highestAntennaZ_ and distCST_
highestAntennaZ_ = 1.5,  distCST_ = 550.0
SORTING LISTS ...DONE!
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$ awk -f parameter.awk out30.tr
Packet Sent :1209
Packet Received :1105
Packet Loss :104
Packet Loss :8.60215 %
Packet Delivery Ratio :91.3978 %
Average Throughput[kbps] = 93.25        StartTime=1.00  StopTime = 99.97
End-to-End Delay = 4067.26 ms
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$
```

Gambar 5.14 Hasil Implementasi 4 Serangan *Rushing* pada 30 *Node*

Pada gambar 5.14 merupakan hasil implementasi protokol AODV dengan jumlah 30 *node* dan jumlah serangan *rushing* 4 *node* yang menunjukkan dalam proses pengiriman paket dari *source node* (*node* 4) menuju ke *node receiver* (*node* 18) *Packet loss* 14,2266 %, *Packet Delivery Ratio* 85,7734 %, *Throughput* 90,49 kbps, dan *End to end delay* 1146,09 ms.

5. 40 *Node* dengan 4 *Node* Serangan *Rushing*



Gambar 5.15 40 *Node* dengan Jumlah Serangan *Rushing* 4 *Node*

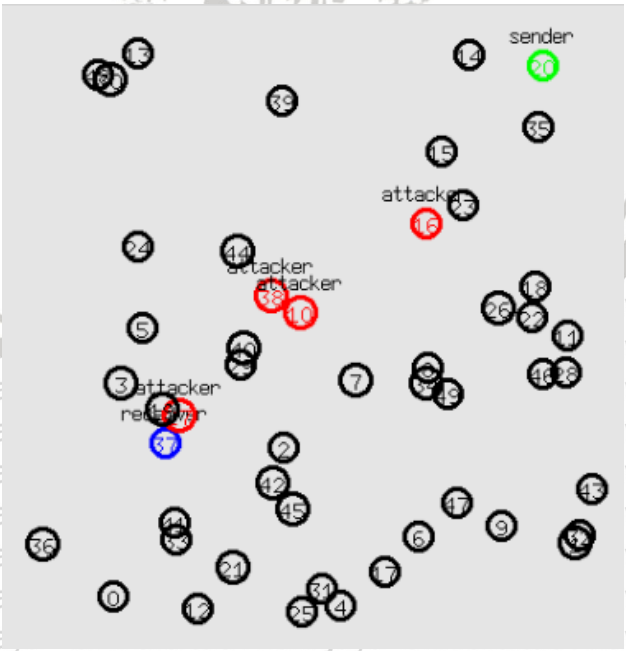
Pada gambar 5.15 merupakan implementasi AODV dengan 40 *node* dengan jumlah serangan *rushing* 4 *node* dengan pergerakan *Random Way Point*. Bisa dilihat pada gambar dibawah *source node* (*node* 20) yang berwarna hijau dan label *sender* dan *node receiver* (*node* 29) yang berwarna biru dan label *receiver* dan juga *node* serangan *rushing* (*node* 3, *node* 7, *node* 26, *node* 27) yang berwarna merah dan label *attacker*.

```
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$ ns rushingattacks_40.tcl
num_nodes is set 40
INITIALIZE THE LIST xListHead
Loading scenario file...
channel.cc:sendUp - Calc highestAntennaZ_ and distCST_
highestAntennaZ_ = 1.5,  distCST_ = 550.0
SORTING LISTS ...DONE!
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$ awk -f parameter.awk out40.tr
Packet Sent :1209
Packet Received :711
Packet Loss :498
Packet Loss :41.1911 %
Packet Delivery Ratio :58.8089 %
Average Throughput[kbps] = 83.29        StartTime=1.00  StopTime = 72.30
End-to-End Delay = 256.856 ms
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$
```

Gambar 5.16 Hasil Implementasi 4 Serangan *Rushing* pada 40 *Node*

Pada gambar 5.16 merupakan hasil implementasi protokol AODV dengan jumlah 40 *node* dan jumlah serangan *rushing* 4 *node* yang menunjukkan dalam proses pengiriman paket dari *source node* (*node* 20) menuju ke *node receiver* (*node* 29) *Packet loss* 23,0769 %, *Packet Delivery Ratio* 76,9231 %, *Throughput* 78,46 kbps, dan *End to end delay* 365,731 ms.

6. 50 *Node* dengan 4 *Node* Serangan *Rushing*



Gambar 5.17 50 *Node* dengan Jumlah Serangan *Rushing* 4 *Node*

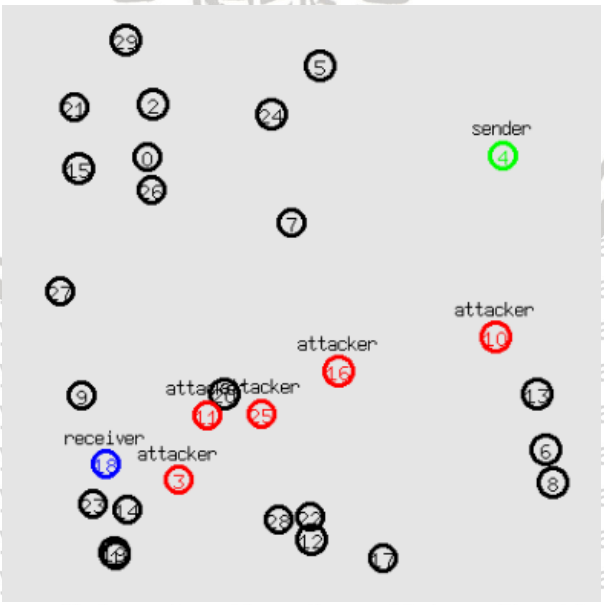
Pada gambar 5.17 merupakan implementasi AODV dengan 50 *node* dengan jumlah serangan *rushing* 4 *node* dengan pergerakan *Random Way Point*. Bisa dilihat pada gambar bawah *source node* (*node* 20) yang berwarna hijau dan label *sender* dan *node receiver* (*node* 37) yang berwarna biru dan label *receiver* dan juga *node* serangan *rushing* (*node* 10, *node* 16, *node* 27, *node* 38) yang berwarna merah dan label *attacker*.

```
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$ ns rushingattacks_50.tcl
num_nodes is set 50
INITIALIZE THE LIST xListHead
Loading scenario file...
channel.cc:sendUp - Calc highestAntennaZ_ and distCST_
highestAntennaZ_ = 1.5,  distCST_ = 550.0
SORTING LISTS ...DONE!
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$ awk -f parameter.awk out50.tr
Packet Sent :1209
Packet Received :1016
Packet Loss :193
Packet Loss :15.9636 %
Packet Delivery Ratio :84.0364 %
Average Throughput[kbps] = 81.59        StartTime=1.00  StopTime = 105.00
End-to-End Delay = 1265.02 ms
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$
```

Gambar 5.18 Hasil Implementasi 4 Serangan *Rushing* pada 50 *Node*

Pada gambar 5.18 merupakan hasil implementasi protokol AODV dengan jumlah 50 *node* dan jumlah serangan *rushing* 4 *node* yang menunjukkan dalam proses pengiriman paket dari *source node* (*node* 20) menuju ke *node receiver* (*node* 37) *Packet loss* 18,6931 %, *Packet Delivery Ratio* 81,3069 %, *Throughput* 82,95 kbps, dan *End to end delay* 192,181 ms.

7. 30 *Node* dengan 5 *Node* Serangan *Rushing*



Gambar 5.20 30 *Node* dengan Jumlah Serangan *Rushing* 5 *Node*

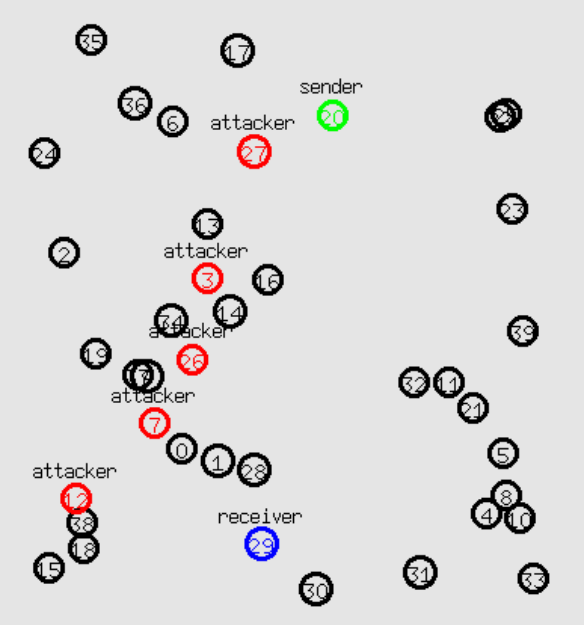
Pada gambar 5.20 merupakan implementasi AODV dengan 30 *node* dengan jumlah serangan *rushing* 5 *node* dengan pergerakan *Random Way Point*. Bisa dilihat pada gambar dibawah *source node* (*node* 4) yang berwarna hijau dan label *sender* dan *node receiver* (*node* 18) yang berwarna biru dan label *receiver* dan juga *node* serangan *rushing* (*node* 3, *node* 10, *node* 11, *node* 16, dan *node* 25) yang berwarna merah dan label *attacker*.

```
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$ ns rushingattacks_30.tcl
num_nodes is set 30
INITIALIZE THE LIST xListHead
Loading scenario file...
channel.cc:sendUp - Calc highestAntennaZ_ and distCST_
highestAntennaZ_ = 1.5,  distCST_ = 550.0
SORTING LISTS ...DONE!
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$ awk -f parameter.awk out30.tr
Packet Sent :1209
Packet Received :1138
Packet Loss :71
Packet Loss :5.87262 %
Packet Delivery Ratio :94.1274 %
Average Throughput[kbps] = 96.04        StartTime=1.00  StopTime = 99.97
End-to-End Delay = 772.44 ms
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$
```

Gambar 5.21 Hasil Implementasi 5 Serangan *Rushing* pada 30 *Node*

Pada gambar 5.21 merupakan hasil implementasi protokol AODV dengan jumlah 30 *node* dan jumlah serangan *rushing* 5 *node* yang menunjukkan dalam proses pengiriman paket dari *source node* (*node* 4) menuju ke *node receiver* (*node* 18) *Packet loss* 23,1596 %, *Packet Delivery Ratio* 76,8404 %, *Throughput* 78,51 kbps, dan *End to end delay* 826,801 ms.

8. 40 *Node* dengan 5 *Node* Serangan *Rushing*



Gambar 2.22 40 *Node* dengan Jumlah Serangan *Rushing* 5 *Node*

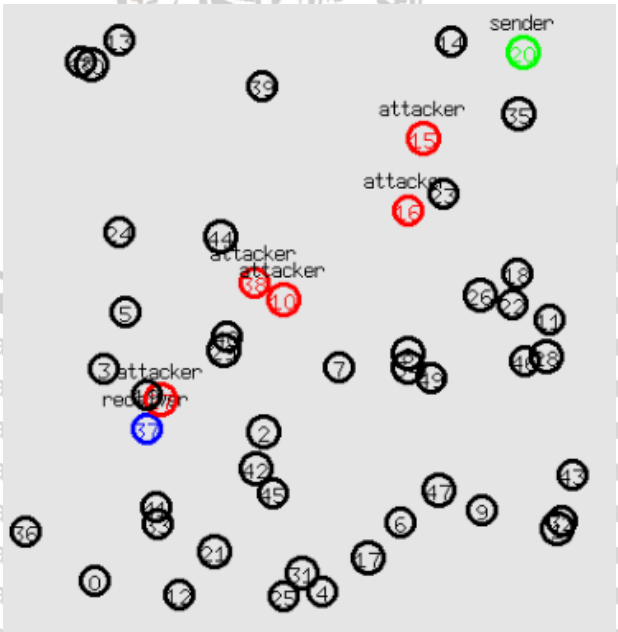
Pada gambar 5.22 merupakan implementasi AODV dengan 40 *node* dengan jumlah serangan *rushing* 5 *node* dengan pergerakan *Random Way Point*. Bisa dilihat pada gambar dibawah *source node* (*node* 20) yang berwarna hijau dan label *sender* dan *node receiver* (*node* 29) yang berwarna biru dan label *receiver* dan juga *node* serangan *rushing* (*node* 3, *node* 7, *node* 12, *node* 26, dan *node* 27) yang berwarna merah dan label *attacker*.

```
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$ ns rushingattacks_40.tcl
num_nodes is set 40
INITIALIZE THE LIST xListHead
Loading scenario file...
channel.cc:sendUp - Calc highestAntennaZ_ and distCST_
highestAntennaZ_ = 1.5,  distCST_ = 550.0
SORTING LISTS ...DONE!
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$ awk -f parameter.awk out40.tr
Packet Sent :1209
Packet Received :711
Packet Loss :498
Packet Loss :41.1911 %
Packet Delivery Ratio :58.8089 %
Average Throughput[kbps] = 83.29        StartTime=1.00  StopTime = 72.30
End-to-End Delay = 256.856 ms
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$
```

Gambar 5.23 Hasil Implementasi 5 Serangan *Rushing* pada 40 *Node*

Pada gambar 5.23 merupakan hasil implementasi protokol AODV dengan jumlah 40 *node* dan jumlah serangan *rushing* 5 *node* yang menunjukkan dalam proses pengiriman paket dari *source node* (*node* 4) menuju ke *node receiver* (*node* 18) *Packet loss* 11,3317 %, *Packet Delivery Ratio* 88,6683 %, *Throughput* 90,51 kbps, dan *End to end delay* 107,147 ms.

9. 50 *Node* dengan 5 *Node* Serangan *Rushing*



Gambar 5.24 50 *Node* dengan Jumlah Serangan *Rushing* 5 *Node*

Pada gambar 5.24 merupakan implementasi AODV dengan 50 *node* dengan jumlah serangan *rushing* 5 *node* dengan pergerakan *Random Way Point*. Bisa dilihat pada gambar dibawah *source node* (*node* 20) yang berwarna hijau dan label *sender* dan *node receiver* (*node* 37) berwarna biru dan label *receiver* dan juga *node* serangan *rushing* (*node* 10, *node* 15, *node* 16, *node* 27, dan *node* 38) yang berwarna merah dan label *attacker*.

```
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$ ns rushingattacks_50.tcl
num_nodes is set 50
INITIALIZE THE LIST xListHead
Loading scenario file...
channel.cc:sendUp - Calc highestAntennaZ_ and distCST_
highestAntennaZ_ = 1.5,  distCST_ = 550.0
SORTING LISTS ...DONE!
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$ awk -f parameter.awk out50.tr
Packet Sent :1209
Packet Received :1073
Packet Loss :136
Packet Loss :11.249 %
Packet Delivery Ratio :88.751 %
Average Throughput[kbps] = 90.53        StartTime=1.00  StopTime = 99.99
End-to-End Delay = 1337.27 ms
eldyto@eldyto-VirtualBox:~/ns-allinone-2.35/ns-2.35/rushing attack$
```

Gambar 5.25 Hasil Implementasi 5 Serangan *Rushing* pada 50 *Node*

Pada gambar 5.25 merupakan hasil implementasi protokol AODV dengan jumlah 50 *node* dan jumlah serangan *rushing* 5 *node* yang menunjukkan dalam proses pengiriman paket dari *source node* (*node* 4) menuju ke *node receiver* (*node* 18) *Packet loss* 12,8205 %, *Packet Delivery Ratio* 87,1795 %, *Throughput* 88,93 kbps, dan *End to end delay* 145,563 ms.

BAB 6 HASIL DAN ANALISIS

Bagian ini berisi mengenai hasil dari implementasi yang sudah dilakukan. implementasi yang sudah dilakukan adalah protokol AODV tanpa serangan *rushing* dengan 30, 40, dan 50 *node* dan protokol AODV dengan serangan *rushing* 3, 4, dan 5 *node* dengan keadaan *node* serangan *rushing* diam dan bergerak dengan 30, 40, dan 50 *node* kemudian hasil dari implementasi tersebut akan dilakukan pengukuran dengan menggunakan parameter *throughput, end to end delay, packet delivery ratio,* dan *packet loss.*

6.1 Hasil

Hasil yang sudah didapatkan dari implementasi AODV tanpa serangan *rushing* dengan 30, 40, dan 50 *node* dan protokol AODV dengan serangan *rushing* 3, 4, dan 5 *node* dengan keadaan *node* serangan *rushing* diam dan bergerak dengan 30, 40, dan 50 akan dilakukan pengukuran dengan menggunakan parameter *throughput, end to end delay, packet delivery ratio,* dan *packet loss*. Berikut merupakan hasil pengujian dengan menggunakan parameter pengujian *packet delivery ratio, packet loss throughput* dan *end to end delay* pada protokol AODV tanpa serangan *rushing* dan dengan serangan *rushing*. Dimana, *packet delivery Ratio* adalah rasio antara bayaknya paket yang diterima dengan paket yang dikirim, *packet loss* terjadi karena kegagalan transmisi paket untuk mencapai tujuannya, *throughput* adalah banyaknya *byte* yang diteruma dalam waktu tertentu dengan satuan *byte/second* atau dengan kata lain kondisi data *rate* sebenarnya dalam suatu jaringan dan *end to end delay* merupakan waktu yang digunakan ketika paket dikirimkan hingga paket diterima.

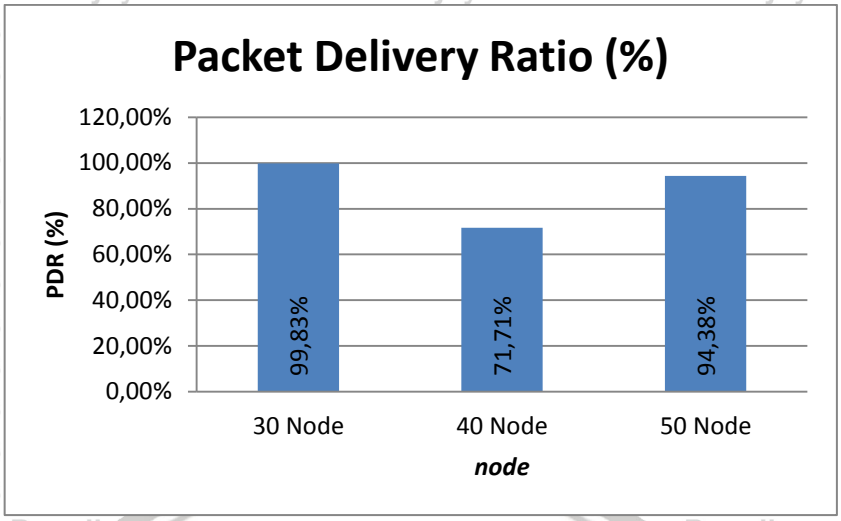
6.1.1 Hasil Implementasi Tanpa Serangan *rushing*

Berikut adalah hasil implementasi tanpa serangan *rushing* dengan jumlah *node* 30, 40, dan 50 *node* dengan parameter ukur *packet delivery ratio, packet loss, throughput, dan end to end delay* yang ditunjukkan pada tabel 6.1.

Tabel 6.1 Hasil Implementasi Tanpa Serangan *Rushing*

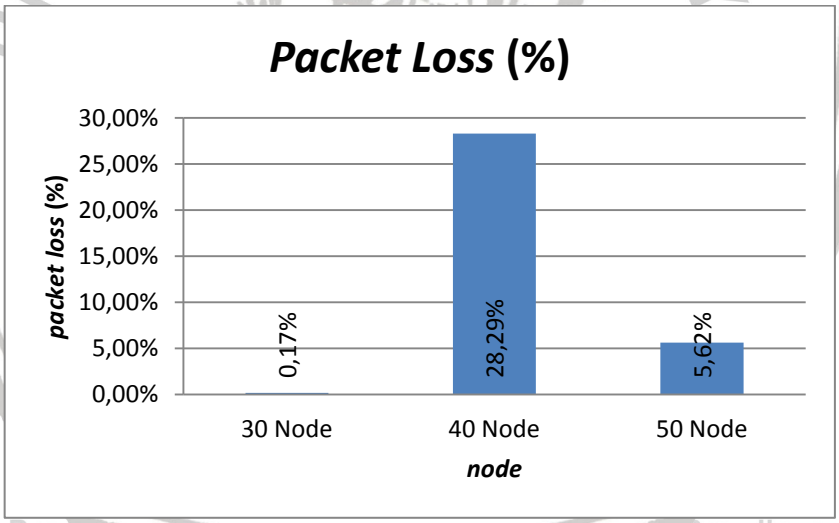
Jumlah *Node*	Parameter Ukur			
	Packet Delivery Ratio (%)	*Packet Loss* (%)	*Throughput* (kbps)	*End to End Delay* (ms)
30 *Node*	99,8346 %	0,1654 %	101,86 kbps	31,9341 ms
40 *Node*	71,7122 %	28,2878 %	101,56 kbps	19,5681 ms
50 *Node*	88,751 %	11,249 %	92,80 kbps	356,205 ms

Pada tabel 6.1 menunjukkan hasil implementasi tanpa serangan *rushing* dari 30 *node,* 40 *node,* dan 50 *node*.



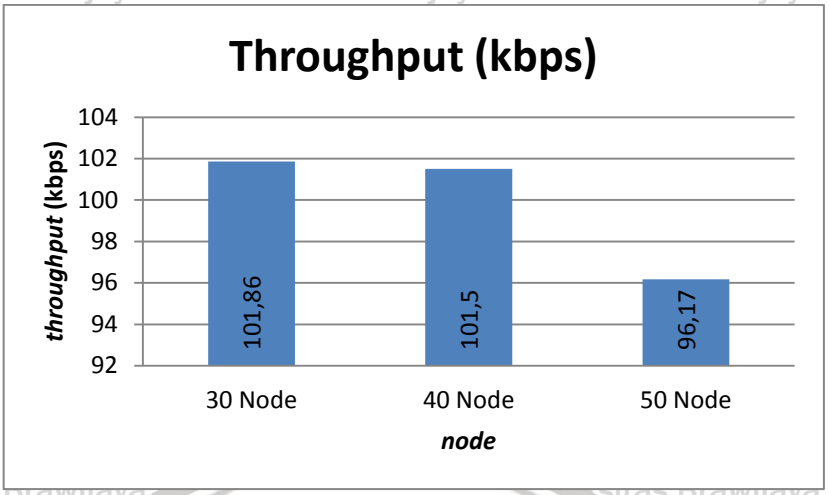
Gambar 6.1 Grafik *Packet Delivery Ratio* Tanpa Serangan *Rushing*

Pada gambar 6.1 menunjukkan hasil *packet delivery ratio* (PDR) dari hasil implementasi tanpa serangan *rushing*. Dapat dilihat dari grafik 30 *node* dan 50 *node* memiliki nilai sebesar 99,83 % dan 94,38 %, sedangkan 40 *node* memiliki nilai yang lebih rendah yaitu sebesar 71,71 %. Semakin tinggi nilai PDR yang dihasilkan maka akan semakin baik pula performa jaringannya



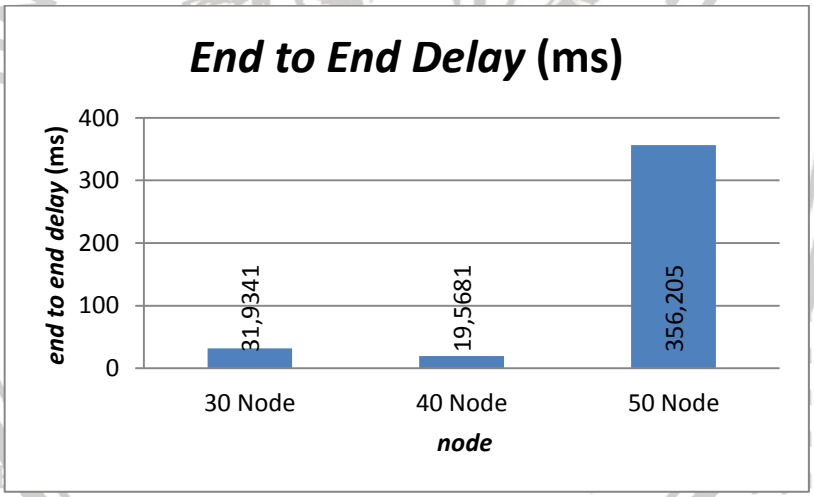
Gambar 6.2 Grafik *Packet Loss* Tanpa Serangan *Rushing*

Pada gambar 6.2 menunjukkan hasil *packet loss* dari hasil implementasi tanpa serangan *rushing*. Dapat dilihat dari grafik 30 *node* dan 50 *node* memiliki nilai 0,1 % dan 5,62 % sedangkan 40 *node* memiliki nilai yang lebih tinggi yaitu sebesar 28,29 %. Semakin rendah nilai *packet loss* maka semakin baik pula performa jaringannya



Gambar 6.3 Grafik *Throughput* Tanpa Serangan *Rushing*

Pada gambar 6.3 menunjukkan hasil *throughput* dari hasil implementasi tanpa serangan *rushing*. Dapat dilihat dari grafik 30 *node* dan 40 *node* memiliki nilai 101.86 kbps dan 101,5 kbps sedangkan 50 *node* memiliki nilai yang lebih rendah yaitu sebesar 966,17 kbps. Semakin tinggi nilai *throughput* maka semakin baik pula performa jaringannya



Gambar 6.4 Grafik *End to End Delay* Tanpa Serangan *Rushing*

Pada gambar 6.4 menunjukkan hasil *end to end delay* dari hasil implementasi tanpa serangan *rushing*. Dapat dilihat dari grafik *node* 30 dan 40 *node* memiliki nilai 31,9341 ms dan 19,5681 ms sedangkan *node* 50 memiliki nilai *end to end delay* yang tinggi yaitu sebesar 366,205 ms. Semakin rendah nilai *end to end delay* maka semakin baik pula performa jaringannya.

6.1.2 Hasil Implementasi Dengan Serangan *rushing*

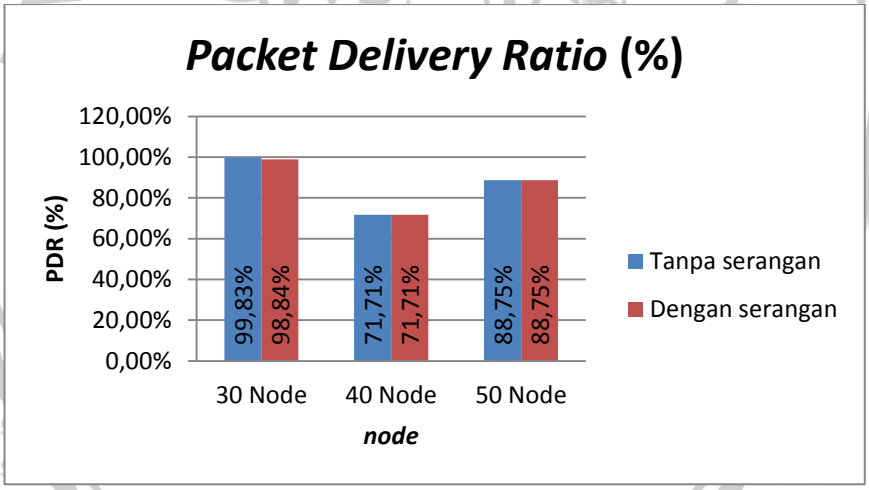
1. Hasil Implementasi Serangan *rushing* dengan Jumlah 3 *Node* Serangan

Berikut adalah hasil implementasi 3 serangan *rushing* dengan keadaan diam dan bergerak dengan jumlah *node* 30, 40, dan 50 *node* dengan parameter ukur *packet delivery ratio, packet loss, throughput, dan end to end delay* yang ditunjukkan pada tabel 6.2.

Tabel 6.2 Hasil Implementasi 3 *Node* Serangan *Rushing*

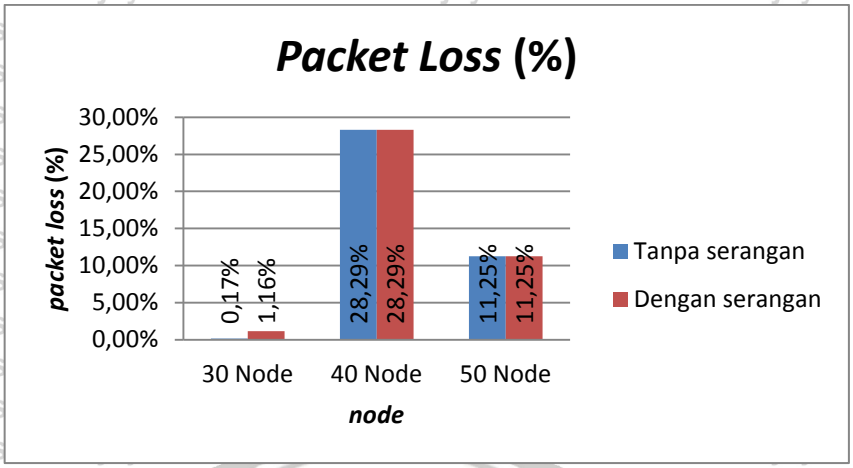
Jumlah *Node*	Parameter Ukur			
	Packet Delivery Ratio (%)	*Packet Loss* (%)	*Throughput* (kbps)	*End to End Delay* (ms)
30 *Node*	98,842 %	1,15798 %	100,85 kbps	790,375 ms
40 *Node*	71,7122 %	28,2878 %	101,56 kbps	52,1334 ms
50 *Node*	88,751 %	11,249 %	92,80 kbps	377,791 ms

Pada tabel 6.2 menunjukkan hasil implementasi 3 serangan *rushing* dari 30 *node,* 40 *node,* dan 50 *node*.



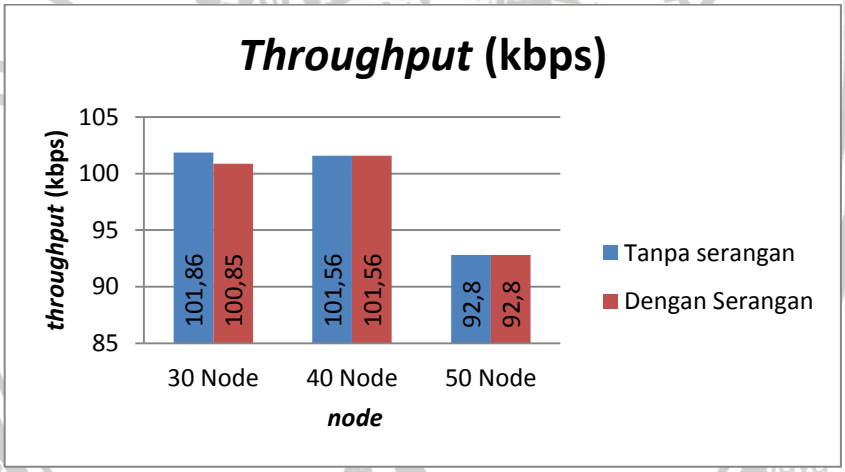
Gambar 6.5 Grafik *Packet Delivery Ratio* dari 3 *Node* Serangan *Rushing*

Pada gambar 6.5 menunjukkan hasil *packet delivery ratio* (PDR) dari hasil implementasi serangan *rushing* dengan jumlah 3 *node*. Dapat dilihat dari grafik terdapat penurunan nilai *packet delivery ratio* (PDR) dibandingkan dengan implementasi tanpa serangan. Pada 30 *node* terjadi penurunan sebanyak 0,99 %, 40 *node* dan 50 *node* tidak terjadi penurunan PDR.



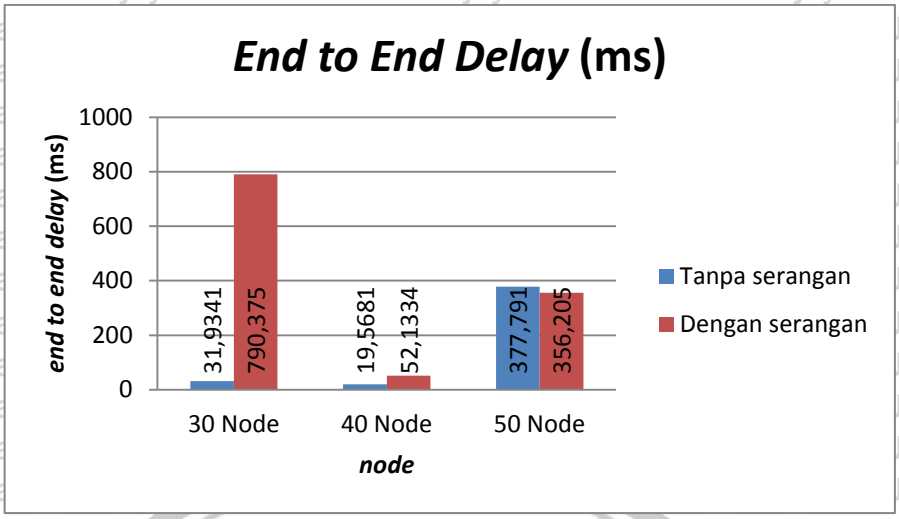
Gambar 6.6 Grafik *Packet Loss* dari 3 *Node* Serangan *Rushing*

Pada gambar 6.6 menunjukkan hasil *packet loss* dari hasil implementasi serangan *rushing* dengan jumlah 3 *node*. Dapat dilihat dari grafik terdapat penaikan nilai *packet loss* dibandingkan dengan implementasi tanpa serangan. Pada 30 *node* terjadi kenaikan sebanyak 0,99 %, 40 *node* dan 50 *node* tidak terjadi kenaikan.



Gambar 6.7 Grafik *Throughput* dari 3 *Node* Serangan *Rushing*

Pada gambar 6.7 menunjukkan hasil *throughput* dari hasil implementasi serangan *rushing* dengan jumlah 3 *node*. Dapat dilihat dari grafik terdapat penurunan nilai *throughput* dibandingkan dengan implementasi tanpa serangan. Pada 30 *node* terjadi penurunan sebanyak 1,01 kbps, 40 *node* dan 50 *node* tidak terjadi penurunan.



Gambar 6.8 Grafik *End to End Delay* dari 3 *Node* Serangan *Rushing*

Pada gambar 6.8 menunjukkan hasil *end to end delay* dari hasil implementasi serangan *rushing* dengan jumlah 3 *node*. Dapat dilihat dari grafik terdapat kenaikan nilai *end to end delay* dibandingkan dengan implementasi tanpa serangan. Pada 30 *node* terjadi kenaikan sebanyak 758,4409 ms, 40 *node* terjadi kenurunan sebanyak 32,5653 ms, dan 50 *node* terjadi penurunan sebanyak 21,586 ms. Penaikan nilai *end to end delay* pada implementasi 3 serangan *rushing* paling besar terdapat pada 30 *node* dengan penaikan sebanyak 758,4409 ms.

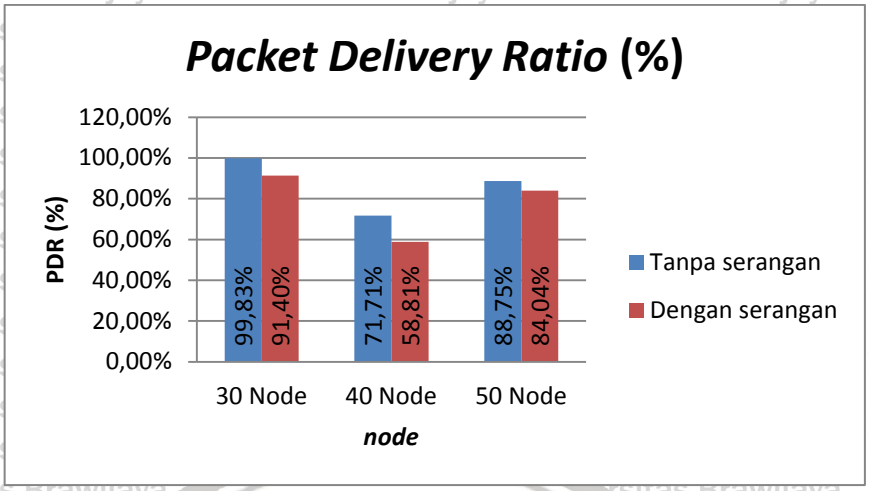
2. Hasil Implementasi Serangan *rushing* dengan Jumlah 4 *Node* Serangan

Berikut adalah hasil implementasi 4 serangan *rushing* dengan jumlah *node* 30, 40, dan 50 *node* dengan parameter ukur *packet delivery ratio, packet loss, throughput, dan end to end delay* yang ditunjukkan pada tabel 6.3.

Tabel 6.3 Hasil Implementasi 4 *Node* Serangan *Rushing*

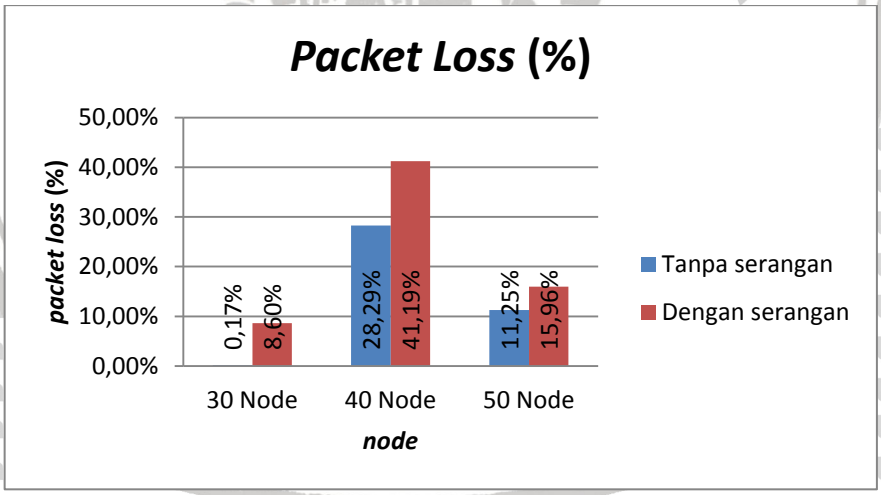
Jumlah *Node*	Parameter Ukur			
	Packet Delivery Ratio (%)	*Packet Loss* (%)	*Throughput* (kbps)	*End to End Delay* (ms)
30 *Node*	91,3978 %	8,60215 %	93,25 kbps	4067,26 ms
40 *Node*	58,8089 %	41,1911 %	83,29 kbps	256,856 ms
50 *Node*	84,0364 %	15,9636 %	81,59 kbps	1265,02 ms

Pada tabel 6.3 menunjukkan hasil implementasi 4 serangan *rushing* dari 30 *node,* 40 *node,* dan 50 *node*.



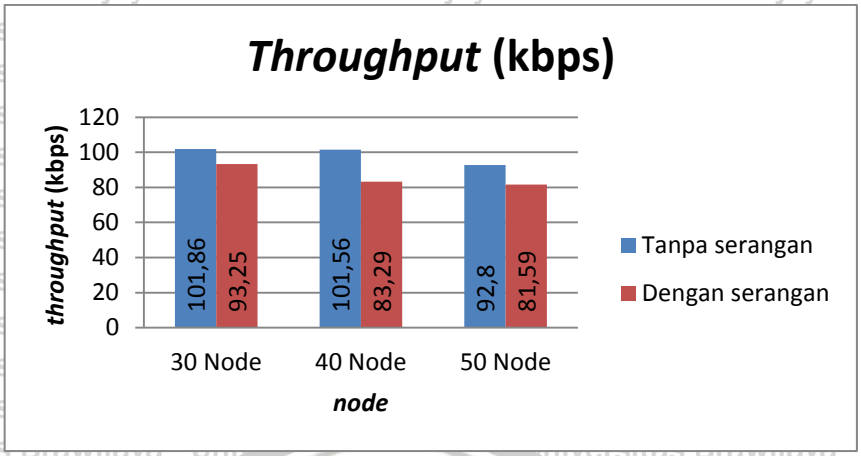
Gambar 6.9 Grafik *Packet Delivery Ratio* dari 4 *Node* Serangan *Rushing*

Pada gambar 6.9 menunjukkan hasil *packet delivery ratio* (PDR) dari hasil implementasi serangan *rushing* dengan jumlah 4 *node*. Dapat dilihat dari grafik terdapat penurunan nilai *packet delivery ratio* (PDR) dibandingkan dengan implementasi tanpa serangan. Pada 30 *node* terjadi penurunan sebanyak 8,44%, 40 *node* terjadi penurunan sebanyak 12,90%, dan 50 *node* terjadi penurunan sebanyak 4,71%. Penurunan nilai PDR pada implementasi 4 serangan *rushing* paling besar terdapat pada 40 *node* dengan penurunan sebanyak 12,90%.



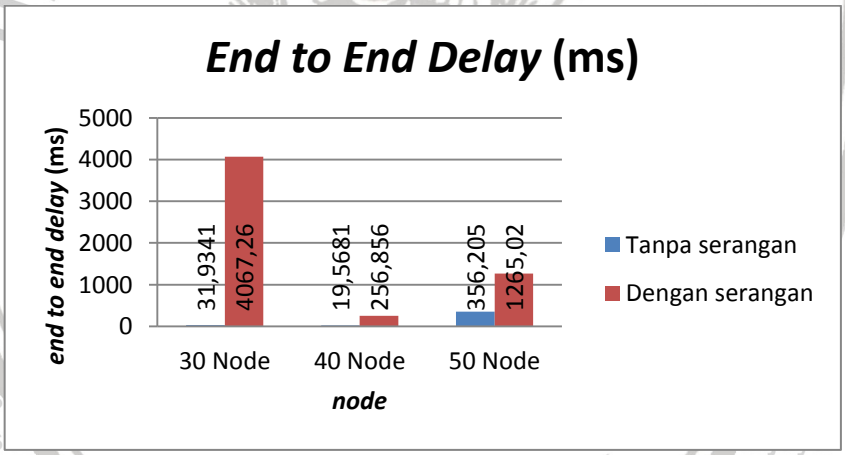
Gambar 6.10 Grafik *Packet Loss* dari 4 *Node* Serangan *Rushing*

Pada gambar 6.10 menunjukkan hasil *packet loss* dari hasil implementasi serangan *rushing* dengan jumlah 4 *node*. Dapat dilihat dari grafik terdapat kenaikan nilai *packet loss* dibandingkan dengan implementasi tanpa serangan. Pada 30 *node* terjadi penaikan sebanyak 8,44%, 40 *node* terjadi penaikan sebanyak 12,90%, dan 50 *node* terjadi penaikan sebanyak 4,71%. Penaikan nilai *packet loss* pada implementasi 4 serangan *rushing* paling besar terdapat pada 40 *node* dengan penaikan sebanyak 12,90%.



Gambar 6.11 Grafik *Throughput* dari Serangan 4 *Node* Serangan *Rushing*

Pada gambar 6.11 menunjukkan hasil *throughput* dari hasil implementasi serangan *rushing* dengan jumlah 4 *node*. Dapat dilihat dari grafik terdapat penurunan nilai *throughput* dibandingkan dengan implementasi tanpa serangan. Pada 30 *node* terjadi penurunan sebanyak 8,61 kbps, 40 *node* terjadi penurunan sebanyak 18,27 kbps, dan 50 *node* terjadi penurunan sebanyak 11,21 kbps. penurunan nilai *throughput* pada implementasi 4 serangan *rushing* paling besar terdapat pada 40 *node* dengan penaikan sebanyak 18,27 kbps.



Gambar 6.12 Grafik *End to End Delay* dari 4 *Node* Serangan *Rushing*

Pada gambar 6.12 menunjukkan hasil *end to end delay* dari hasil implementasi serangan *rushing* dengan jumlah 4 *node*. Dapat dilihat dari grafik terdapat kenaikan nilai *end to end delay* dibandingkan dengan implementasi tanpa serangan. Pada 30 *node* terjadi kenaikan sebanyak 4035,3259 ms, 40 *node* terjadi kenaikan sebanyak 237,2879 ms, dan 50 *node* terjadi kenaikan sebanyak 908,815 ms. Penaikan nilai *end to end delay* pada implementasi 4 serangan *rushing* paling besar terdapat pada 30 *node* dengan penaikan sebanyak 4035,3259 ms.

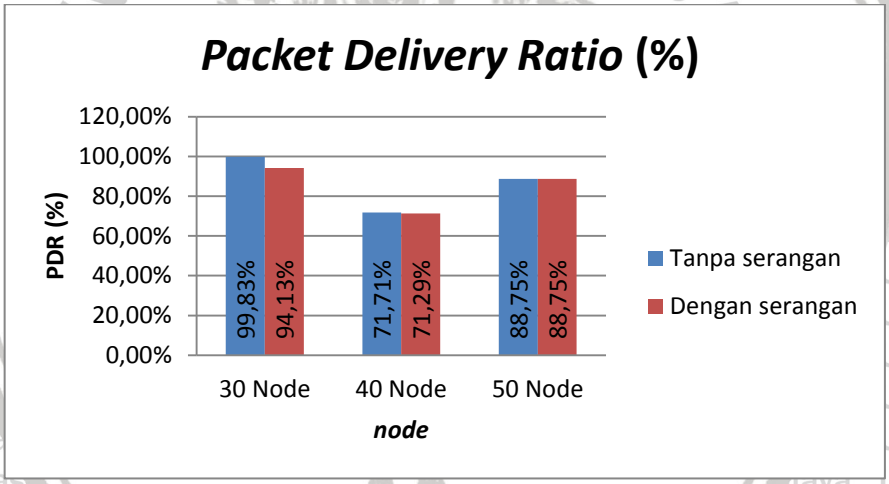
3. Hasil Implementasi Serangan *rushing* dengan Jumlah 5 *Node* Serangan

Berikut adalah hasil implementasi 5 serangan *rushing* dengan keadaan dengan jumlah *node* 30, 40, dan 50 *node* dengan parameter ukur *packet delivery ratio, packet loss, throughput, dan end to end delay* yang ditunjukkan pada tabel 6.4.

Tabel 6.4 Hasil Implementasi 5 *Node* Serangan *Rushing*

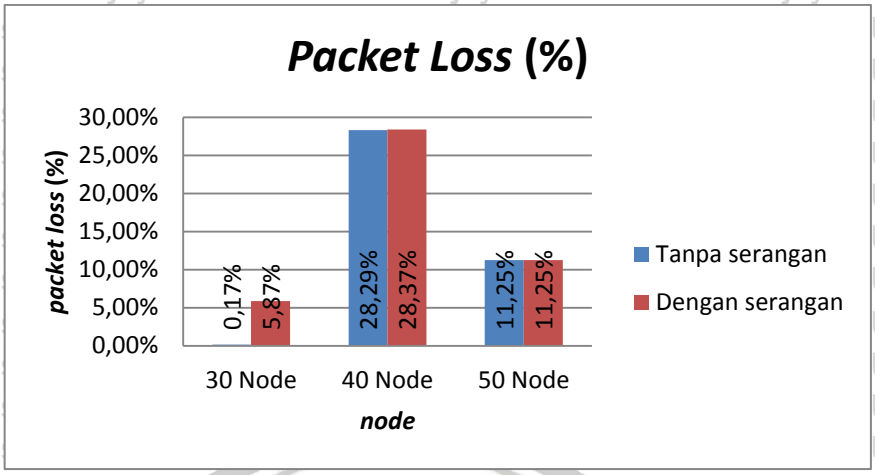
Jumlah *Node*	Parameter Ukur			
	Packet Delivery Ratio (%)	*Packet Loss* (%)	*Throughput* (kbps)	*End to End Delay* (ms)
30 *Node*	94,1274 %	5,87262 %	96,04 kbps	772,44 ms
40 *Node*	71,294 %	28,3706 %	101,44 kbps	18,2695 ms
50 *Node*	88,751 %	11,249 %	90,53 kbps	1337,27 ms

Pada tabel 6.4 menunjukkan hasil implementasi 4 serangan *rushing* dari 30 *node,* 40 *node,* dan 50 *node*.



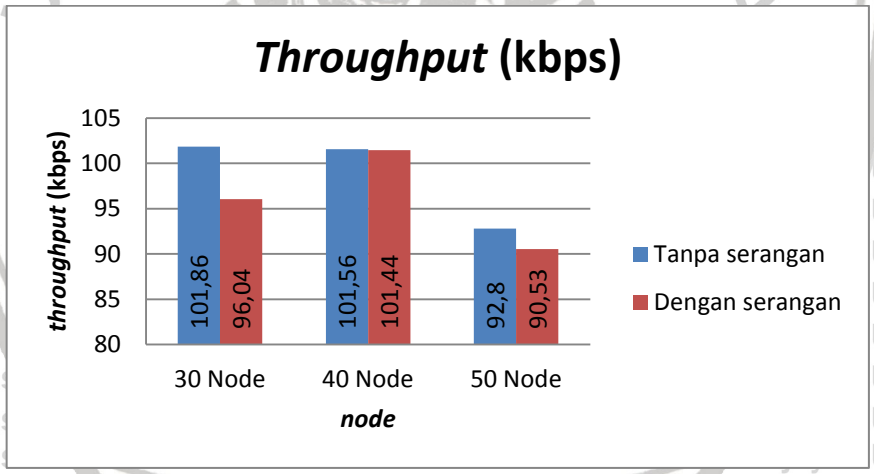
Gambar 6.13 Grafik *Packet Delivery Ratio* dari 5 *Node* Serangan *Rushing*

Pada gambar 6.13 menunjukkan hasil *packet delivery ratio* (PDR) dari hasil implementasi serangan *rushing* dengan jumlah 5 *node*. Dapat dilihat dari grafik terdapat penurunan nilai *packet delivery ratio* (PDR) dibandingkan dengan implementasi tanpa serangan. Pada 30 *node* terjadi penurunan sebanyak 5,71%, 40 *node* terjadi penurunan sebanyak 0,42%, dan 50 *node* tidak terjadi penurunan. Penurunan nilai PDR pada implementasi 5 serangan *rushing* paling besar terdapat pada 30 *node* dengan penurunan sebanyak 5,71%.



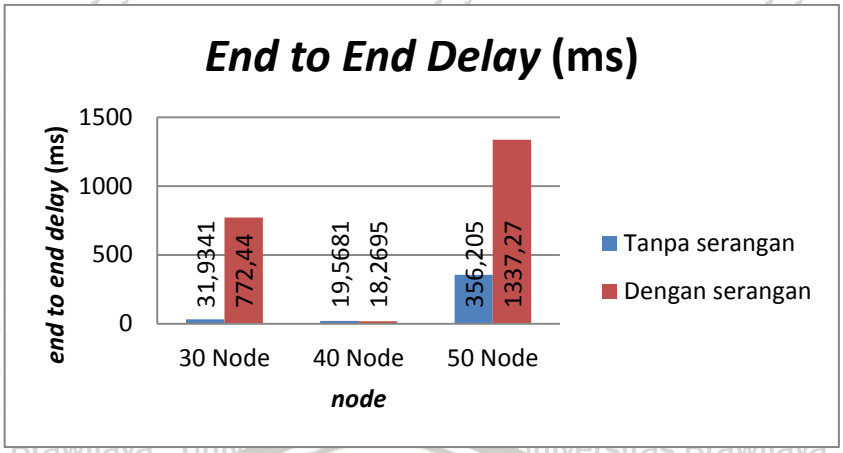
Gambar 6.14 Grafik *Packet Loss* dari 5 *Node* Serangan *Rushing*

Pada gambar 6.14 menunjukkan hasil *packet loss* dari hasil implementasi serangan *rushing* dengan jumlah 5 *node*. Dapat dilihat dari grafik terdapat kenaikan nilai *packet loss* dibandingkan dengan implementasi tanpa serangan. Pada 30 *node* terjadi kenaikan sebanyak 5,71%, 40 *node* terjadi kenaikan sebanyak 0,42%, dan 50 *node* tidak terjadi kenaikan. Kenaikan nilai *packet loss* pada implementasi 5 serangan *rushing* paling besar terdapat pada 30 *node* dengan penaikan sebanyak 5,71%.



Gambar 6.15 Grafik *Throughput* dari 5 *Node* Serangan *Rushing*

Pada gambar 6.15 menunjukkan hasil *throughput* dari hasil implementasi serangan *rushing* dengan jumlah 5 *node*. Dapat dilihat dari grafik terdapat penurunan nilai *throughput* dibandingkan dengan implementasi tanpa serangan. Pada 30 *node* terjadi penurunan sebanyak 5,82 kbps, 40 *node* terjadi penurunan sebanyak 0,12 kbps, dan 50 *node* terjadi penurunan sebanyak 2,27 kbps. penurunan nilai *throughput* pada implementasi 5 serangan *rushing* paling besar terdapat pada 30 *node* dengan penaikan sebanyak 5,82 kbps.



Gambar 6.16 Grafik *End to End Delay* dari 5 *Node* Serangan *Rushing*

Pada gambar 6.16 menunjukkan hasil *end to end delay* dari hasil implementasi serangan *rushing* dengan jumlah 5 *node*. Dapat dilihat dari grafik terdapat kenaikan nilai *end to end delay* dibandingkan dengan implementasi tanpa serangan. Pada 30 *node* terjadi kenaikan sebanyak 740,5059 ms, 40 *node* terjadi penurunan sebanyak 1,2986 ms, dan 50 *node* terjadi kenaikan sebanyak 981,065 ms. Kenaikan nilai *end to end delay* pada implementasi 5 serangan *rushing* paling besar terdapat pada 50 *node* dengan penaikan sebanyak 981,065 ms.

6.2 Analisis

Berdasarkan dari implementasi yang dilakukan didapatkan simpulan bahwa serangan *rushing* memepengaruhi kinerja dari *routing* protokol AODV dengan mengubah jalur yang seharusnya dilalui. Serangan *rushing* berdampak pada ketiga jumlah *node* yang diujikan 30 *node*, 40 *node*, dan 50 *node* dengan adanya penurunan kualitas dengan ditunjukkan nilai pada *packet delivery ratio* yang turun, *packet loss* yang naik, *throughput* yang turun, dan *end to end delay* yang naik dibandingkan dengan implementasi tanpa serangan *rushing*. Penurunan paling besar terjadi pada implementasi 4 serangan *rushing* terjadi penurunan kualitas dimana *packet delivary ratio* pada 40 *node* terjadi penurunan sebesar 12,90% , *packet loss* pada 40 *node* terjadi kenaikan sebesar 12,90%, *throughput* pada 40 *node* terjadi penurunan sebesar 18,27 kbps,dan *end to end delay* pada 30 *node* terjadi kenaikan sebesar 4035,3259 ms. Dibandingkan dengan percobaan pada 30 *node* dan 50 *node* terjadinya penurunan kualitas sangat kecil dan cenderung tidak ada penurunan.

Berdasarkan hal tersebut serangan *rushing* secara keseluruhan memberikan dampak yang lebih pada jumlah node yang sedikit, namun serangan *rushing* sangat dipengaruhi oleh posisi *node* serangan, pergerakan *node* serangan, pergerakan *node sender* dan *receiver*, dan pergerakan *node sekitarnya*.

BAB 7 KESIMPULAN

7.1 Kesimpulan

1. Serangan *rushing* memepengaruhi kinerja dari *routing* protokol AODV dengan mengubah jalur yang seharusnya dilalui.
2. Serangan *rushing* memberikan dampak pada *routing* protokol AODV dengan menurunkan kinerjanya. Dengan dibuktikan dari nilai *packet delivery ratio, packet loss, throughput,* dan *end to end delay* yang turun kualitasnya. Penurunan paling besar terjadi pada implementasi 4 serangan *rushing* terjadi penurunan kualitas dimana *packet delivary ratio* pada 40 *node* terjadi penurunan sebesar 12,90% , *packet loss* pada 40 *node* terjadi kenaikan sebesar 12,90%, *throughput* pada 40 *node* terjadi penurunan sebesar 18,27 kbps,dan *end to end delay* pada 30 *node* terjadi kenaikan sebesar 4035,3259 ms. Dibandingkan dengan percobaan pada 30 *node* dan 50 *node* terjadinya penurunan kualitas sangat kecil dan cenderung tidak ada penurunan. Berdasarkan hal tersebut serangan *rushing* secara keseluruhan memberikan dampak yang lebih pada jumlah node yang sedikit, namun serangan *rushing* sangat dipengaruhi oleh posisi *node* serangan, pergerakan *node* serangan, pergerakan *node sender* dan *receiver*, dan pergerakan *node sekitarnya*.

7.2 Saran

Saran untuk penelitian yang sejenis atau penelitian selanjutanya dapat menggunakan protokol selain AODV ataupun dapat menggunakan model pergerakan selain *Random Way Point* sehingga didapatkan hasil dari serangan *rushing* yang lebih beragam. Pelatakan *node* juga dapat menggunakan SUMO dan juga dapat menggunakan *network simulator* yang lebih baru.

DAFTAR PUSTAKA

Bayu Aji, M. A., S., & Zahra, A. A. (2015). EVALUASI KINERJA PROTOKOL ROUTING DSDV TERHADAP PENGARUH MALICIOUS *NODE* PADA MANET MENGGUNAKAN NETWORK SIMULATOR 2 (NS-2). TRANSIENT.

Bhattacharyya, A., Banerjee, A., & Bose, D. (2011). Different types of attacks in Mobile ADHOC Network: Prevention and mitigation techniques. Rasearch Gate.

Iskandar, I., & Hidayat, A. (2015). *Analisa Quality of Service (QoS) Jaringan Internet Kampus (Studi Kasus: UIN Suska Riau)* (Vol. 1). Jurnal CoreIT.

Jayanti, V. N. (2014). Routing Protocols in MANET: Comparative Study. Routing Protocols in MANET: Comparative Study. IEEE.

Mobile Ad Hoc Networking. (2017, September 26). Retrieved January 10, 2018, from https://www.cisco.com/c/en/us/products/ios-nx-os-software/mobile-ad-hoc-networking/index.html

Prasad, J. R., & Babu, D. R. (2015). Implentation of Rushing Attack in Manets. *International Journal of Computational Mathematical Ideas*. Jain, A., & Sharma, S. D. (2014). Rushing attack prevention algorithm for manet using random route selection to make DSR and AODV more efficient. *International Journal Of Engineering And Computer Scienc, 3*(6), 6520-6524.

Pratomo, I., & Hizburrahman, M. H. (2015). Pendeteksian Dan Pencegahan Serangan Black Hole & Gray Hole Pada Manet. JAVA Journal of Electricl and Electronics Engineering.

Purba, D. U., Primananda, R., & Amron, K. (2018). Analisis Kinerja Protokol Ad Hoc On-Demand Distance Vector (AODV) dan Fisheye State Routing (FSR) pada Mobile Ad Hoc Network. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*

Sari, R. F., Syarif, A., & Budiardjo, B. (2010). Analisis Kinerja Protokol Routing Ad Hoc On-Demand Distance Vector (Aodv) Pada Jaringan Ad Hoc Hybrid: Perbandingan Hasil Simulasi Dengan Ns-2 Dan Implementasi Pada Testbed Dengan Pda. MAKARA of Technology Series, 12(1). doi:10.7454/mst.v12i1.517

Seyyedtaj, M., & Jamali, M. A. (2014). Different Types of Attacks and Detection Techniques in Mobile Ad Hoc Network. International Journal of Computer Applications Technology and Research, 3(9), 541-546. doi:10.7753/ijcatr0309.1002

Seyyedtaj, M., & Jamali, M. A. (2014). Different Types of Attacks and Detection Techniques in Mobile Ad Hoc Network. *International Journal of Computer Applications Technology and Research, 3*(9), 541-546. doi:10.7753/ijcatr0309.1002

Thilagarasi, R., & Geetha, D. (2015). Review On Rushing Attack And Its Prevention Techniques In MANET. Research Gate

LAMPIRAN 1 *SCRIPT* AODV.TCL

```
#===================================
#     Simulation parameters setup
#===================================
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)     30                         ;# number of
mobilenodes
set val(rp)     AODV                      ;# routing protocol
set val(x)      1000                   ;# X dimension of
topography
set val(y)      1000                      ;# Y dimension of
topography
set val(stop)   100.0                        ;# time of
simulation end
set val(t1)     0.0                         ;
set val(t2)     0.0                         ;
set val(sc) "scenario_30"

#===================================
#        Initialization
#===================================
#Create a ns simulator
set ns_ [new Simulator]


#Setup topography object
set topo       [new Topography]
$topo load_flatgrid $val(x) $val(y)
set god_ [create-god $val(nn)]


#Open the NS trace file
set tracefile [open out30.tr w]
$ns_ trace-all $tracefile

#Open the NAM trace file
set namfile [open out30.nam w]
$ns_ namtrace-all $namfile
$ns_ namtrace-all-wireless $namfile $val(x) $val(y)
set chan [new $val(chan)];#Create wireless channel

#===================================
#     Mobile node parameter setup
#===================================
$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) \
                -channel        $chan \
                -topoInstance   $topo \
                -agentTrace     ON \
                -routerTrace    ON \
                -macTrace       ON \
                -movementTrace  ON

#===================================
#        Nodes Definition
#===================================

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

puts "Loading scenario file..."
source $val(sc)

for {set i 0} {$i < $val(nn)} {incr i} {
            $ns_ initial_node_pos $node_($i) 50
}
for {set i 0} {$i < $val(nn)} {incr i} {
            $ns_ at $val(stop).0 "$node_($i) reset";
}

$node_(16) color red
$ns_ at 0.0 "$node_(16) color red"
$ns_ at 0.0 "$node_(16) label attacker"

$node_(25) color red
$ns_ at 0.0 "$node_(25) color red"
$ns_ at 0.0 "$node_(25) label attacker"

$node_(11) color red
$ns_ at 0.0 "$node_(11) color red"
$ns_ at 0.0 "$node_(11) label attacker"

$node_(10) color red
$ns_ at 0.0 "$node_(10) color red"
$ns_ at 0.0 "$node_(10) label attacker"

$node_(3) color red
$ns_ at 0.0 "$node_(3) color red"
$ns_ at 0.0 "$node_(3) label attacker"

$node_(18) color blue
$ns_ at 0.0 "$node_(18) color blue"
$ns_ at 0.0 "$node_(18) label receiver"
```

```
$node_(4) color green
$ns_ at 0.0 "$node_(4) color green"
$ns_ at 0.0 "$node_(4) label sender"
#Rushing attackers
$ns_ at 0.0 "[$node_(16) set ragent_] rushingattack"
$ns_ at 0.0 "[$node_(25) set ragent_] rushingattack"
$ns_ at 0.0 "[$node_(11) set ragent_] rushingattack"
$ns_ at 0.0 "[$node_(3) set ragent_] rushingattack"
$ns_ at 0.0 "[$node_(10) set ragent_] rushingattack"

#=====================================
#        Agents Definition
#=====================================
#Setup a UDP connection
set udp0 [new Agent/UDP]
$ns_ attach-agent $node_(4) $udp0
set null1 [new Agent/Null]
$ns_ attach-agent $node_(18) $null1
$ns_ connect $udp0 $null1
$udp0 set packetSize_ 1024


#Setup a CBR Application over UDP connection
set cbr0 [new Application/Traffic/CBR]
$cbr0 attach-agent $udp0
$cbr0 set packetSize_ 1024
$cbr0 set rate_ 0.1Mb
$cbr0 set random_ null
$ns_ at 1.0 "$cbr0 start"
$ns_ at 100.0 "$cbr0 stop"
#Setup a UDP connection
set udp1 [new Agent/UDP]
$ns_ attach-agent $node_(4) $udp1
set null2 [new Agent/Null]
$ns_ attach-agent $node_(18) $null2
$ns_ connect $udp1 $null1
$udp1 set packetSize_ 1024

#=====================================
#        Termination
#=====================================
#Define a 'finish' procedure
proc finish {} {
    global ns_ tracefile namfile
    $ns_ flush-trace
    close $tracefile
    close $namfile
    exec nam out30.nam &
    exit 0
}
for {set i 0} {$i < $val(nn) } { incr i } {
    $ns_ at $val(stop).0 "$node_($i) reset"
}
$ns_ at $val(stop) "$ns_ nam-end-wireless $val(stop)"
$ns_ at $val(stop) "finish"
$ns_ at $val(stop) "puts \"done\" ; $ns_ halt"
$ns_ run
```

LAMPIRAN 2 SKENARIO PERGERAKAN NODE

```
#
# nodes: 30, pause: 0.00, max speed: 50.00, max x: 1000.00, max y:
1000.00
#
$node_(0) set X_ 215.930176878692
$node_(0) set Y_ 777.425335884204
$node_(0) set Z_ 0.000000000000
$node_(1) set X_ 157.073377322234
$node_(1) set Y_ 36.530929989748
$node_(1) set Z_ 0.000000000000
$node_(2) set X_ 227.148203935201
$node_(2) set Y_ 876.714839217413
$node_(2) set Z_ 0.000000000000
$node_(3) set X_ 274.507894203449
$node_(3) set Y_ 176.327314618063
$node_(3) set Z_ 0.000000000000
$node_(4) set X_ 878.265528399318
$node_(4) set Y_ 780.842990478348
$node_(4) set Z_ 0.000000000000
$node_(5) set X_ 537.042851491113
$node_(5) set Y_ 948.644482139045
$node_(5) set Z_ 0.000000000000
$node_(6) set X_ 958.261347682765
$node_(6) set Y_ 233.831902881394
$node_(6) set Z_ 0.000000000000
$node_(7) set X_ 485.077484719482
$node_(7) set Y_ 655.429984053931
$node_(7) set Z_ 0.000000000000
$node_(8) set X_ 971.358289486385
$node_(8) set Y_ 171.511245815628
$node_(8) set Z_ 0.000000000000
$node_(9) set X_ 93.258623359211
$node_(9) set Y_ 334.528864031193
$node_(9) set Z_ 0.000000000000
$node_(10) set X_ 865.088119390032
$node_(10) set Y_ 443.095910168241
$node_(10) set Z_ 0.000000000000
$node_(11) set X_ 328.072291854545
$node_(11) set Y_ 297.368508263642
$node_(11) set Z_ 0.000000000000
$node_(12) set X_ 522.189392153032
$node_(12) set Y_ 65.612144220463
$node_(12) set Z_ 0.000000000000
$node_(13) set X_ 941.762194243850
$node_(13) set Y_ 336.569603999722
$node_(13) set Z_ 0.000000000000
$node_(14) set X_ 179.253704679378
$node_(14) set Y_ 122.637210997879
$node_(14) set Z_ 0.000000000000
$node_(15) set X_ 89.900911482840
$node_(15) set Y_ 757.952169900306
$node_(15) set Z_ 0.000000000000
$node_(16) set X_ 573.412664297464
$node_(16) set Y_ 380.249182482210
$node_(16) set Z_ 0.000000000000
$node_(17) set X_ 656.005867395834
```

```
$node_(17) set Y_ 33.038145599871
$node_(17) set Z_ 0.000000000000
$node_(18) set X_ 138.141647384843
$node_(18) set Y_ 205.946248452370
$node_(18) set Z_ 0.000000000000
$node_(19) set X_ 155.357172553030
$node_(19) set Y_ 42.944187515506
$node_(19) set Z_ 0.000000000000
$node_(20) set X_ 359.959277061636
$node_(20) set Y_ 337.319123829337
$node_(20) set Z_ 0.000000000000
$node_(21) set X_ 80.676211691613
$node_(21) set Y_ 872.093443618025
$node_(21) set Z_ 0.000000000000
$node_(22) set X_ 518.766162661687
$node_(22) set Y_ 107.730381518584
$node_(22) set Z_ 0.000000000000
$node_(23) set X_ 113.517237736747
$node_(23) set Y_ 130.710049576054
$node_(23) set Z_ 0.000000000000
$node_(24) set X_ 447.655018212331
$node_(24) set Y_ 857.558901741228
$node_(24) set Z_ 0.000000000000
$node_(25) set X_ 428.700273663191
$node_(25) set Y_ 300.302050184185
$node_(25) set Z_ 0.000000000000
$node_(26) set X_ 222.362110682605
$node_(26) set Y_ 718.017466680061
$node_(26) set Z_ 0.000000000000
$node_(27) set X_ 54.480819108898
$node_(27) set Y_ 528.644888186394
$node_(27) set Z_ 0.000000000000
$node_(28) set X_ 461.164620221183
$node_(28) set Y_ 103.634038371006
$node_(28) set Z_ 0.000000000000
$node_(29) set X_ 176.988431907630
$node_(29) set Y_ 997.399812205499
$node_(29) set Z_ 0.000000000000
$ns_ at 0.000000000000 "$node_(0) setdest 605.332521234910
194.173451836239 7.455103099500"
$ns_ at 0.000000000000 "$node_(1) setdest 183.986208697113
316.418711518657 5.330564898615"
$ns_ at 0.000000000000 "$node_(2) setdest 917.976608718544
186.744440776027 21.307598725422"
$ns_ at 0.000000000000 "$node_(3) setdest 532.375088830949
492.409142297949 20.039528461225"
$ns_ at 0.000000000000 "$node_(4) setdest 584.100025820733
760.302327140906 26.471634932351"
$ns_ at 0.000000000000 "$node_(5) setdest 897.245168133405
741.443718136357 44.094121077000"
$ns_ at 0.000000000000 "$node_(6) setdest 947.531848467566
927.519628760424 20.554209250322"
$ns_ at 0.000000000000 "$node_(7) setdest 117.385102067166
483.925705695652 37.054477168091"
$ns_ at 0.000000000000 "$node_(8) setdest 257.762628983387
920.996670044797 1.128340031648"
$ns_ at 0.000000000000 "$node_(9) setdest 740.418782226533
124.862784280316 43.313654758791"
$ns_ at 0.000000000000 "$node_(10) setdest 725.753539697439
```

```
207.440082018156 9.424361460904"
$ns_ at 0.000000000000 "$node_(11) setdest 162.703944799123
978.085661875507 11.181654973371"
$ns_ at 0.000000000000 "$node_(12) setdest 582.806239422340
758.227373638957 36.978417435081"
$ns_ at 0.000000000000 "$node_(13) setdest 535.665269572841
317.913208418979 7.637146100990"
$ns_ at 0.000000000000 "$node_(14) setdest 196.304950823875
261.218718330724 19.692372180525"
$ns_ at 0.000000000000 "$node_(15) setdest 320.966608999543
538.563173502018 31.831678753018"
$ns_ at 0.000000000000 "$node_(16) setdest 110.438384108987
243.643369916319 7.990126500825"
$ns_ at 0.000000000000 "$node_(17) setdest 509.016850235757
25.577730108092 42.997542732276"
$ns_ at 0.000000000000 "$node_(18) setdest 893.674556138997
114.496468989008 12.867396726836"
$ns_ at 0.000000000000 "$node_(19) setdest 770.031640112070
691.631375546410 29.943682702325"
$ns_ at 0.000000000000 "$node_(20) setdest 532.114519430282
424.819457661930 19.521098062486"
$ns_ at 0.000000000000 "$node_(21) setdest 204.922161675945
486.953526801973 37.270897627423"
$ns_ at 0.000000000000 "$node_(22) setdest 681.018513080629
972.469387639694 13.721099892629"
$ns_ at 0.000000000000 "$node_(23) setdest 522.800642238588
447.571717690424 45.590690775510"
$ns_ at 0.000000000000 "$node_(24) setdest 982.488938224898
198.649751329596 23.129084231530"
$ns_ at 0.000000000000 "$node_(25) setdest 358.990835181925
140.803758354667 16.947896376988"
$ns_ at 0.000000000000 "$node_(26) setdest 557.634906142033
328.905653537339 49.132014361550"
$ns_ at 0.000000000000 "$node_(27) setdest 229.317156294819
763.763119900303 5.225565362935"
$ns_ at 0.000000000000 "$node_(28) setdest 194.954518356952
315.674264603352 23.146808930798"
$ns_ at 0.000000000000 "$node_(29) setdest 935.226672684576
142.811308313336 41.259142961796"
$god_ set-dist 0 1 5
$god_ set-dist 0 2 1
$god_ set-dist 0 3 4
$god_ set-dist 0 4 16777215
$god_ set-dist 0 5 2
$god_ set-dist 0 6 16777215
$god_ set-dist 0 7 2
$god_ set-dist 0 8 16777215
$god_ set-dist 0 9 3
$god_ set-dist 0 10 16777215
$god_ set-dist 0 11 4
$god_ set-dist 0 12 6
$god_ set-dist 0 13 16777215
$god_ set-dist 0 14 4
$god_ set-dist 0 15 1
$god_ set-dist 0 16 6
$god_ set-dist 0 17 6
$god_ set-dist 0 18 4
$god_ set-dist 0 19 5
$god_ set-dist 0 20 5
```

```
$god_ set-dist 0 21 1
$god_ set-dist 0 22 6
$god_ set-dist 0 23 4
$god_ set-dist 0 24 1
$god_ set-dist 0 25 5
$god_ set-dist 0 26 1
$god_ set-dist 0 27 2
$god_ set-dist 0 28 5
$god_ set-dist 0 29 1
$god_ set-dist 1 2 5
$god_ set-dist 1 3 1
$god_ set-dist 1 4 16777215
$god_ set-dist 1 5 7
$god_ set-dist 1 6 16777215
$god_ set-dist 1 7 7
$god_ set-dist 1 8 16777215
$god_ set-dist 1 9 2
$god_ set-dist 1 10 16777215
$god_ set-dist 1 11 2
$god_ set-dist 1 12 3
$god_ set-dist 1 13 16777215
$god_ set-dist 1 14 1
$god_ set-dist 1 15 4
$god_ set-dist 1 16 3
$god_ set-dist 1 17 3
$god_ set-dist 1 18 1
$god_ set-dist 1 19 1
$god_ set-dist 1 20 2
$god_ set-dist 1 21 5
$god_ set-dist 1 22 3
$god_ set-dist 1 23 1
$god_ set-dist 1 24 6
$god_ set-dist 1 25 2
$god_ set-dist 1 26 5
$god_ set-dist 1 27 3
$god_ set-dist 1 28 2
$god_ set-dist 1 29 6
$god_ set-dist 2 3 4
$god_ set-dist 2 4 16777215
$god_ set-dist 2 5 2
$god_ set-dist 2 6 16777215
$god_ set-dist 2 7 2
$god_ set-dist 2 8 16777215
$god_ set-dist 2 9 3
$god_ set-dist 2 10 16777215
$god_ set-dist 2 11 4
$god_ set-dist 2 12 6
$god_ set-dist 2 13 16777215
$god_ set-dist 2 14 4
...
$ns_ at 0.058538743041 "$god_ set-dist 12 16 2"
$ns_ at 0.058538743041 "$god_ set-dist 12 20 2"
$ns_ at 0.058538743041 "$god_ set-dist 12 25 1"
$ns_ at 0.248214270787 "$god_ set-dist 17 20 2"
$ns_ at 0.248214270787 "$god_ set-dist 20 28 1"
$ns_ at 0.291506096509 "$god_ set-dist 1 7 6"
$ns_ at 0.291506096509 "$god_ set-dist 3 7 5"
$ns_ at 0.291506096509 "$god_ set-dist 7 9 4"
$ns_ at 0.291506096509 "$god_ set-dist 7 11 5"
```

```
$ns_ at 0.291506096509 "$god_ set-dist 7 12 7"
$ns_ at 0.291506096509 "$god_ set-dist 7 14 5"
$ns_ at 0.291506096509 "$god_ set-dist 7 15 2"
$ns_ at 0.291506096509 "$god_ set-dist 7 16 7"
$ns_ at 0.291506096509 "$god_ set-dist 7 17 7"
$ns_ at 0.291506096509 "$god_ set-dist 7 18 5"
$ns_ at 0.291506096509 "$god_ set-dist 7 19 6"
$ns_ at 0.291506096509 "$god_ set-dist 7 20 6"
$ns_ at 0.291506096509 "$god_ set-dist 7 21 2"
$ns_ at 0.291506096509 "$god_ set-dist 7 22 7"
$ns_ at 0.291506096509 "$god_ set-dist 7 23 5"
$ns_ at 0.291506096509 "$god_ set-dist 7 25 6"
$ns_ at 0.291506096509 "$god_ set-dist 7 26 1"
$ns_ at 0.291506096509 "$god_ set-dist 7 27 3"
$ns_ at 0.291506096509 "$god_ set-dist 7 28 6"
$ns_ at 0.296993066209 "$god_ set-dist 1 7 5"
$ns_ at 0.296993066209 "$god_ set-dist 1 26 4"
$ns_ at 0.296993066209 "$god_ set-dist 3 7 4"
$ns_ at 0.296993066209 "$god_ set-dist 3 26 3"
$ns_ at 0.296993066209 "$god_ set-dist 7 9 3"
$ns_ at 0.296993066209 "$god_ set-dist 7 11 4"
$ns_ at 0.296993066209 "$god_ set-dist 7 12 6"
$ns_ at 0.296993066209 "$god_ set-dist 7 14 4"
$ns_ at 0.296993066209 "$god_ set-dist 7 16 6"
$ns_ at 0.296993066209 "$god_ set-dist 7 17 6"
$ns_ at 0.296993066209 "$god_ set-dist 7 18 4"
$ns_ at 0.296993066209 "$god_ set-dist 7 19 5"
$ns_ at 0.296993066209 "$god_ set-dist 7 20 5"
$ns_ at 0.296993066209 "$god_ set-dist 7 22 6"
$ns_ at 0.296993066209 "$god_ set-dist 7 23 4"
$ns_ at 0.296993066209 "$god_ set-dist 7 25 5"
$ns_ at 0.296993066209 "$god_ set-dist 7 27 2"
$ns_ at 0.296993066209 "$god_ set-dist 7 28 5"
$ns_ at 0.296993066209 "$god_ set-dist 9 26 2"
$ns_ at 0.296993066209 "$god_ set-dist 11 26 3"
$ns_ at 0.296993066209 "$god_ set-dist 12 26 5"
$ns_ at 0.296993066209 "$god_ set-dist 14 26 3"
$ns_ at 0.296993066209 "$god_ set-dist 16 26 5"
$ns_ at 0.296993066209 "$god_ set-dist 17 26 5"
$ns_ at 0.296993066209 "$god_ set-dist 18 26 3"
$ns_ at 0.296993066209 "$god_ set-dist 19 26 4"
$ns_ at 0.296993066209 "$god_ set-dist 20 26 4"
$ns_ at 0.296993066209 "$god_ set-dist 22 26 5"
$ns_ at 0.296993066209 "$god_ set-dist 23 26 3"
$ns_ at 0.296993066209 "$god_ set-dist 25 26 4"
$ns_ at 0.296993066209 "$god_ set-dist 26 27 1"
$ns_ at 0.296993066209 "$god_ set-dist 26 28 4"
$ns_ at 0.403460978758 "$god_ set-dist 0 22 5"
$ns_ at 0.403460978758 "$god_ set-dist 1 22 2"
$ns_ at 0.403460978758 "$god_ set-dist 2 22 5"
$ns_ at 0.403460978758 "$god_ set-dist 3 22 1"
$ns_ at 0.403460978758 "$god_ set-dist 5 22 7"
$ns_ at 0.403460978758 "$god_ set-dist 7 22 5"
$ns_ at 0.403460978758 "$god_ set-dist 9 22 2"
$ns_ at 0.403460978758 "$god_ set-dist 14 22 2"
...
#
# Destination Unreachables: 973
#
```

```
# Route Changes: 10196
#
# Link Changes: 1251
#
# Node | Route Changes | Link Changes
#    0 |           431 |           78
#    1 |           803 |           36
#    2 |           720 |           80
#    3 |           679 |           75
#    4 |           699 |           51
#    5 |           491 |           65
#    6 |           917 |           49
#    7 |           634 |          105
#    8 |           678 |           26
#    9 |           751 |          111
#   10 |           730 |           87
#   11 |          1003 |           51
#   12 |           680 |          149
#   13 |           513 |           79
#   14 |           833 |           84
#   15 |           516 |          100
#   16 |           684 |           81
#   17 |           671 |          114
#   18 |           687 |           55
#   19 |           611 |           94
#   20 |           622 |          115
#   21 |           909 |           59
#   22 |           653 |           73
#   23 |           737 |          125
#   24 |           544 |           99
#   25 |           702 |           86
```

LAMPIRAN 3 *TRACE FILE*

```
M 0.00000 0 (127.29, 852.62, 0.00), (744.99, 691.07), 4.32
M 0.00000 1 (129.80, 399.13, 0.00), (554.82, 430.05), 9.84
M 0.00000 2 (15.96, 673.03, 0.00), (952.46, 420.58), 3.75
M 0.00000 3 (495.61, 118.28, 0.00), (87.38, 757.35), 1.81
M 0.00000 4 (784.04, 658.12, 0.00), (102.69, 671.66), 4.11
M 0.00000 5 (675.18, 259.49, 0.00), (10.76, 782.81), 0.21
M 0.00000 6 (992.92, 733.00, 0.00), (35.16, 293.73), 6.53
M 0.00000 7 (728.73, 57.52, 0.00), (642.46, 167.70), 0.11
M 0.00000 8 (229.14, 639.76, 0.00), (473.33, 173.62), 9.05
M 0.00000 9 (879.59, 481.75, 0.00), (468.57, 136.70), 1.92
M 0.00000 10 (429.41, 55.39, 0.00), (894.58, 870.14), 5.82
M 0.00000 11 (320.03, 989.08, 0.00), (420.27, 919.46), 7.21
M 0.00000 12 (389.88, 641.91, 0.00), (698.35, 970.59), 3.44
M 0.00000 13 (306.99, 543.80, 0.00), (139.58, 290.74), 4.54
M 0.00000 14 (482.87, 255.02, 0.00), (411.22, 657.46), 8.37
M 0.00000 15 (744.95, 985.81, 0.00), (329.62, 551.10), 3.45
M 0.00000 16 (799.02, 328.70, 0.00), (15.93, 380.53), 3.25
M 0.00000 17 (991.95, 165.10, 0.00), (941.35, 711.90), 9.67
M 0.00000 18 (51.13, 651.65, 0.00), (489.08, 854.26), 4.60
M 0.00000 19 (240.54, 375.20, 0.00), (487.75, 861.03), 5.34
M 0.00000 20 (409.82, 963.82, 0.00), (811.23, 107.13), 8.06
M 0.00000 21 (398.48, 57.04, 0.00), (989.27, 596.63), 5.50
M 0.00000 22 (245.77, 408.49, 0.00), (870.57, 6.25), 2.23
M 0.00000 23 (469.52, 394.19, 0.00), (303.37, 77.22), 3.60
M 0.00000 24 (431.43, 911.28, 0.00), (629.02, 711.13), 3.94
M 0.00000 25 (635.02, 642.83, 0.00), (137.16, 815.29), 0.25
M 0.00000 26 (237.68, 830.16, 0.00), (841.79, 265.85), 2.95
M 0.00000 27 (704.60, 720.74, 0.00), (343.33, 781.64), 5.09
M 0.00000 28 (797.30, 593.12, 0.00), (789.48, 845.37), 2.99
M 0.00000 29 (190.49, 589.41, 0.00), (33.16, 642.55), 1.91
s 1.000000000 _4_ AGT  --- 0 cbr 1024 [0 0 0 0] ------- [4:0 18:0
32 0] [0] 0 4
r 1.000000000 _4_ RTR  --- 0 cbr 1024 [0 0 0 0] ------- [4:0 18:0
32 0] [0] 0 4
s 1.000000000 _4_ RTR  --- 0 AODV 48 [0 0 0 0] ------- [4:255 -
1:255 30 0] [0x2 1 1 [18 0] [4 4]] (REQUEST)
s 1.000535000 _4_ MAC  --- 0 AODV 106 [0 ffffffff 4 800] -------
[4:255 -1:255 30 0] [0x2 1 1 [18 0] [4 4]] (REQUEST)
r 1.001383215 _28_ MAC  --- 0 AODV 48 [0 ffffffff 4 800] -------
[4:255 -1:255 30 0] [0x2 1 1 [18 0] [4 4]] (REQUEST)
r 1.001383341 _27_ MAC  --- 0 AODV 48 [0 ffffffff 4 800] -------
[4:255 -1:255 30 0] [0x2 1 1 [18 0] [4 4]] (REQUEST)
r 1.001383487 _25_ MAC  --- 0 AODV 48 [0 ffffffff 4 800] -------
[4:255 -1:255 30 0] [0x2 1 1 [18 0] [4 4]] (REQUEST)
r 1.001383677 _9_ MAC  --- 0 AODV 48 [0 ffffffff 4 800] -------
[4:255 -1:255 30 0] [0x2 1 1 [18 0] [4 4]] (REQUEST)
r 1.001383731 _6_ MAC  --- 0 AODV 48 [0 ffffffff 4 800] -------
[4:255 -1:255 30 0] [0x2 1 1 [18 0] [4 4]] (REQUEST)
r 1.001408215 _28_ RTR  --- 0 AODV 48 [0 ffffffff 4 800] -------
[4:255 -1:255 30 0] [0x2 1 1 [18 0] [4 4]] (REQUEST)
s 1.001408215 _28_ RTR  --- 0 AODV 48 [0 ffffffff 4 800] -------
[28:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.001408341 _27_ RTR  --- 0 AODV 48 [0 ffffffff 4 800] -------
[4:255 -1:255 30 0] [0x2 1 1 [18 0] [4 4]] (REQUEST)
s 1.001408341 _27_ RTR  --- 0 AODV 48 [0 ffffffff 4 800] -------
[27:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
```

```
r 1.001408487 _25_ RTR  --- 0 AODV 48 [0 ffffffff 4 800] -------
[4:255 -1:255 30 0] [0x2 1 1 [18 0] [4 4]] (REQUEST)
r 1.001408677 _9_ RTR  --- 0 AODV 48 [0 ffffffff 4 800] -------
[4:255 -1:255 30 0] [0x2 1 1 [18 0] [4 4]] (REQUEST)
s 1.001408677 _9_ RTR  --- 0 AODV 48 [0 ffffffff 4 800] -------
[9:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.001408731 _6_ RTR  --- 0 AODV 48 [0 ffffffff 4 800] -------
[4:255 -1:255 30 0] [0x2 1 1 [18 0] [4 4]] (REQUEST)
s 1.001408731 _6_ RTR  --- 0 AODV 48 [0 ffffffff 4 800] -------
[6:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
s 1.001483677 _9_ MAC  --- 0 AODV 106 [0 ffffffff 9 800] -------
[9:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
s 1.001550533 _25_ RTR  --- 0 AODV 48 [0 ffffffff 4 800] -------
[25:255 -1:255 29 0] [0x2 1 1 [18 0] [4 4]] (REQUEST)
r 1.002332147 _28_ MAC  --- 0 AODV 48 [0 ffffffff 9 800] -------
[9:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.002332252 _16_ MAC  --- 0 AODV 48 [0 ffffffff 9 800] -------
[9:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.002332353 _4_ MAC  --- 0 AODV 48 [0 ffffffff 9 800] -------
[9:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.002357147 _28_ RTR  --- 0 AODV 48 [0 ffffffff 9 800] -------
[9:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.002357252 _16_ RTR  --- 0 AODV 48 [0 ffffffff 9 800] -------
[9:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.002357353 _4_ RTR  --- 0 AODV 48 [0 ffffffff 9 800] -------
[9:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
s 1.002682147 _28_ MAC  --- 0 AODV 106 [0 ffffffff 1c 800] -------
[28:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.003530362 _4_ MAC  --- 0 AODV 48 [0 ffffffff 1c 800] -------
[28:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.003530617 _9_ MAC  --- 0 AODV 48 [0 ffffffff 1c 800] -------
[28:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.003530677 _27_ MAC  --- 0 AODV 48 [0 ffffffff 1c 800] -------
[28:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.003530711 _25_ MAC  --- 0 AODV 48 [0 ffffffff 1c 800] -------
[28:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.003530922 _6_ MAC  --- 0 AODV 48 [0 ffffffff 1c 800] -------
[28:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.003555362 _4_ RTR  --- 0 AODV 48 [0 ffffffff 1c 800] -------
[28:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.003555617 _9_ RTR  --- 0 AODV 48 [0 ffffffff 1c 800] -------
[28:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.003555677 _27_ RTR  --- 0 AODV 48 [0 ffffffff 1c 800] -------
[28:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.003555711 _25_ RTR  --- 0 AODV 48 [0 ffffffff 1c 800] -------
[28:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.003555922 _6_ RTR  --- 0 AODV 48 [0 ffffffff 1c 800] -------
[28:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
s 1.003740922 _6_ MAC  --- 0 AODV 106 [0 ffffffff 6 800] -------
[6:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.004589652 _4_ MAC  --- 0 AODV 48 [0 ffffffff 6 800] -------
[6:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.004589696 _28_ MAC  --- 0 AODV 48 [0 ffffffff 6 800] -------
[6:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.004614652 _4_ RTR  --- 0 AODV 48 [0 ffffffff 6 800] -------
[6:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.004614696 _28_ RTR  --- 0 AODV 48 [0 ffffffff 6 800] -------
[6:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
s 1.005044131 _25_ MAC  --- 0 AODV 106 [0 ffffffff 19 800] -------
```

```
[25:255 -1:255 29 0] [0x2 1 1 [18 0] [4 4]] (REQUEST)
r 1.005892471 _27_ MAC  --- 0 AODV 48 [0 ffffffff 19 800] -------
[25:255 -1:255 29 0] [0x2 1 1 [18 0] [4 4]] (REQUEST)
r 1.005892618 _4_ MAC  --- 0 AODV 48 [0 ffffffff 19 800] -------
[25:255 -1:255 29 0] [0x2 1 1 [18 0] [4 4]] (REQUEST)
r 1.005892695 _28_ MAC  --- 0 AODV 48 [0 ffffffff 19 800] -------
[25:255 -1:255 29 0] [0x2 1 1 [18 0] [4 4]] (REQUEST)
r 1.005892940 _12_ MAC  --- 0 AODV 48 [0 ffffffff 19 800] -------
[25:255 -1:255 29 0] [0x2 1 1 [18 0] [4 4]] (REQUEST)
r 1.005917471 _27_ RTR  --- 0 AODV 48 [0 ffffffff 19 800] -------
[25:255 -1:255 29 0] [0x2 1 1 [18 0] [4 4]] (REQUEST)
r 1.005917618 _4_ RTR  --- 0 AODV 48 [0 ffffffff 19 800] -------
[25:255 -1:255 29 0] [0x2 1 1 [18 0] [4 4]] (REQUEST)
r 1.005917695 _28_ RTR  --- 0 AODV 48 [0 ffffffff 19 800] -------
[25:255 -1:255 29 0] [0x2 1 1 [18 0] [4 4]] (REQUEST)
r 1.005917940 _12_ RTR  --- 0 AODV 48 [0 ffffffff 19 800] -------
[25:255 -1:255 29 0] [0x2 1 1 [18 0] [4 4]] (REQUEST)
s 1.005917940 _12_ RTR  --- 0 AODV 48 [0 ffffffff 19 800] -------
[12:255 -1:255 28 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
s 1.006062471 _27_ MAC  --- 0 AODV 106 [0 ffffffff 1b 800] -------
[27:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.006910811 _25_ MAC  --- 0 AODV 48 [0 ffffffff 1b 800] -------
[27:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.006910812 _4_ MAC  --- 0 AODV 48 [0 ffffffff 1b 800] -------
[27:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.006911001 _28_ MAC  --- 0 AODV 48 [0 ffffffff 1b 800] -------
[27:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.006935811 _25_ RTR  --- 0 AODV 48 [0 ffffffff 1b 800] -------
[27:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.006935812 _4_ RTR  --- 0 AODV 48 [0 ffffffff 1b 800] -------
[27:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.006936001 _28_ RTR  --- 0 AODV 48 [0 ffffffff 1b 800] -------
[27:255 -1:255 29 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
s 1.007805527 _12_ MAC  --- 0 AODV 106 [0 ffffffff c 800] -------
[12:255 -1:255 28 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.008653982 _13_ MAC  --- 0 AODV 48 [0 ffffffff c 800] -------
[12:255 -1:255 28 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.008654058 _8_ MAC  --- 0 AODV 48 [0 ffffffff c 800] -------
[12:255 -1:255 28 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.008654229 _29_ MAC  --- 0 AODV 48 [0 ffffffff c 800] -------
[12:255 -1:255 28 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.008654323 _26_ MAC  --- 0 AODV 48 [0 ffffffff c 800] -------
[12:255 -1:255 28 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.008654335 _25_ MAC  --- 0 AODV 48 [0 ffffffff c 800] -------
[12:255 -1:255 28 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
r 1.008678982 _13_ RTR  --- 0 AODV 48 [0 ffffffff c 800] -------
[12:255 -1:255 28 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
s 1.008678982 _13_ RTR  --- 0 AODV 48 [0 ffffffff c 800] -------
[13:255 -1:255 27 0] [0x2 3 1 [18 0] [4 4]] (REQUEST)
r 1.008679058 _8_ RTR  --- 0 AODV 48 [0 ffffffff c 800] -------
[12:255 -1:255 28 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
s 1.008679058 _8_ RTR  --- 0 AODV 48 [0 ffffffff c 800] -------
[8:255 -1:255 27 0] [0x2 3 1 [18 0] [4 4]] (REQUEST)
r 1.008679229 _29_ RTR  --- 0 AODV 48 [0 ffffffff c 800] -------
[12:255 -1:255 28 0] [0x2 2 1 [18 0] [4 4]] (REQUEST)
.
.
.
```

```
r 20.611104577 _24_ MAC  --- 0 CTS 38 [474 18 0 0]
s 20.611114577 _24_ MAC  --- 0 AODV 102 [13a 1b 18 800] -------
[18:255 4:255 28 27] [0x4 3 [18 8] 10.000000] (REPLY)
r 20.611931114 _27_ MAC  --- 0 AODV 44 [13a 1b 18 800] -------
[18:255 4:255 28 27] [0x4 3 [18 8] 10.000000] (REPLY)
s 20.611941114 _27_ MAC  --- 0 ACK 38 [0 18 0 0]
r 20.611956114 _27_ RTR  --- 0 AODV 44 [13a 1b 18 800] -------
[18:255 4:255 28 27] [0x4 3 [18 8] 10.000000] (REPLY)
f 20.611956114 _27_ RTR  --- 0 AODV 44 [13a 1b 18 800] -------
[18:255 4:255 27 4] [0x4 4 [18 8] 10.000000] (REPLY)
r 20.612245651 _24_ MAC  --- 0 ACK 38 [0 18 0 0]
s 20.612405418 _25_ MAC  --- 0 AODV 106 [0 ffffffff 19 800] ------
- [25:255 -1:255 5 0] [0x2 1 2 [18 5] [4 6]] (REQUEST)
r 20.613253655 _4_ MAC  --- 0 AODV 48 [0 ffffffff 19 800] -------
[25:255 -1:255 5 0] [0x2 1 2 [18 5] [4 6]] (REQUEST)
r 20.613253745 _27_ MAC  --- 0 AODV 48 [0 ffffffff 19 800] -------
[25:255 -1:255 5 0] [0x2 1 2 [18 5] [4 6]] (REQUEST)
r 20.613253971 _28_ MAC  --- 0 AODV 48 [0 ffffffff 19 800] -------
[25:255 -1:255 5 0] [0x2 1 2 [18 5] [4 6]] (REQUEST)
r 20.613254078 _12_ MAC  --- 0 AODV 48 [0 ffffffff 19 800] -------
[25:255 -1:255 5 0] [0x2 1 2 [18 5] [4 6]] (REQUEST)
r 20.613254171 _20_ MAC  --- 0 AODV 48 [0 ffffffff 19 800] -------
[25:255 -1:255 5 0] [0x2 1 2 [18 5] [4 6]] (REQUEST)
r 20.613254227 _6_ MAC  --- 0 AODV 48 [0 ffffffff 19 800] -------
[25:255 -1:255 5 0] [0x2 1 2 [18 5] [4 6]] (REQUEST)
r 20.613278655 _4_ RTR  --- 0 AODV 48 [0 ffffffff 19 800] -------
[25:255 -1:255 5 0] [0x2 1 2 [18 5] [4 6]] (REQUEST)
r 20.613278745 _27_ RTR  --- 0 AODV 48 [0 ffffffff 19 800] -------
[25:255 -1:255 5 0] [0x2 1 2 [18 5] [4 6]] (REQUEST)
r 20.613278971 _28_ RTR  --- 0 AODV 48 [0 ffffffff 19 800] -------
[25:255 -1:255 5 0] [0x2 1 2 [18 5] [4 6]] (REQUEST)
r 20.613279078 _12_ RTR  --- 0 AODV 48 [0 ffffffff 19 800] -------
[25:255 -1:255 5 0] [0x2 1 2 [18 5] [4 6]] (REQUEST)
r 20.613279171 _20_ RTR  --- 0 AODV 48 [0 ffffffff 19 800] -------
[25:255 -1:255 5 0] [0x2 1 2 [18 5] [4 6]] (REQUEST)
r 20.613279227 _6_ RTR  --- 0 AODV 48 [0 ffffffff 19 800] -------
[25:255 -1:255 5 0] [0x2 1 2 [18 5] [4 6]] (REQUEST)
s 20.613683745 _27_ MAC  --- 0 RTS 44 [5ae 4 1b 0]
r 20.614036164 _4_ MAC  --- 0 RTS 44 [5ae 4 1b 0]
s 20.614046164 _4_ MAC  --- 0 CTS 38 [474 1b 0 0]
r 20.614350583 _27_ MAC  --- 0 CTS 38 [474 1b 0 0]
s 20.614360583 _27_ MAC  --- 0 AODV 102 [13a 4 1b 800] -------
[18:255 4:255 27 4] [0x4 4 [18 8] 10.000000] (REPLY)
r 20.615177002 _4_ MAC  --- 0 AODV 44 [13a 4 1b 800] -------
[18:255 4:255 27 4] [0x4 4 [18 8] 10.000000] (REPLY)
s 20.615187002 _4_ MAC  --- 0 ACK 38 [0 1b 0 0]
r 20.615202002 _4_ RTR  --- 0 AODV 44 [13a 4 1b 800] -------
[18:255 4:255 27 4] [0x4 4 [18 8] 10.000000] (REPLY)
s 20.615202002 _4_ RTR  --- 239 cbr 1044 [0 0 0 0] ------- [4:0
18:0 30 27] [239] 0 4
r 20.615491422 _27_ MAC  --- 0 ACK 38 [0 1b 0 0]
s 20.615641239 _25_ MAC  --- 0 RTS 44 [24ee c 19 0]
r 20.615993898 _12_ MAC  --- 0 RTS 44 [24ee c 19 0]
s 20.616003898 _12_ MAC  --- 0 CTS 38 [23b4 19 0 0]
r 20.616308558 _25_ MAC  --- 0 CTS 38 [23b4 19 0 0]
s 20.616318558 _25_ MAC  --- 178 cbr 1102 [13a c 19 800] -------
[4:0 18:0 29 12] [178] 1 4
r 20.625135217 _12_ MAC  --- 178 cbr 1044 [13a c 19 800] -------
[4:0 18:0 29 12] [178] 2 4
```

```
s 20.625145217 _12_ MAC  --- 0 ACK 38 [0 19 0 0]
r 20.625160217 _12_ RTR  --- 178 cbr 1044 [13a c 19 800] -------
[4:0 18:0 29 12] [178] 2 4
D 20.625160217 _12_ RTR NRTE 178 cbr 1044 [13a c 19 800] -------
[4:0 18:0 28 12] [178] 2 4
s 20.625160217 _12_ RTR  --- 0 AODV 32 [0 0 0 0] ------- [12:255 -
1:255 1 0] [0x8 1 [18 0] 0.000000] (ERROR)
r 20.625449877 _25_ MAC  --- 0 ACK 38 [0 19 0 0]
s 20.625639217 _12_ MAC  --- 0 AODV 90 [0 ffffffff c 800] -------
[12:255 -1:255 1 0] [0x8 1 [18 0] 0.000000] (ERROR)
r 20.626359640 _20_ MAC  --- 0 AODV 32 [0 ffffffff c 800] -------
[12:255 -1:255 1 0] [0x8 1 [18 0] 0.000000] (ERROR)
r 20.626359775 _24_ MAC  --- 0 AODV 32 [0 ffffffff c 800] -------
[12:255 -1:255 1 0] [0x8 1 [18 0] 0.000000] (ERROR)
r 20.626359779 _27_ MAC  --- 0 AODV 32 [0 ffffffff c 800] -------
[12:255 -1:255 1 0] [0x8 1 [18 0] 0.000000] (ERROR)
r 20.626359827 _26_ MAC  --- 0 AODV 32 [0 ffffffff c 800] -------
[12:255 -1:255 1 0] [0x8 1 [18 0] 0.000000] (ERROR)
r 20.626359870 _11_ MAC  --- 0 AODV 32 [0 ffffffff c 800] -------
[12:255 -1:255 1 0] [0x8 1 [18 0] 0.000000] (ERROR)
r 20.626359877 _25_ MAC  --- 0 AODV 32 [0 ffffffff c 800] -------
[12:255 -1:255 1 0] [0x8 1 [18 0] 0.000000] (ERROR)
r 20.626384640 _20_ RTR  --- 0 AODV 32 [0 ffffffff c 800] -------
[12:255 -1:255 1 0] [0x8 1 [18 0] 0.000000] (ERROR)
r 20.626384775 _24_ RTR  --- 0 AODV 32 [0 ffffffff c 800] -------
[12:255 -1:255 1 0] [0x8 1 [18 0] 0.000000] (ERROR)
r 20.626384779 _27_ RTR  --- 0 AODV 32 [0 ffffffff c 800] -------
[12:255 -1:255 1 0] [0x8 1 [18 0] 0.000000] (ERROR)
r 20.626384827 _26_ RTR  --- 0 AODV 32 [0 ffffffff c 800] -------
[12:255 -1:255 1 0] [0x8 1 [18 0] 0.000000] (ERROR)
r 20.626384870 _11_ RTR  --- 0 AODV 32 [0 ffffffff c 800] -------
[12:255 -1:255 1 0] [0x8 1 [18 0] 0.000000] (ERROR)
r 20.626384877 _25_ RTR  --- 0 AODV 32 [0 ffffffff c 800] -------
[12:255 -1:255 1 0] [0x8 1 [18 0] 0.000000] (ERROR)
s 20.627034055 _8_ MAC  --- 0 ARP 86 [0 ffffffff 8 806] -------
[REQUEST 8/8 0/13]
r 20.627722139 _19_ MAC  --- 0 ARP 28 [0 ffffffff 8 806] -------
[REQUEST 8/8 0/13]
r 20.627722258 _13_ MAC  --- 0 ARP 28 [0 ffffffff 8 806] -------
[REQUEST 8/8 0/13]
r 20.627722264 _1_ MAC  --- 0 ARP 28 [0 ffffffff 8 806] -------
[REQUEST 8/8 0/13]
r 20.627722375 _22_ MAC  --- 0 ARP 28 [0 ffffffff 8 806] -------
[REQUEST 8/8 0/13]
r 20.627722540 _14_ MAC  --- 0 ARP 28 [0 ffffffff 8 806] -------
[REQUEST 8/8 0/13]
r 20.627722683 _23_ MAC  --- 0 ARP 28 [0 ffffffff 8 806] -------
[REQUEST 8/8 0/13]
r 20.627722744 _29_ MAC  --- 0 ARP 28 [0 ffffffff 8 806] -------
[REQUEST 8/8 0/13]
s 20.627792258 _13_ MAC  --- 0 RTS 44 [52e 8 d 0]
r 20.628144462 _8_ MAC  --- 0 RTS 44 [52e 8 d 0]
s 20.628154462 _8_ MAC  --- 0 CTS 38 [3f4 d 0 0]
r 20.628458666 _13_ MAC  --- 0 CTS 38 [3f4 d 0 0]
s 20.628468666 _13_ MAC  --- 0 ARP 86 [13a 8 d 806] -------- [REPLY
13/13 8/8]
r 20.629156869 _8_ MAC  --- 0 ARP 28 [13a 8 d 806] ------- [REPLY
13/13 8/8]
M 192.82593 14 (437.01, 225.24, 0.00), (477.24, 62.63), 5.39
```

```
M 195.78071 18 (833.13, 921.36, 0.00), (967.86, 870.43), 5.59
M 201.24815 24 (237.99, 885.42, 0.00), (473.01, 233.21), 9.03
M 201.78761 11 (436.54, 671.10, 0.00), (608.48, 57.89), 4.29
M 203.86291 13 (844.00, 159.91, 0.00), (822.04, 92.00), 0.98
M 213.80111 15 (603.79, 707.90, 0.00), (263.85, 793.73), 0.15
M 215.72109 27 (928.10, 762.16, 0.00), (981.72, 936.50), 4.24
M 219.78140 28 (604.72, 674.73, 0.00), (491.77, 314.22), 6.84
M 221.52754 18 (967.86, 870.43, 0.00), (425.89, 229.07), 0.26
M 223.88013 14 (477.24, 62.63, 0.00), (507.68, 88.33), 2.44
M 225.51228 29 (561.14, 404.87, 0.00), (139.09, 796.26), 3.36
M 231.13948 8 (646.92, 886.29, 0.00), (334.53, 310.69), 7.11
M 237.05385 17 (564.19, 483.32, 0.00), (281.28, 619.82), 8.79
M 240.19085 14 (507.68, 88.33, 0.00), (374.60, 319.59), 2.46
M 241.29761 16 (15.93, 380.53, 0.00), (896.94, 556.12), 5.97
M 242.17577 0 (495.60, 914.71, 0.00), (245.25, 460.04), 8.68
M 254.90732 21 (380.16, 993.98, 0.00), (725.73, 20.20), 4.14
M 255.05776 12 (985.90, 377.39, 0.00), (961.47, 289.21), 1.31
M 258.70808 27 (981.72, 936.50, 0.00), (735.23, 225.53), 7.56
M 258.85736 2 (952.46, 420.58, 0.00), (712.87, 232.82), 4.86
M 272.78462 17 (281.28, 619.82, 0.00), (602.81, 142.95), 2.07
M 275.02356 28 (491.77, 314.22, 0.00), (220.96, 330.66), 7.52
M 276.69154 13 (822.04, 92.00, 0.00), (103.92, 215.17), 0.16
M 278.04743 24 (473.01, 233.21, 0.00), (537.44, 882.38), 0.94
M 279.37122 9 (468.57, 136.70, 0.00), (844.43, 630.66), 9.10
M 280.34717 26 (841.79, 265.85, 0.00), (903.33, 212.10), 2.86
M 284.02408 10 (180.95, 35.83, 0.00), (655.48, 19.74), 0.24
M 301.97039 0 (245.25, 460.04, 0.00), (795.86, 593.79), 1.76
M 308.91650 26 (903.33, 212.10, 0.00), (930.85, 792.97), 0.26
M 311.10876 28 (220.96, 330.66, 0.00), (642.49, 669.41), 5.61
M 321.45961 2 (712.87, 232.82, 0.00), (340.23, 31.40), 0.93
M 323.20312 8 (334.53, 310.69, 0.00), (319.88, 201.76), 4.52
M 324.76111 12 (961.47, 289.21, 0.00), (622.93, 85.15), 2.63
M 333.59164 22 (870.57, 6.25, 0.00), (47.95, 318.17), 0.98
M 347.51529 8 (319.88, 201.76, 0.00), (203.01, 524.49), 2.41
M 347.54682 9 (844.43, 630.66, 0.00), (534.05, 509.54), 0.45
M 348.52432 14 (374.60, 319.59, 0.00), (769.16, 959.21), 1.74
M 350.28285 11 (608.48, 57.89, 0.00), (222.03, 42.78), 6.13
M 358.23895 27 (735.23, 225.53, 0.00), (359.12, 193.41), 1.78
M 391.80709 16 (896.94, 556.12, 0.00), (124.62, 788.77), 3.69
M 396.76170 29 (139.09, 796.26, 0.00), (357.93, 371.53), 4.34
M 407.54990 28 (642.49, 669.41, 0.00), (966.54, 536.50), 0.91
M 413.38318 11 (222.03, 42.78, 0.00), (946.65, 349.85), 3.35
M 418.42083 3 (87.38, 757.35, 0.00), (398.93, 12.94), 4.92
M 431.40811 20 (241.06, 724.79, 0.00), (734.51, 926.02), 7.66
M 474.95389 12 (622.93, 85.15, 0.00), (280.66, 416.46), 1.41
M 490.17544 8 (203.01, 524.49, 0.00), (568.60, 626.22), 2.60
M 500.97186 20 (734.51, 926.02, 0.00), (51.61, 752.53), 0.32
M 504.76312 21 (725.73, 20.20, 0.00), (66.22, 985.61), 4.58
M 506.77880 29 (357.93, 371.53, 0.00), (501.96, 700.72), 6.26
M 541.19951 1 (23.42, 571.04, 0.00), (194.90, 646.41), 6.75
M 551.29541 17 (602.81, 142.95, 0.00), (557.27, 942.86), 6.46
M 564.22038 29 (501.96, 700.72, 0.00), (169.65, 215.97), 3.22
M 568.93747 1 (194.90, 646.41, 0.00), (344.46, 248.93), 4.19
M 569.81150 27 (359.12, 193.41, 0.00), (571.94, 789.81), 3.52
M 582.33057 3 (398.93, 12.94, 0.00), (671.59, 853.03), 0.59
M 599.66747 4 (248.97, 214.46, 0.00), (87.17, 921.59), 6.41
M 610.38289 16 (124.62, 788.77, 0.00), (379.18, 706.64), 8.78
M 623.59285 0 (795.86, 593.79, 0.00), (925.93, 547.39), 2.47
M 636.08598 8 (568.60, 626.22, 0.00), (451.42, 313.18), 6.20
```

```
M 640.83149 16 (379.18, 706.64, 0.00), (612.21, 10.49), 3.93
M 648.51645 11 (946.65, 349.85, 0.00), (197.87, 966.71), 3.26
M 670.34597 1 (344.46, 248.93, 0.00), (922.19, 550.92), 7.91
M 675.26446 17 (557.27, 942.86, 0.00), (780.65, 857.74), 8.94
M 679.58439 0 (925.93, 547.39, 0.00), (394.97, 727.91), 2.55
M 689.97125 8 (451.42, 313.18, 0.00), (929.16, 811.73), 0.60
M 702.01899 17 (780.65, 857.74, 0.00), (73.90, 489.34), 5.82
M 712.91032 4 (87.17, 921.59, 0.00), (584.44, 696.31), 7.76
M 746.96673 29 (169.65, 215.97, 0.00), (977.82, 347.74), 5.28
M 749.52022 27 (571.94, 789.81, 0.00), (244.71, 410.32), 4.96
M 752.74301 1 (922.19, 550.92, 0.00), (864.41, 139.90), 3.04
M 760.17873 21 (66.22, 985.61, 0.00), (645.61, 356.62), 3.16
M 778.16275 2 (340.23, 31.40, 0.00), (638.59, 975.22), 8.86
M 779.79579 14 (769.16, 959.21, 0.00), (819.03, 814.00), 4.38
M 783.23274 4 (584.44, 696.31, 0.00), (488.33, 381.18), 3.84
M 793.06429 28 (966.54, 536.50, 0.00), (736.23, 947.84), 0.47
M 812.72082 12 (280.66, 416.46, 0.00), (17.66, 699.86), 8.14
M 814.87130 14 (819.03, 814.00, 0.00), (805.97, 95.43), 7.61
M 827.86765 16 (612.21, 10.49, 0.00), (266.99, 169.61), 8.26
M 839.03098 17 (73.90, 489.34, 0.00), (348.50, 333.57), 1.64
M 850.60155 27 (244.71, 410.32, 0.00), (311.86, 272.74), 8.04
M 860.24662 12 (17.66, 699.86, 0.00), (684.78, 591.79), 6.33
M 869.09857 4 (488.33, 381.18, 0.00), (423.41, 319.41), 5.84
M 869.64546 27 (311.86, 272.74, 0.00), (185.07, 329.24), 2.17
M 873.90594 16 (266.99, 169.61, 0.00), (940.90, 307.90), 2.90
M 884.44348 4 (423.41, 319.41, 0.00), (656.72, 160.15), 5.70
M 889.47694 1 (864.41, 139.90, 0.00), (324.22, 640.06), 9.84
M 889.89624 2 (638.59, 975.22, 0.00), (296.51, 965.33), 6.17
M 899.15235 0 (394.97, 727.91, 0.00), (877.56, 976.16), 6.70
M 902.17727 29 (977.82, 347.74, 0.00), (833.78, 211.06), 7.78
M 909.37184 14 (805.97, 95.43, 0.00), (869.96, 103.62), 3.47
M 927.69019 29 (833.78, 211.06, 0.00), (81.27, 630.29), 9.61
M 927.98854 14 (869.96, 103.62, 0.00), (286.87, 906.70), 0.67
M 928.60904 6 (299.08, 861.64, 0.00), (275.32, 844.08), 6.65
M 933.05423 6 (275.32, 844.08, 0.00), (675.59, 17.67), 0.29
M 933.52693 27 (185.07, 329.24, 0.00), (765.05, 902.76), 8.18
M 934.00052 4 (656.72, 160.15, 0.00), (8.63, 71.53), 4.85
M 945.34253 2 (296.51, 965.33, 0.00), (776.15, 980.53), 6.09
M 945.68388 11 (197.87, 966.71, 0.00), (930.16, 487.93), 0.76
M 964.28529 1 (324.22, 640.06, 0.00), (173.23, 484.67), 1.96
M 967.05557 12 (684.78, 591.79, 0.00), (138.04, 843.50), 6.81
M 969.91372 24 (537.44, 882.38, 0.00), (565.54, 425.42), 9.68
M 980.12968 0 (877.56, 976.16, 0.00), (547.92, 393.38), 1.32
D 1000.000000000 _5_ IFQ  END 0 AODV 44 [13a 5 5 800] -------
[25:255 8:255 26 23] [0x4 8 [18 4] 9.000000] (REPLY)
```

LAMPIRAN 4 FILE AWK

```
# AWK Script for Packet Delivery Calculation for OLD Trace Format

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

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

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

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


END {
 printf "Packet Sent :%d",sendLine;
 printf "\nPacket Received :%d",recvLine;
 printf     "\nPacket Loss :%d" ,sendLine-recvLine;
 print "\nPacket Loss :" ((sendLine-recvLine)/sendLine)*100 " %";
 print "Packet Delivery Ratio :" (recvLine/sendLine)*100 " %";


}


# The BEGIN section, as noted above, is unnecessary.
{
  event = $1
  time = $2
  node_id = $3
  pkt_size = $8
  level = $4

  if (level == "AGT" && event == "s" && $7 == "cbr") {
    sent++
# Note the change in the next line. This initializes the startTime
with the first encountered "time" value.
    if (!startTime || (time < startTime)) {
      startTime = time
    }
  }

  if (level == "AGT" && event == "r" && $7 == "cbr") {
    receive++
    if (time > stopTime) {
      stopTime = time
    }
    recvdSize += pkt_size
  }
}
```

```
END {
  printf("Average Throughput[kbps] =
%.2f\tStartTime=%.2f\tStopTime = %.2f\n", (recvdSize/(stopTime-
startTime))*(8/1000),startTime,stopTime);
}

BEGIN {
seqno = -1;

count = 0;

}

{

if($4 == "AGT" && $1 == "s" && seqno < $6) {

seqno = $6;

}

#end-to-end delay

if($4 == "AGT" && $1 == "s") {

start_time[$6] = $2;

} else if(($7 == "cbr") && ($1 == "r")) {

end_time[$6] = $2;

} else if($1 == "D" && $7 == "cbr") {

end_time[$6] = -1;

}

}

END {

for(i=0; i<=seqno; i++) {

if(end_time[i] > 0) {

delay[i] = end_time[i] - start_time[i];

count++;

}

else

{

delay[i] = -1;
```

```
}
}
for(i=0; i<count; i++) {
if(delay[i] > 0) {
n_to_n_delay = n_to_n_delay + delay[i];
}
}
n_to_n_delay = n_to_n_delay/count;
print "End-to-End Delay = " n_to_n_delay * 1000" ms";
}
```