TESIS – TE142599

ANALISA KINERJA *AODV BACKUP ROUTING* PADA SISTEM KOMUNIKASI VMES SEBAGAI ALAT KOMUNIKASI KAPAL NELAYAN

MASDUKIL MAKRUF
2213203010

DOSEN PEMBIMBING
Dr. Ir. Achmad Affandi, DEA
Dr. Istas Pratomo, ST, MT.

PROGRAM MAGISTER
BIDANG KEAHLIAN TELEKOMUNIKASI MULTIMEDIA
JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2015

TESIS – TE142599

ANALISA KINERJA *AODV BACKUP ROUTING* PADA SISTEM KOMUNIKASI VMES SEBAGAI ALAT KOMUNIKASI KAPAL NELAYAN

MASDUKIL MAKRUF
2213203010

DOSEN PEMBIMBING
Dr. Ir. Achmad Affandi, DEA
Dr. Istas Pratomo, ST, MT.

PROGRAM MAGISTER
BIDANG KEAHLIAN TELEKOMUNIKASI MULTIMEDIA
JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2015

TESIS – TE142599

PERFORMANCE ANALYSIS OF AODV BACKUP ROUTING ON VMES COMMUNICATION SYSTEM AS A COMMUNCATION TOOL OF FISHING VESSEL

MASDUKIL MAKRUF
2213203010

Supervisors
Dr. Ir. Achmad Affandi, DEA
Dr. Istas Pratomo, ST, MT.

MAGISTER PROGRAME
MULTIMEDIA TELECOMMUNICATION
ELECTRICAL ENGINEERING DEPARTMENT
FACULTY OF INDUSTRIAL TECHNOLOGY
SEPULUH NOPEMBER INSTITUTE OF TECHNOLOGY
SURABAYA
2015

Tesis ini disusun untuk memenuhi salah satu syarat memperoleh gelar
Magister Teknik (MT)
Di
Institut Teknologi Sepuluh Nopember

Oleh:

Masdukil Makruf
NRP. 2213203010

Tanggal Ujian : **17 Juni 2015**
Periode Wisuda : **September 2015**

Disetujui oleh:

1. **Dr. Ir. Achmad Affandi, DEA**
   NIP. 196510141990021001 **(Pembimbing I)**

2. **Dr. Istas Pratomo, ST, MT.**
   NIP. 197903252003121001 **(Pembimbing II)**

3. **Dr. Ir. Wirawan, DEA**
   NIP. 196311091989031011 **(Penguji)**

4. **Eko Setidjadi, ST. MT., Ph.D**
   NIP. 197210012003121002 **(Penguji)**

5. **Dr. Ir. Titiek Suryani, MT.**
   NIP. 196411301989032001 **(Penguji)**

Direktur Pasca Sarjana,

Prof. Dr. Ir. Adi Soeprijanto, MT.
NIP. 19640405 199002 1 001

ANALISA KINERJA *AODV BACKUP ROUTING* PADA SISTEM KOMUNIKASI VMES SEBAGAI ALAT KOMUNIKASI KAPAL NELAYAN

Nama Mahasiswa : Masdukil Makruf
NRP : 2213203010
Pembimbing I : Dr. Ir. Achmad Affandi, DEA
Pembimbing II : Dr. Istas Pratomo, ST, MT

ABSTRAK

Indonesia sebagian besar wilayahnya terdiri dari wilayah maritim. Wilayah laut yang luas tentunya memiliki potensi yang besar dan dapat dimanfaatkan oleh banyak kalangan khusunya para nelayan. Dibutuhkan sistem telekomunikasi yang memadai untuk memaksimalkan potensi yang ada. Sistem telekomunikasi alternatif yang dikembangkan sebelumnya untuk kapal nelayan dikenal dengan *Vessel Messaging System (VMeS)*. VMeS merupakan sistem komunikasi yang digunakan untuk bertukar informasi antar kapal atau ke *basestation* di darat dengan menggunakan sistem *ad hoc*. *Ad hoc* yang diadopsi pada VMeS adalah *Mobile Ad hoc Network (MANET)* dimana masing-masing *node* (kapal nelayan) dapat bergerak secara acak. MANET membutuhkan protokol *routing* untuk memandu pengiriman paket data. Protokol *routing* yang digunakan adalah AODV. AODV sudah mampu memberikan mekanisme *routing* yang tepat pada komunikasi VMeS, akan tetapi masih sering terjadi *route break* saat terjadi mobilitas node. Untuk mengatasi masalah ini, digunakan protokol AODV *Backup Routing* pada komunikasi VMeS. Pada penelitian ini, dilakukan analisa terhadap kinerja protokol *routing* AODV-BR pada komunikasi VMeS dengan melakukan simulasi pada *Network Simulator* 2.35 dan *testbed* menggunakan protokol *Medium Access Control* 802.11. Dari hasil analisa simulasi menggunakan *Network Simulator 2* didapatkan rata-rata *delay* 1869 ms untuk AODV-BR sedangkan pada AODV 2546 ms pada variasi data 128 KB dengan 4 *hop*. Kemudian hasil simulasi dibandingkan dengan hasil pengujian pada *testbed*, didapatkan rata-rata delay 4916 ms untuk AODV-BR sedangkan AODV 5724 ms pada skenario yang sama.

Kata Kunci: *MANET, AODV, Backup Routing, VMeS*

PERFORMANCE ANALYSIS OF AODV BACKUP ROUTING ON VMES COMMUNICATION SYSTEM AS A COMMUNCATION TOOL OF FISHING VESSEL

Name	: Masdukil Makruf
NRP	: 2213203010
Supervisor	: Dr. Ir. Achmad Affandi, DEA
Co-Supervisor	: Dr. Istas Pratomo, ST, MT

ABSTRACT

Indonesian territory consists largely of maritime territory. Vast sea area certainly has great potential and can be used by many people but especially the fishermen. Alternative telecommunications system needs to maximize the potential. Alternative telecommunications systems previously developed to fishing boats known as Vessel Messaging System (VMeS). VMeS is a communication system that is used to exchange information between ships or to a base station on land by using ad hoc system. Ad hoc VMeS was adopted on Mobile Ad hoc Network (MANET) where each node (fishing boat) can move randomly. MANET requires a routing protocol to guide the delivery of data packets. Routing protocol used is AODV. AODV has been able to provide the appropriate routing mechanisms on communication VMeS, but still common route break occurs when the node mobility. To resolve this problem, use Backup Routing AODV protocol on communication VMeS. In this study, an analysis of the performance of the routing protocols AODV-BR on VMeS communication by performing simulations on Network Simulator 2.35 and testbed using the 802.11 Medium Access Control Protocols. From the analysis of simulation using Network Simulator 2 obtained an average delay of 1869 ms for AODV-BR whereas in AODV 2546 ms at 128 KB of data variation with 4 hops. Then the simulation results compared with the results of testing on testbed, obtained an average delay 4916 ms for AODV-BR AODV 5724 ms while at the same scenario.

Keyword: *MANET, AODV, Back-up Routing, VMeS*

KATA PENGANTAR

Dengan Nama Allah Yang Maha Pengasih lagi Maha Penyayang.

Segala puja dan puji syukur kepada Allah SWT atas segala rahmat dan karunia yang telah dilimpahkan, sehingga penulisan tesis dengan judul :

"ANALISA KINERJA *AODV BACKUP ROUTING* PADA SISTEM KOMUNIKASI VMES SEBAGAI ALAT KOMUNIKASI KAPAL NELAYAN"

dapat diselesaikan dengan baik. Buku tesis ini disusun untuk memenuhi salah satu syarat memperoleh gelar Magister pada Program Studi Teknik Elektro, Bidang Keahlian Telekomunikasi Multimedia, Institut Teknologi Sepuluh Nopember.

Pada kesempatan ini penulis sampaikan terima kasih yang sedalam-dalamnya kepada :

1. Kedua orangtuaku Ayahanda Hanari dan Ibunda Samarni tercinta yang telah mendidik penulis dari kecil hingga menjadi orang sukses.
2. Bapak Dr. Ir. Achmad Affandi, DEA dan Dr. Istas Pratomo, ST., MT. atas bimbingan, kesabaran dan pendorong semangat dalam menyelesaikan tesis ini.
3. Bapak Dr. Ir. Suwadi, MT., selaku dosen wali yang telah membimbing selama perkuliahan.
4. Bapak Dr. Ir. Wirawan, DEA., Bapak Eko Setidjadi, ST., MT., P.hD., dan Ibu Dr. Ir. Titiek Suryani, MT., yang telah sudi menguji dan mengkoreksi isi dan penyusunan buku tesis ini.
5. Bapak dan Ibu dosen S2 dan S1 terima kasih atas bimbingan dan ilmu pengetahuan yang diberikan selama masa perkuliahan.
6. Rekan-rekan S2 dan S1 di lab Jaringan Telekomunikasi B301, lab Pengolahan Sinyal Digital B304, dan lab Antena dan Propagasi 306, terima kasih atas kebaikan dan kerjasamanya dalam penelitian ini.

7. Terimakasih khusus untuk saudara Afredo Agung Randita dan sudara Fanush Sofi Akbar yang telah membantu dan mengarahkan proses penyusunan tesis ini.
8. Terimakasih khusus untuk Sarindani Oktarina yang telah memberikan dukungan penuh baik secara emosi, mental, dan dukungan semangatnya pada saat proses pembuatan tesis ini.

Penulis menyadari bahwa dalam penulisan thesis ini masih jauh dari sempurna, untuk itu demi perbaikan dan penyempurnaan tesis, maka kritik dan saran sangat diharapkan. Besar harapan penulis bahwa buku thesis ini dapat memberikan informasi dan manfaat bagi pembaca pada umumnya dan mahasiswa Jurusan Teknik Elektro pada khususnya.

Surabaya, 20 Juni 2015

Penulis

Masdukil Makruf
2213203010

DAFTAR ISI

DAFTAR TABEL

BAB 3

BAB 4

!!-- Halaman ini sengaja dikosongkan --!!

DAFTAR GAMBAR

BAB 2

BAB 3

BAB 4

RIWAYAT HIDUP

Masdukil Makruf, lahir di Pamekasan, 02 Juni 1990, merupakan putra pertama dari pasangan Bpk Hanari dan Ibu Samarni.

Menyelesaikan pendidikan jenjang S1

Di Universitas Islam Madura Pamekasan

Jurusan Teknik Informatika

Lulus tahun 2012

Terdaftar sebagai mahasiswa Program Pasca Sarjana

Program Strata Dua Tahun 2013 pada Bidang Studi Telekomunikasi Multimedia

di Institut Teknologi Sepuluh Nopember Surabaya

Email: masdukil.makruf13@mhs.ee.its.ac.id

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Negara Kesatuan Republik Indonesia (NKRI) sebagian besar terdiri dari kepulauan yang dikelilingi oleh lautan. Potensi laut Indonesia yang melimpah ruah dapat dimanfaatkan oleh banyak kalangan khususnya para nelayan. Teknologi telekomunikasi dan informasi sangat membantu terhadap kinerja manusia, yang difungsikan sebagai sarana informasi baik untuk kepentingan keamanan maupun lainnya. Pada wilayah darat, melakukan komunikasi jarak jauh sudah didukung oleh banyak infrastruktur telekomunikasi sehingga memudahkan masyarakat untuk merasakan manfaat dari teknologi ini degan mudah. Beda halnya dengan wilayah laut, yang tidak mungkin dapat membangun infrastruktur tetap (lokasi fix).

Minimnya infrastruktur telekomunikasi pada wilayah laut mengakibatkan susahnya para nelayan untuk melakukan koordinasi, mendapatkan atau mengirim informasi disaat terjadi suatu insiden. Sebenarnya sudah ada alat telekomunikasi untuk wilayah maritim yang memanfaatkan layanan satelit, namun hal ini membutuhkan *budget* yang sangat besar sehingga tidak memungkinkan para nelayan tradisional dapat memanfaatkan alat tersebut [1].

Saat ini, sudah ada penelitian di Lab Jaringan Telekomunikasi ITS tentang *Vessel Messagging System (VMeS)* yang didesain secara *lowcost* dengan harapan dapat memberikan solusi atas keterbatasan alat komunikasi bagi nelayan. VMeS merupakan alternatif solusi untuk alat komunikasi kapal nelayan dibawah ukuran 30 GT. VMeS merupakan pengembangan teknologi dari *Vessel Monitoring System (VMS)* yang digunakan untuk memonitor kapal-kapal penangkap ikan untuk mencegah penangkapan ikan secara ilegal. Kebutuhan sistem komunikasi bagi nelayan merupakan suatu hal yang sangat vital. Dengan adanya sistem komunikasi seperti VMeS, akan memberikan kemudahan pada nelayan untuk memberikan dan mendapatkan informasi dari pusat kontrol atau dari nelayan lain yang juga sedang melakukan pelayaran. VMeS dapat dimanfaatkan oleh nelayan untuk memberikan laporan kepada pusat kontrol mengenai situasi dan kondisi lingkungan, atau

dimanfaatkan untuk menerima informasi dari pusat kontrol misalnya mengenai informasi lokasi yang strategis, peringatan, dan batas-batas wilayah [2].

Mobile Ad Hoc Networks (MANETs) merupakan teknologi telekomunikasi yang sudah dikembangkan penelitiannya untuk mengatasi sulitnya infrastruktur pada suatu wilayah terutama pada wilayah laut. MANET juga merupakan salah satu teknologi yang dikembangkan pada VMeS. MANET tidak mebutuhkan infrastruktur terpusat dikarenakan setiap *node*-nya dirancangan sebagai *router* bergerak secara acak yang dilengkapi dengan *transceiver wireless*. Untuk memungkinkan terjadinya komunikasi antar *node* dibutuhkan protokol *routing* untuk memetakan jalur pengiriman paket data [3] [4].

Adhoc On-demand Distance Vector (AODV) merupakan salah satu protokol *routing* reaktif berbasis *on-demand* dari salah satu ketiga protokol reaktif lainnya yaitu DSDV dan DSR. AODV memiliki performasi yang lebih baik dari pada protokol reaktif yang lain berdasarkan konsumsi delay yang lebih sedikit [5]. AODV yang memiliki ciri utama menjaga *timer-based state* pada setiap node sesuai dengan penggunaan tabel routing. Jalur yang tercatat dalam tabel *routing* akan kadaluarsa apabila tidak ada permintaan. AODV memiliki komponen *Route Discovery* dan *Route Maintenance* sebagai proses pembuatan jalur. *Route Discovery* berupa pesan *Route Request (RREQ)* dan *Route Reply (RREP)*, sedangkan *Route Maintenance* berupa pesan *Route update* dan *Route Error (RRER)* [6]. Protokol AODV lebih efektif dibandingkan protokol MANET lainnya dikarenakan AODV sangat menjaga kualitas jalur dengan meng-*update* tabel *routing* secara periodik sehingga cocok untuk diterapkan pada *node* yang memiliki mobilitas tinggi [7].

Pengembangan penelitian terkait protokol routing AODV sudah banyak dilakukan, salah satunya oleh S.J Lee dan M. Gerla dari University of California – Los Angeles yang telah mengembangkan sekema *flooding* pada AODV menjadi AODV *Backup Routing* (AODV-BR) [8], mereka mengemukakan dua alasan atas dikembangkannya AODV menjadi AODV-BR. Pertama, AODV-BR akan menggunakan jalur alternatif untuk mengirimkan paket data jika jalur terputus, yang mana AODV tradisional membuangnya dikarenakan AODV bukan protokol *multipath*. Kedua, ketika terdapat multi jalur alternatif, ia membuat redundansi dan

meningkatkan kualitas jumlah data transmisi. AODV-BR dalam proses pembuatan jalur alternatif memasrahkan sepenuhnya pada proses penyebaran pesan RREP dan tidak dibutuhkan pesan tambahan lainnya. Dengan teknik seperti ini AODV-BR lebih mampu menstabilkan koneksi jalur dibandingkan dengan AODV [9].

Permasalahan ini menjadi urgen pada MANET dikarenakan sering terjadi putusnya jalur *(route break)* akibat mobilitas *node*. Hal ini dipicu oleh topologi jaringan yang dinamis disebabkan *node* yang selalu bergerak sewaktu-waktu dengan arah yang tidak bisa ditentukan. Putusnya jalur akan berpengaruh pada jumlah waktu yang dibutuhkan untuk pengiriman paket, jumlah kesuksesan pengiriman paket, dan kualitas dari protokol *routing* itu sendiri. Peneltian terdahulu, AODV sudah berhasil disimulasikan dan diuji coba pada VMeS. Namun, dikarenakan sering terjadinya *route break* akibat AODV tradisional merupakan protokol *unipath*, maka sangat perlu untuk menerapkan AODV dengan skema perbaikan *Backup Routing* untuk mengatasi permasalahan tersebut sehingga apabila terjadi putusnya jalur akibat mobilitas node, node sumber atau *next hop* tidak perlu membangun jalur kembali melainkan memanfaatkan *route backup* yang sudah terbentuk saat *route discovery* [8].

Pada penelitian ini, akan menganalisa kinerja protokol routing AODV-BR pada komunikasi VMeS dengan melakukan simulasi menggunakan perangkat lunak *Network Simulator 2.35* dan melakukan uji coba kinerja protokol *routing* pada perangkat *testbed* berupa laptop dengan menggunakan protokol *Medium Access Control (MAC)* 802.11 untuk mengetahui efektifitas kinerja protokol *routing* AODV-BR pada lingkungan VMeS. Untuk dapat menganalisa kinerja dari protokol *routing* AODV-BR pada komunikasi VMeS ini, dibutuhkan hasil pengukuran terhadap performa jaringan berdasarkan beberapa parameter *Quality of Service (QOS)* yaitu *average delay* dan *average throughput* agar dapat menganalisa dan mengambil sebuah kesimpulan.

1.2 Rumusan Masalah

Berdasarkan latar belakang diatas dapat dirumuskan permasalahan sebagai berikut:

1. Bagaimana mensimulasikan dan menguji kinerja protokol *routing* AODV-BR pada komunikasi VMeS?
2. Bagaimana analisa perbandingan terhadap kinerja protokol routing AODV tradisional dengan AODV *Backup Routing* pada komunikasi VMeS?
3. Apakah protokol *routing* AODV-BR dapat memberikan kontribusi mekanisme *routing* yang tepat untuk VMeS?
4. Bagaimana mengimplementasikan *routing* AODV-BR pada VMeS?

1.3 Batasan Masalah

Batasan masalah dari penelitian terhadap analisa kinerja protokol routing AODV-BR pada Komunikasi VMeS, adalah sebagai berikut:

1. AODV-BR disimulasikan dan diimplementasikan pada *testbed* dengan menggunakan *Medium Access Control* 802.11.
2. Protokol yang digunakan adalah AODV *Backup Routing*
3. *Software* simulasi yang digunakan adalah *Network Simulator* versi 2.35
4. Implementasi AODV-BR diuji pada *testbed* laptop menggunakan AODV-UU.
5. Membandingkan kinerja AODV-BR dengan AODV.
6. Untuk menguji *backup routing* diasumsikan sebagian node bergerak dinamis dengan kecepatan tertentu tanpa keluar dari *caverage area*, dan beberapa node pada jalur utama bergerak hingga terjadi *route break.*
7. Pengujian unjuk kerja pada testbed dilakukan di daerah pesisir dan laut.
8. Parameter yang diukur untuk analisa kinerja adalah *average delay* dan *average throughput.*

1.4 Tujuan Penelitian

Tujuan yang akan dicapai dari penelitian ini adalah sebagai berikut:

1. Menguji kinerja protokol AODV-BR dengan menambahkan skema *Backup Routing* untuk mengembangkan *routing* AODV yang sesuai dengan kebutuhan sistem.
2. Untuk menganalisa kinerja protokol routing AODV-BR pada VMeS agar teknologi MANET yang ada pada komunikasi VMeS lebih efektif kinerjanya dengan melakukan simulasi dan *testbed* pada laptop terlebih dahulu sebagai acuan pokok rekomendasi untuk pengembangan protokol pada penelitian VMeS selanjutnya.
3. Dengan adanya pengembangan skema routing pada protokol routing MANET di VMeS ini dapat memberikan kontribusi terhadap sebagian pengembangan teknologi VMeS yang masih membutuhkan pengembangan lebih besar.

1.5 Manfaat Penelitian

Dari usulan penelitian tesis ini diharapkan dapat memberikan kontribusi keilmuan bagi pembaca pada umumnya, bagi penulis pada khususnya mengenai kinerja protokol *routing* AODV *Backup Routing* pada VMeS serta berharap dapat dijadikan bahan acuan untuk penelitian selanjutnya dalam tahapan-tahapan atau proses pengembangan protokol *routing* pada sistem komunikasi VMeS.

1.6 Metodologi

Metodologi penelitian dalam menganalisa kinerja protokol *routing* AODV-BR pada Komunikasi VMeS ini meliputi tahapan-tahapan berikut:

1. Perumusan Masalah

   Langkah awal yang dilakukan untuk memulai penelitian pada Tesis ini adalah studi literatur. Dari hasil studi litertur, dapat dirumuskan beberapa permasalah sebagai materi penelitian pada tesis ini.

2. Perancangan Sistem

   Pada tahap ini dilakukan perancangan untuk menganalisa kinerja protokol *routing* AODV *Backup Routing* pada sistem komunikasi VMeS menggunakan *software Network Simulator* 2 dan merancang *testbed* protokol *routing* menggunakan laptop untuk melakukan uji coba secara riil.

3. Implementasi Sistem

   Dari hasil perancangan sistem tersebut, dibuat implementasi jaringan *ad hoc* pada VMeS diantaranya mengimplementasikan secara simulasi dan *testbed* menggunakan laptop.

4. Pengujian dan Analisa

   Dari hasil implementasi, akan dilakukan uji coba kinerja dengan mengukur beberapa parameter QoS seperti *delay* dan *throghput* berdasarkan banyak variasi *hop* yang sudah ditentukan. Kemudian, hasil pengukuran akan dianalisa untuk mendapatkan sebuah kesimpulan.

5. Kesimpulan dan Saran

   Berisi kesimpulan tentang hasil yang sudah didapatkan, dan memberikan saran untuk penelitian selanjutnya.

1.7 Relevansi

Hasil yang diperoleh dari tesis ini diharapkan dapat memberikan manfaat sebagai berikut:

1. Menghasilkan tulisan ilmiah mengenai Analisa Kinerja Protokol AODV-BR pada komunikasi VMeS.
2. Memberikan rekomendasi dalam memilih mekanisme *routing* yang tepat untuk penelitian selanjutnya.
3. Memberikan ulasan mengenai penggunaan AODV pada *Medium Access Control (MAC)* dengan protokol AX.25.

BAB 2
TINJAUAN PUSTAKA

2.1 Sistem Komunikasi VMeS

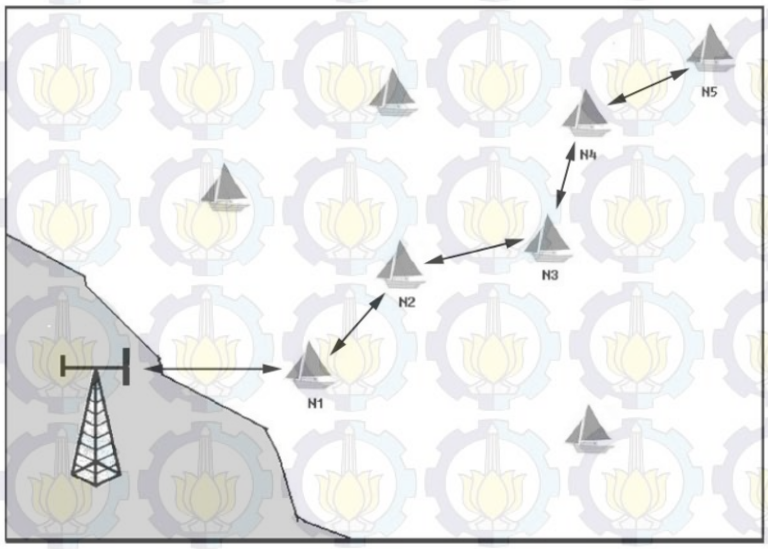
Sistem komunikasi VMeS *(Vessel Messaging System)* merupakan salah satu dari projek penelitian yang saat ini sedang dikembangkan di Laboratorium Jaringan Telekomonukasi Elektro ITS. VMeS merupakan sistem komunikasi laut berbasis *wireless* yang terdiri dari satu *node gateway* dan node tersebar secara *mobile* dan acak. VMeS dikembangkan dari sistem komunikasi setelit yang disebut dengan VMS *(Vessel Monitoring System)*. VMS dirancang khusus untuk memonitoring posisi kapal pada wilayah laut, namun sistem ini dianggap tidak efisien apabila digunakan sebagai alat komunikasi pada kapal nelayan, dikarenakan sistem ini membutuhkan biaya yang sangat besar dan konsumsi daya yang juga besar. Oleh sebab itu, timbulah inisiatif untuk membuat sebuah sistem komunikasi alternatif untuk kapal nelayan yang murah dan terjangkau [2].

VMeS dirancang dengan menggunakan gelombang radio pada frekuensi VHF [10] untuk menyampaikan informasi secara dua arah. Informasi yang dikirimkan dapat berupa data kapal, data kondisi lingkungan di sekitar kapal, dan data peringatan tanda bahaya. Dengan demikian para nelayan bisa memberikan informasi tentang kejadian-kejadian di lingkungan sekitar kapal yang sedang mereka gunakan, misalnya terjadi kecelakaan serta adanya tindak pencurian ikan oleh kapal nelayan asing. Informasi yang diberikan oleh nelayan tersebut bisa diteruskan ke pihak yang berwenang sehingga apabila terjadi hal-hal yang tidak diinginkan dapat cepat direspon dan ditanggulangi.

Banyak hal yang sudah dikembangkan dalam VMeS, baik dari segi sistem komunikasi, teknik pengiriman data, bahkan perangkat yang digunakanpun disesain agar semakin murah. Sistem komunikasi yang digunakan pada VMeS adalah sistem *ad hoc*, dikarena sistem *ad hoc* mampu melakukan komunikasi dengan memanfaatkan jangkauan perangkat lain sehingga data yang hendak dikirimkan dapat sampai pada tujuan [1].

Alasan dipilihanya *ad hoc* sebagai sistem komunikasi yang dimanfaatkan oleh sistem VMeS ialah dikarenakan VMeS dirancang khusus untuk diterapkan pada suatu wilayah yang tidak dapat dibangun infrastruktur komunikasi sama sekali yaitu pada wilayah laut. Tidak ada pilihan sistem komunikasi lain yang dapat digunakan selain menggunakan sistem *ad hoc*, dikarekan jika menggunakan layanan satelit, perangkatnya dapat dibilang mahal apalagi untuk kalangan nelayan. Semua sistem yang diterapkan dalam sistem *ad hoc* di VMeS adalah sama dengan sistem *ad hoc* yang ada pada sistem komunikasi lain, misalnya dari segi teknik *routing* dari sistem ini. Semua teknik *routing* yang digunakan dalam VMeS juga memiliki peran penting dalam melakukan komunikasi, seperti pemetaan jalur pengiriman paket data berupa informasi dan lainnya [11].

Node yang tersebar dalam sistem VMeS berupa kapal nelayan sama halnya dengan *node* yang bergerak dalam sistem *ad hoc*. Pergerakan yang terjadi secara acak merupakan salah satu teknik dalam *ad hoc* yang memiliki *node* bergerak, hal tersebut biasa disebut dengan *Mobile Ad Hoc Network (MANET)*. MANET memiliki topologi jaringan dinamis yang selalu berubah tanpa dapat ditentukan. MANET diterapkan pada VMeS, disesuiakan dengan kondisi lingkungan pada VMeS yang memiliki node bergerak dan susahnya mendirikan infrastruktur fix dan terpusat pada wilayah laut [4].



Gambar 2.1 Ilustrasi VMeS dengan Teknologi *Ad Hoc* [1]

2.2 *Mobile Adhoc Network (MANET)*

Jaringan *wireless* terdiri dari dua model yaitu *fixed wireless*, dan *mobile wireless*. Jaringan *fixed wireless* tidak mendukung *mobility* karena infrastrukturnya terbangun dengan paten pada suatu titik lokasi, dan kebanyakan jaringan ini adalah *point to point*. Lain halnya dengan *mobile wireless* yang dapat bergerak namun tetap bisa berkomunikasi meskipun dengan topologi yang dinamis dan sangat dibutuhkan oleh user bergerak. Jaringan *mobile* terbagi menjadi dua kategori utama, yaitu jaringan yang memiliki infrastruktur dan jaringan yang tidak memiliki infrastruktur.

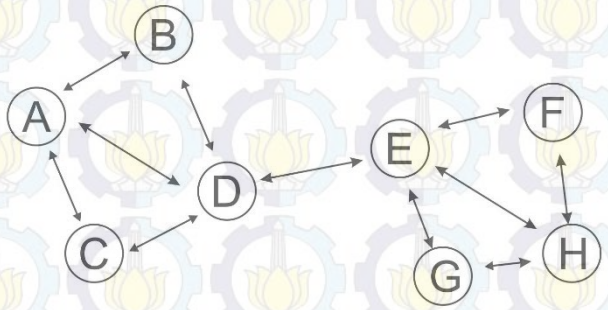
Ad hoc Network merupakan desentralisasi dari jaringan nirkabel, disebut *ad hoc network* karena tidak bergantungnya jaringan ini terhadap adanya infrastruktur, seperti *router* dalam jaringan kabel ataupun *Access Point* pada jaringan nirkabel. Dalam *Ad hoc network*, setiap node memiliki tugas me-*routing* data kepada *node* lain. Jadi, penentuan *node* mana yang mengirimkan data dibuat secara dinamis berdasarkan konektivitas dari jaringan itu sendiri. Sifat desentralisasi, protokol *routing* dinamis, dan mudah untuk diterapkan menjadikan *ad hoc network* cocok untuk diimplementasikan disaat jaringan terpusat tidak dapat digunakan (seperti ditengah laut atau situasi darurat seperti bencana alam atau konflik militer). Dalam beberapa tahun terakhir, banyak pakar jaringan mengalihkan perhatian mereka dari jaringan terpusat seperti jaringan internet dan telepon seluler dan berpindah ke *ad-hoc network* [3].

Mobile ad hoc Network (MANET) adalah jenis sistem komunikasi *ad hoc* yang mana setiap *node*-nya bebas bergerak secara independen dan acak serta topologi jaringannya terus berubah-ubah sesuai dengan kondisi mobilitas. Setiap perangkat *(node)* harus meneruskan paket yang dikirimkan sesuai tujuan alamat kendatipun tidak ada kaitannya dengan *node* itu sendiri, sebab itulah setiap *node* pada jaringan *ad hoc* atau *mobile ad hoc* disebut sebagai *router* dikarenakan memiliki tugas untuk me-*routing* paket yang dikirim oleh *sender* ke *destination*. Tantangan utama dalam membangun MANET adalah melengkapi setiap perangkat untuk terus menjaga informasi yang diperlukan untuk merutekan lalu lintas data secara benar. Jaringan pada MANET dapat beroperasi sendiri atau dapat dihubungkan ke jaringan internet yang lebih besar. Salah satu konsep dasar yang

ada pada MANET merupakan konsep dasar jaringan ad hoc yang memliki tabel *routing* dan berada di atas layer *Data Link* tepatnya pada layer *Network*.

Dalam jaringan *mobile ad hoc*, tidak terdapat *base-station*, dan tidak ada pengawas yang memantau kinerja jaringan secara keseluruhan. Node yang digunakan di jaringan *mobile ad hoc* akan aktif dan mencoba untuk menentukan berapa banyak *node* aktif lainnya yang berada dalam jangkauan transmisi. Secara bersama-sama, *node* kemudian mengumpulkan informasi apapun yang dibutuhkan untuk melakukan tugas secara kolektif.

Dalam pengertian lain, jaringan *ad hoc* adalah jaringan bersifat sementara tanpa bergantung pada infrastruktur yang ada dan bersifat independen. Jaringan *Ad Hoc* adalah jaringan *wireless* yang terdiri dari kumpulan *mobile node (mobile station)* yang bersifat dinamik dan spontan, dapat diaplikasikan di mana pun tanpa menggunakan jaringan infrastruktur (seluler ataupun PSTN) yang telah ada. Contoh *mobile* node adalah *notebook*, PDA dan ponsel. Jaringan ad hoc disebut juga dengan s*pontaneous network* atau disebut *MANET (Mobile Ad hoc NETwork)* [12].



Gambar 2.2 Struktur Dasar Jaringan *Ad Hoc* [12]

Saat ini *mobile ad hoc network* sudah banyak diterapkan dalam segala jenis aspek kebutuhan telekomunikasi. Jaringannya yang fleksibel menjadikan jaringan *ad hoc* banyak diminati untuk dikembangkan. Berikut ini adalah contoh-contoh aspek yang sudah menggunakan dan menerapkan sistem *mobile ad hoc network* di lapangan:

- Operasi militer, seperti yang telah diujicobakan kawasan pertempuran di Sudan. Dengan jaringan *ad hoc*, mempermudah untuk akses informasi antar personil militer.

- Komersial, jaringan *ad hoc* dapat digunakan pada situasi *emergency* atau upaya penyelamatan *(rescue operation)*, seperti banjir atau gempa bumi dan *entertainment* seperti acara *live music*.
- Jaringan yang cepat tersedia dengan menggunakan *notebook* untuk menyebarkan dan berbagi informasi di antara user seperti dalam konferensi atau ruang kuliah.
- *Personal Area Network*, untuk jarak pendek *(short distance)* lebih kurang 10 m, *ad hoc network* secara mudah berkomunikasi antar bermacam peralatan (seperti PDA, laptop dan telepon seluler) dengan laju data yang rendah.
- VMeS, digunakan untuk komunikasi antar node yang tersebar di wilayah laut yang membutuhkan sistem *ad hoc* untuk dapat berkomunikasi dengan pusat kontrol di darat, atau sebaliknya dikarenakan hanya dengan sistem *ad hoc*-lah VMeS dapat bekerja pada wilayah yang sama sekali tidak ada infrastruktur telekomunikasi.

Setiap sistem telekomunikasi memiliki kelebihan dan kekurangan tersendiri dalam penerapannya, masing-masing sistem berfokus pada kinerjanya masing-masing. Demikian juga dengan sistem jaringan *mobile ad hoc* yang juga memiliki kelebihan dan kekurangan dalam setiap penerapannya. Banyak faktor yang menyebabkan adanya kelebihan dan kekurangan dari pada sistem telekomunikasi seperti MANET, misalnya seperti kondisi lingkungan yang tidak dapat diprediksi menjadikan sistem telekomunikasi dipaksa menyesuaikan dengan fungsinya masing-masing. Adapun kelebihan dalam penerapan jaringan *mobile ad hoc* adalah sebagai berikut:

1. Tidak memerlukan dukungan *backbone* infrastruktur sehingga mudah diimplementasikan dan sangat berguna ketika infrastruktur tidak ada ataupun tidak berfungsi lagi.
2. *Mobile node* yang selalu bergerak, dapat mengakses informasi secara *real time* ketika berhubungan dengan *mobile node* lain, sehingga pertukaran data dan pengambilan keputusan dapat segera dilaksanakan.
3. Fleksibel terhadap suatu keperluan tertentu karena jaringan ini memang bersifat sementara.

4. Dapat direkonfigurasi dalam beragam topologi baik untuk jumlah user kecil hingga banyak sesuai dengan aplikasi dan instalasi *(scalability).*

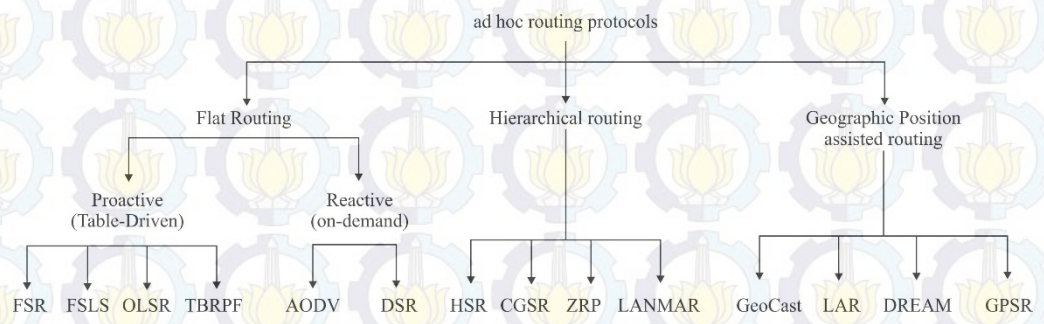
Jika kelebihan pada jaringan *mobile ad hoc* dapat dikemukakan, sudah tentu jaringan *mobile ad hoc* memiliki kekurangan atau kerugian jika menggunakan jaringan *mobile ad hoc*.

1. *Packet loss* (rugi-rugi paket) akan terjadi bila transmisi mengalami kesalahan *(error).*
2. Seringkali terjadi *disconnection*, karena tidak selalu berada dalam area cakupan.
3. *Bandwidth* komunikasi yang terbatas.
4. *Lifetime* daya yang singkat.
5. Kapasitas kemampuan jangkauan *mobile node* yang terbatas dan bervariasi [3].

2.3 Protokol Routing dalam MANET

Protokol *routing* dalam teknologi *Mobile ad Hoc Network (MANET)* memiliki peran penting dalam melakukan komunikasi antar node. Protokol *routing* yang memiliki peran pada lapisan 3 *(network layer)* dalam lapisan OSI ini sudah menjadi tulang punggung dalam suksesnya melakukan komunikasi pada lingkungan *ad hoc*. Protokol *routing* difungsikan sebagai standar komunikasi atau aturan-aturan yang harus berlaku dengan teknik tertentu. Setiap protokol *routing* memiliki algoritma *routing* tersendiri sesuai standar yang dibuat oleh penemu dari teknik *routing* tertentu.

Telah banyak protokol *routing* yang dikembangkan pada komunkasi *ad hoc* ini. Mereka mengelompokkan protokol *routing ad hoc* menjadi tiga bagian utama, yaitu; *flat rooting protocols, Hierarchical routing protocols,* dan *Location-based routing protocols*.



Gambar 2.3 Pengelompokan Protokol Routing dalam *Ad Hoc* [13]

Flat routing protocols didalamnya terdapat dua kategori, yaitu proaktif dan reaktif. Perbedaan dasar dari ketiga jenis kategori ini adalah; proaktif berbasis *table-driven* sedangkan reaktif berbasis *on-demand*. Kedua kategori tersebut memiliki bagian-bagian jenis protokol *routing ad hoc* yang tentunya memiliki algoritma masing-masing [13]. Berikut ini adalah pembagian protokol *routing* dari *flat routing protocols*:

- Proaktif
  - ✓ *FSR (Fisheye State Routing)*
  - ✓ *FSLS (Fuzzy Sighted Link State)*
  - ✓ *OLSR (Optimized Link State Routing)*
  - ✓ *TBRPF (Topology Broadcast Based on Reverse Path Forwarding)*
- Reaktif
  - ✓ *AODV (Ad Hoc On-demand Distance Vector routing)*
  - ✓ *DSR (Dynamic Source Routing)*

Hierarchical routing protocols terbentuk saat ukuran jaringan semakin meningkat. Flat *routing* tidak mampu dikarekan terjadi *overhead*. *Hierarchical routing protocols* akan membagi jaringan menjadi beberapa grup dan memberikan fungsi yang berbeda antara grup yang di dalam jaringan dan grup di luar jaringan. *Hierarchical routing protocols* juga memiliki beberapa bagian protokol di dalamnya yaitu; *CGSR (Clusterhead-Gateway Switch Routing), HSR (Hierarchical State Routing), ZRP (Zone Routing Protocol), dan LANMAR (Landmark Ad Hoc Routing Protocol).*

Location-based routing protocols merupakan bagian terakhir dari kategori utama pembagian protokol routing dalam jaringan *ah hoc*. Protokol yang dikelompokkan pada *Location-based routing protocols* ini mengandalkan *basic* informasi lokasi pada setiap *node*. Hal tersebut bisa diperoleh dengan menggunakan sensor dan GPS. Protokol ini memanfaatkan *geographical forwarding* untuk mengirim paket data. Adapun protokol routing yang terkelompok dalam kategori ini adalah; *LAR (Location-Aided Routing), DREAM (Distance Routing Effect Algorithm for Mobility), GPSR (Greedy Perimeter Stateless Routing)* dan *GLS (Grid Location Service* [3].

2.4 *Adhoc on-Demand Distance Vector (AODV)*

2.4.1 Definisi AODV

Ad hoc On-demand Distance Vector (AODV) merupakan *routing* protokol reaktif berbasis *on-demand* yang dirancang untuk jaringan *mobile ad hoc*. Algoritma yang dikembangkan dalam protokol routing AODV oleh C. Perkins dan M. B. Royer ini merupakan implementasi dari *Distributed Bellman-Ford (distance vector)* yang digunakan oleh smua protokol *routing Distance Vector (DV)* pada *mobile ad hoc network* termasuk AODV ini [6].

AODV menciptakan suatu jalur dengan menggunakan *Route Request (RREQ)* dan *Route Reply (RREP)*. Ketika *source node* menginginkan suatu jalur menuju *destination node* tetapi belum mempunyai jalur yang benar, maka *source node* akan menginisialisasi *route discovery process* untuk menemukan jalur ke *destination node*. *Source node* akan mem-*broadcast* paket RREQ menuju node tetangganya *(neighbor)*. RREQ paket berisi *source address*, *destination address*, *hop counter*, *source* dan *destination sequence number*, dan *broadcast* ID. Nilai *Broadcast* ID akan bertambah satu setiap suatu *source node* mengirimkan RREQ yang baru dan digunakan sebagai identifikasi sebuah paket RREQ [14].

Jika *node* yang menerima RREQ memiliki informasi jalur menuju *destination node*, maka *node* tersebut akan mengirim paket RREP kembali menuju *source node*. Tetapi jika tidak mengetahui maka *node* tersebut akan mem-*broadcast*

ulang RREQ ke *node* tetangganya setelah menambahkan nilai *hop counter*. *Node* yang menerima RREQ dengan nilai *source address* dan *broadcast* ID yang sama dengan RREQ yang diterima sebelumnya akan membuang RREQ tersebut. *Source sequence number* digunakan oleh suatu *node* untuk memelihara informasi yang valid mengenai *reverse path* (jalur balik) menuju ke *source node*. Pada saat RREQ mengalir menuju *node* tujuan yang diinginkan, dia akan menciptakan *reverse path* menuju ke *node*, setiap *node* akan membaca RREQ dan mengidentifikasi alamat dari *node* tetangga yang mengirim RREQ tersebut.

Ketika *destination node* atau *node* yang memiliki informasi jalur menuju *destination* menerima RREQ maka *node* tersebut akan membandingkan nilai *destination sequence number* yang ia miliki dengan nilai destination *sequence number* yang ada di RREQ. Jika nilai *destination sequence number* yang ada di RREQ lebih besar dari nilai yang dimiliki oleh *node* maka paket RREQ tersebut akan di-*broadcast* kembali ke *node* tetangganya, sebaliknya jika nilai *destination sequence number* yang ada di *node* lebih besar atau sama dengan nilai yang ada di RREQ maka *node* tersebut akan mengirim *route reply (RREP)* menuju *source node* dengan menggunakan *reverse path* yang telah dibentuk oleh RREQ. *Intermediate* node yang menerima RREP akan meng-*update* informasi *timeout* (masa aktif rute) jalur yang telah diciptakan. Informasi rute *source* ke *destination* akan dihapus apabila waktu pada tabel *lifetime*-nya habis [7] [14].

Selama jalur yang ditentukan tetap aktif, jalur tetap akan terus dipertahankan. Sebuah rute dianggap aktif apabila masih ada paket data secara periodik terkirim dari *node* sumber ke tujuan disepanjang jalur yang aktif ini. Setelah sumber berhenti mengirimkan paket data, akan terjadi *link timeout* dan akhirnya akan dihapus paket tersebut dari tabel *routing node* perantara. Jika suatu link terputus sementara rute aktif, node yang mengalami kerusakan akan mengirim pesan *Route Error (RERR)* ke node sumber untuk menginformasikan bahwa jalur ini tidak terjangkau seperti yang diilustrasikan pada Gambar 2.6. Setelah menerima pesan RERR, apabila ternyata node sumber masih tetap membutuhkan rute ini, maka *node* sumber akan mengulang kembali pembuatan jalur selama paket yang

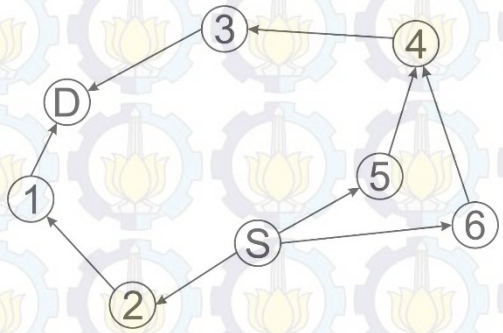
hendak dikirim masih belum kadaluarsa. Untuk prosedur dalam jalur *multicast* dan *broadcast* juga ditetapkan dengan cara yang sama [6].

Dalam algoritma *routing* AODV, setiap *node* diharuskan menjaga tabel *routing* pada masing-masing *node*. Adapun judul dari *field* pada tabel *routing* AODV adalah sebagai berikut:

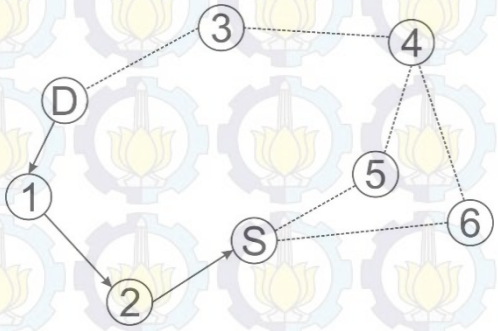
1. *Destination IP Address* berisi alamat IP dari *node* tujuan yang digunakan untuk menentukan jalur.
2. *Destination Sequence Number* bekerjasama untuk menentukan rute
3. *Next Hop* atau "loncatan" berikutnya, bisa berupa tujuan atau node tengah, *field* ini dirancang untuk meneruskan paket ke node tujuan.
4. *Hop Count* Jumlah hop dari alamat IP sumber sampai ke alamat IP tujuan.
5. *Lifetime* waktu dalam milidetik yang digunakan untuk node menerima RREP.
6. *Routing Flags* merupakan status dari sebuah jalur, *up* (valid), *down*(tidak valid) atau sedang diperbaiki.

AODV menggunakan tabel routing dengan satu entri untuk setiap tujuan, dan tanpa menggunakan *routing* sumber. AODV mempercayakan semua pemetaan jalur pada tabel routing pada setiap node-nya untuk menyebarkan *Route Reply (RREP)* kembali ke sumber dan secara sekuensial akan mengarahkan paket data menuju ketujuan. AODV juga menggunakan *sequence number* untuk menjaga setiap tujuan agar didapat informasi *routing* yang terbaru dan untuk menghindari *routing loops*. Semua paket yang diarahkan membawa *sequence number* ini ke *node* tujuan.

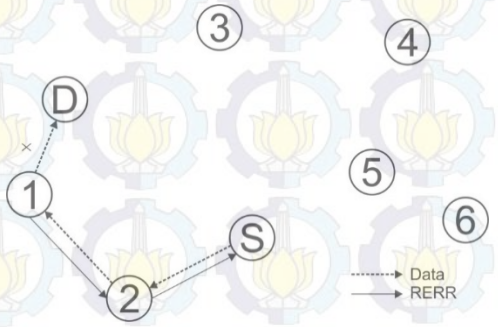
Penemuan jalur *(path discovery)* atau *route discovery* diinisialisasi dengan menyebarkan *Route Reply (RREP)*, seperti terlihat pada Gambar 2.5. Ketika RREP menjelajahi node, ia akan secara otomatis membangun jalur. Jika sebuah *node* menerima RREP, maka *node* tersebut akan mengirimkan RREP lagi ke node sumber berdasarkan *destination sequence number*. Pada proses ini, yang akan dicek pertama kali oleh setiap *node* saat menerima pesan RREP adalah *destination sequence number* ini.



Gambar 2.4 Pengiriman Pesan RREQ Saat Proses *Route Discovery* [7]



Gambar 2.5 Jalur Terdekat yang Dilintasi Oleh Pesan *route reply* (RREP) [7]



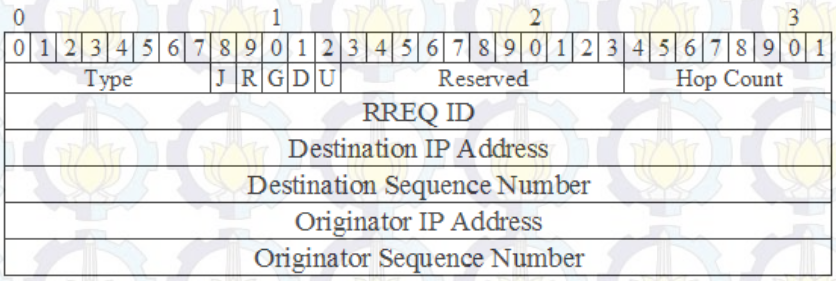
Gambar 2.6 Proses Pengiriman RERR ke *Source* Saat Terjadi *Route Break* [7]

2.4.2 Format Pesan dalam AODV

Protokol routing *Ad hoc On-demand Distance Vector (AODV)* memeliki paket pesan untuk memulai proses setiap segmennya dalam membuat jalur. Paket pesan tersebut antara lain adalah *Route Request (RREQ)*, *Route Reply (RREP)*, dan *Route Error (RERR)*. Setiap pesan memiliki format tersendiri, sesuai dengan standar yang sudah dibuat oleh C. Perkins dan EM. Royer yang tentunya juga sudah disesuaikan dengan kebutuhan *routing* pada AODV itu sendiri [6]. Adapun fotmat-format pesannya berdasarkan standar RFC 3561 adalah sebagai berikut:

2.4.2.1 Format Pesan *RREQ (Route Request)*

Berikut ini adalah format pesan RREQ yang digunakan saat proses *route discovery*.



Gambar 2.7 Format Pesan *RREQ (Route Request)* [14]

Format pesan RREQ yang diilustrasikan seperti Gambar 2.7, berikut adalah penjelasan dari format pesan tersebut:

Type : 1

J : *Join flag*, disediakan untuk *multicast*

R : *Repair flag*, disediakan untuk *multicast*

G : *Gratuitous RREQ flag*; menunjukkan apakah RREQ percuma untuk di-*unicast* ke node yang ditentukan dalam IP address tujuan.

D : *Destination only flag*; mengindikasikan bahawa hanya tujuan yang memungkinkan dapat menanggapi RREQ ini

U : *Unknown sequence number*; yang menunjukkan nomor urut tujuan tidak diketahui.

Reserved : Jika yang dikirim bernilai 0; maka pesan RREQ diabaikan pada penerima.

Hop Count : Jumlah *hop* dari IP *originator* ke node yang menangani *request*.

RREQ ID : Adalah sebuah nomor unik yang mengidentifikasi RREQ tertentu dari organisasi IP pada *node-node*.

Destination IP Address adalah alamat IP tujuan yang diinginkan

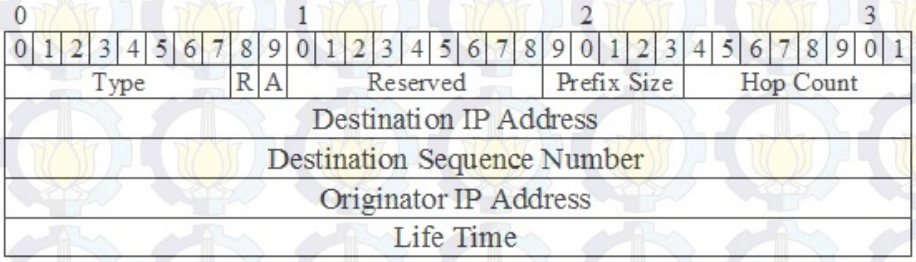
Destination Sequence Number adalah nomor urut terakhir yang diterima oleh *originator* untuk jalur apapun menuju destinasi.

Originator IP Address adalah alamat IP yang berasal dari *node* yang mengirim RREQ

Originator Sequence Number adalah nomor urut *node* yang akan digunakan dari *node* tujuan ke *node* sumber.

2.4.2.2 Format Pesan *RREP (Route Reply)*

Berikut ini adalah format pesan RREP yang digunakan saat proses membentuk jalur dari tujuan ke penerima.



Gambar 2.8 Format Pesan *RREP (Route Reply)* [14]

Format pesan RREP diilustrasikan seperti diatas, berikut adalah penjelasan dari format pesan diatas:

Type : 2

R : *Repair Flag*, disediakan untuk multicast

A : *Acknowledgment required* (pesan ACK)

Reserved : Jika yang dikirim berniali 0; maka diabaikan pada penerima.

Prefix Size : Jika nol, ukuran *Prefix* 5 bit menetapkan bahwa *hop* berikutnya yang ditunjukkan dapat digunakan untuk setiap node dengan awalan *routing* yang sama (seperti yang didefinisikan oleh *PrefixSize*) sebagai tujuan yang diminta

Hop Count : Jumlah *hop* dari alamat IP *originator* ke Alamat IP destinasi. Untuk rute permintaan *multicast* ini menunjukkan jumlah hop keanggota *multicast* yang mengirim RREP

Destination IP Address berisi alamat IP tujuan dan rute yang mana saja yang disediakan

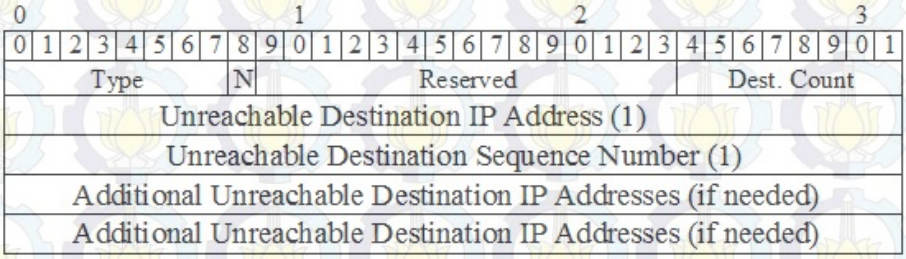
Destination Sequence Number merupakan nomor urut tujuan terkait.

Originator IP address adalah Alamat IP yang berasal dari node yang mengirim RREQ

Life Time adalah masa berlakunya rute yang terhitung dalam milidetik

2.4.2.3 Format Pesan *RERR (Route Error)*

Berikut ini adalah format pesan RRER yang digunakan apabila terjadi *route break* dari *source* ke *destination*.



Gambar 2.9 Format Pesan *RRER (Route Error)* [14]

Format pesan RRER diilustrasikan seperti pada Gambar 2.9, yang berisi bidang-bidang berikut ini:

Type : 3

N : *No delete flag* adalah yang mengatur kapan sebuah *node* akan melakukan perbaikan pada lokal *link*, dan node awal seharusnya tidak menghapus rute.

Reserved : Jika yang dikirim bernilai 0; maka diabaikan pada penerima.

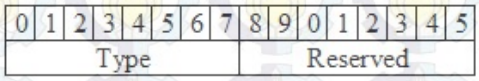
Destination Count adalah jumlah tujuan *unreachable* yang termasuk dalam pesan tersebut; minimal harus 1

Unreachable destination IP address adalah alamat IP tujuan yang tidak terjangkau karena terjadinya *link break (route break).*

Unreachable Destination Sequence Number adalah nomor urut dalam entri tabel routing untuk tujuan yang tercantum dalam *unreachable IP address* tujuan sebelumnya.

2.4.2.4 Format Pesan *Route Reply Acknowledgment (RREP-ACK)*

RREP-ACK merupakan pesan yang digunakan oleh pengirim dalam routing AODV untuk mengetahui apakah paket yang dikirim sudah diterima oleh tujuan atau belum, ini diperlukan agar node pengirim tidak melakukan pembentukan ulang jalur. Adapun standar format pesan *Route Reply Acknowledgement (RREP-ACK)* adalah sebagai berikut:



Gambar 2.10 Format Pesan *RREP-ACK* [14]

Type : 3

Reserved : Jika yang dikirim bernilai 0; maka diabaikan pada penerima.

2.5 Arsitektur Protokol AODV *Backup Routing*

Penelitian demi penelitian terus dilakukan oleh para peneliti untuk menutupi dan untuk memperbaiki kekurangan yang ada pada AODV tradisional, salah satunya adalah AODV *Backup Routing* atau dikenal dengan AODV-BR yang dikembangkan oleh Lee dan Mario Gerla [8], AODV-BR diciptakan untuk menutupi berbagai macam kekurangan yang ada pada AODV tradisional, kekurangan tersebut meliputi:

- Tidak adanya *multiple path* atau jalur alternatif yang dapat menghandel paket data jika terjadi kerusakan pada jalur utama, sehingga harus dilakukan pembentukan jalur kembali.
- Jika terjadi *route break*, data akan di-*drop* akibat tidak adanya jalur alternatif yang dapat dilalui oleh data sampai jalur baru terbentuk kembali.
- Hal ini akan menjadi masalah besar jika pengiriman data memerlukan *real time delivery*, seperti *voice* dan *streaming video* disebabkan delay yang sangat tinggi.

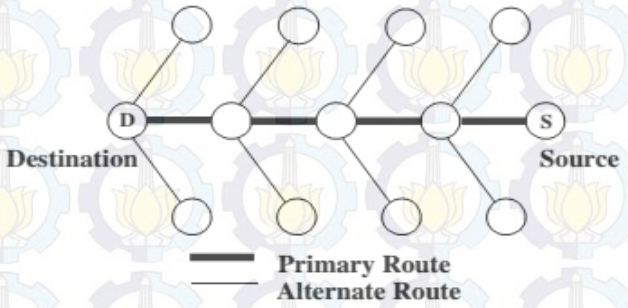
AODV-BR menawarkan sebuah algoritma baru yang dapat mengatasi persoalan-persoalan diatas, yaitu dengan menggunakan struktur *mesh* untuk menyediakan jalur alternatif yang ada pada *on-demand routing protokol* tanpa membentuk pesan tambahan atau dengan kata lain hanya menggunakan tiga pesan utama yang ada pada AODV tradisional yaitu RREP. Jalur alternatif sangat dibutuhkan dalam *ad hoc* karena jaringan *wireless* rentan terjadinya *route break* karena mobilitas *node, signal interference,* dan *packet collision* [9].

AODV-BR memiliki dua konsep utama dalam mengimplementasikan algoritma atau metode yang ditawarkan, yaitu *route construction* dan *Route*

Maintenance [9]. Pada bagian selanjutnya akan dijelaskan mengenai kedua konsep tersebut.

2.5.1 *Route Contruction*

Proses pembentukan AODV-BR tidak melakukan modifikasi terhadap ketiga pesan utama yang ada pada AODV tradisional, AODV-BR hanya memanfaatkan *route request* dan *route reply* yang ada yaitu dengan cara mem-*flooding* RREQ seperti pada AODV tradisional, kemudian RREP yang menuju *source node* akan disebarkan dan dicatat pada tabel *routing* pada masing-masing *node* dengan membuat setiap *node* dapat *overhear* terhadap RREP yang dikirim oleh *destination*, dan pada saat itu juga terbentuklah jalur utama dan jalur alternatif yang disebut struktur *mesh*.



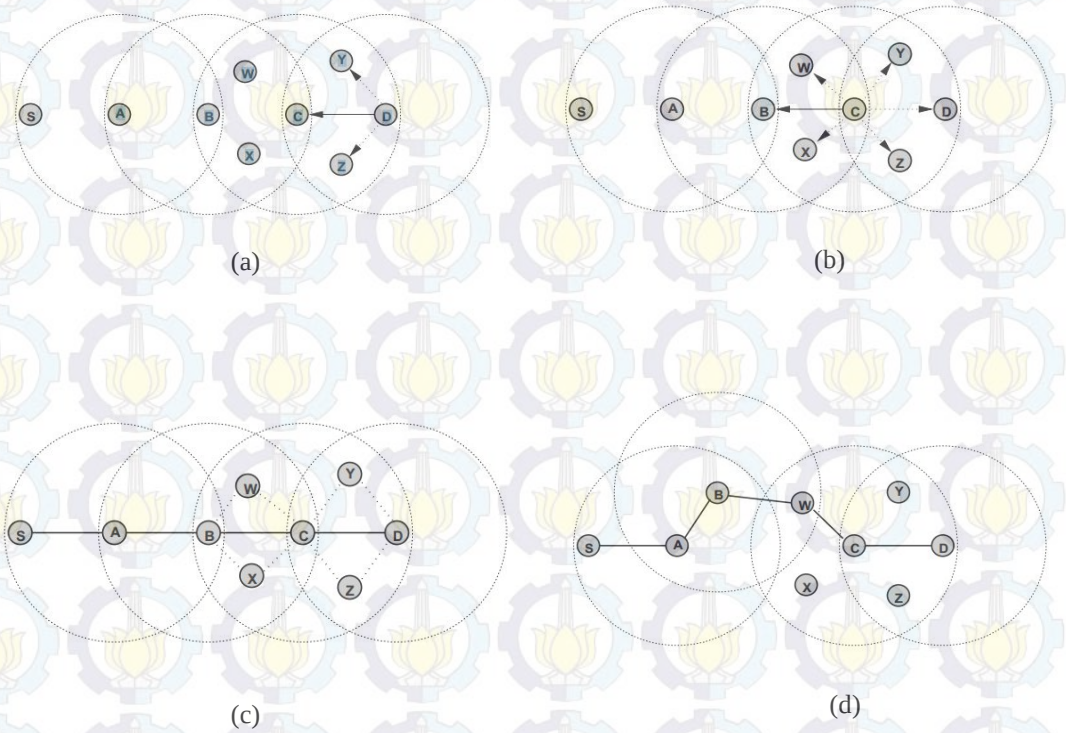
Gambar 2.11 Struktur Mesh yang menyerupai tulang Ikan, terbentuk jalur utama dan jalur alternatif [8]

Skema AODV-BR yaitu dengan memanfaatkan penyebaran *route reply (RREP)* secara *broadcast* yang ada pada AODV tradisional yang nantinya dapat membentuk stuktur *mesh* yang menyerupai tulang ikan, lihat pada Gambar 2.11. Setiap node bisa saja menerima banyak *reply* untuk rute yang sama, namun hanya akan dipilih jalur terbaik dan dimasukkan ke tabel *routing*.

2.5.2 *Route Maintenance*

Paket data dikirim melalui jalur utama sampai terdeteksi terjadinya kerusakan pada jalur utama tersebut misalnya menerima sinyal *feedback link layer* dari MAC *protokol*, tidak menerima *passive acknowledgments*, tidak menerima *hello packets* dalam beberapa waktu atau yang lain, maka satu *hop* akan mem-*broadcast* data dengan *header* yang memberitahukan bahwa *link* rusak dan membutuhkan jalur alternatif. Sedangkan untuk menghindari *looping* saat

menggunakan *alternative route, node* mem-*forward* data jika paket data tidak diterima dari *nexthop* tujuan dan tidak terjadi duplikasi. Ketika sebuah node dari rute utama menerima paket data dari rute alternatif, maka node beroperasi normal dan meneruskan paket ke *hop* berikutnya ketika paket tidak diduplikat. *Node* yang terdeteksi *link break* juga mengirim pesan *Route Error (RERR)* ke node sumber untuk memulai kembali penemuan rute.



Gambar 2.12 Beberapa Konstruksi Jalur dan Penggunaannya: (a) *Node* Mengirim Sebuah RREP, (b) *Node* Meneruskan RREP *multicats*, (c) Jalur Utama dan Jalur Alternatif Ditetapkan, (d) Paket Data Dikirimkan Melalui Jalur Alternatif Bila Jalur Utama Terputus [9]

Di dalam protokol AODV tradisional, jalur yang *time out* akan diperbaharui untuk durasi waktu tertentu jika proses pengiriman data masih berlangsung. Disini menggunakan teknik yang sama untuk waktu keluar dari jalur alternatif. *Node* yang menyediakan jalur alternatif dapat menditeksi paket data. *Node* akan memperbaharui jalur, jika paket tersebut dikirimkan oleh *hop* berikutnya ke tujuan seperti yang ditunjukkan dalam tabel jalur alternatif dan *node* akan menghapus jalur dari tabel jika jalur alternatif tidak diperbaharui pada saat waktu habis.

2.6 Protokol MAC 802.11

Protokol MAC *(Medium Access Control)* merupakan lapisan *Datalink* yang mengatur dan menjaga komunikasi diantara stasiun-stasiun pada jaringan WLAN. Lapisan ini juga berfungsi sebagai pengatur akses protokol ke media fisik jaringan dan mengkoordinasi akses dalam menggunakan kanal radio untuk mempermudah komunikasi melalui media *wireless*.

Pada umumnya, MAC *wireless* LAN 802.11 menggunakan protokol *CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance)*. CSMA/CA dapat menghindari tabrakan paket dengan menggunakan mikanisme *frame* RTS/CTS *(Request to send/ Clear to send)* dan *acknowledgement (ACK)*. RTS/CTS merupakan *frame* yang mengizinkan akses poin untuk mengatur penggunaan media pada sebuah stasiun, maksudnya *frame* RTS digunakan *sender* untuk memberitahukan akan melakukan pengiriman data pada *receiver* setelah mendeteksi medium dalam kondisi *idle*, *frame* CTS digunakan untuk sebagai pesan balasan dari *receiver* untuk memberitahukan *sender* bahwa *receiver* telah menerima RTS dan siap menerima data, sedangkan *frame* ACK dikirim oleh stasiun penerima untuk mengkonfirmasi bahwa paket telah terkirim [15].

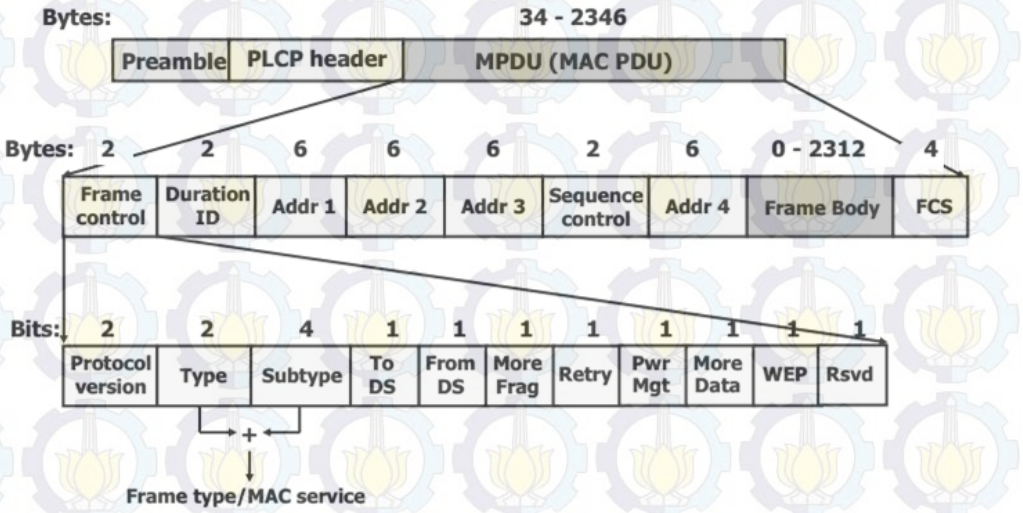
Gambar 2.13 Proses Pengiriman RTS / CTS [11]

Sebuah pesan yang dikirim pada layer *datalink* berupa *frame*, *frame* adalah paket yang dienkoding untuk keperluan transmisi data. Standar WLAN 802.11 mendefinisikan beberapa tipe *frame* yang digunkan pada setiap komunikasi antar station maupun yang digunakan untuk sebagai pengatur dan pengedali *link wireless*.

Setiap *frame* mempunya *header* dengan beberapa *field* yang digunakan diantara *MAC Sublayer* [15].

Terdapat 4 susunan utama dalam wireless LAN MAC melakukan manajemen, dan pengontrolan *frame*, yaitu: *Frame Control, MAC Header, Frame Body,* dan *Frame Check Sequence (FCS).*

Format frame keseluruhan memiliki panjang 2346 Byte pada jaringan wireless LAN 802.11 dapat ditunjukkan pada Gambar 2.14.



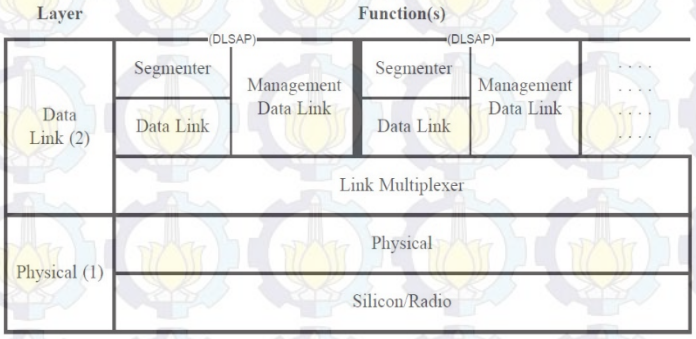
Gambar 2.14 Format *Frame* pada MAC 802.11 [15]

2.7 Protokol Radio AX.25

Protokol adalah sebuah aturan yang mendefinisikan beberapa fungsi yang ada dalam sebuah jaringan komputer, misalnya mengirim pesan, data, informasi dan fungsi lain yang harus dipenuhi oleh sisi pengirim dan sisi penerima agar komunikasi dapat berlangsung dengan benar, walaupun sistem yang ada dalam jaringan tersebut berbeda sama sekali. Protokol ini mengurusi perbedaan format data pada kedua sistem hingga pada masalah koneksi listrik. Standar protokol yang terkenal yaitu OSI *(Open System Interconnecting)* yang ditentukan oleh ISO *(International Standart Organization).*

AX.25 adalah protokol layer 2 (merujuk pada *OSI Layer Reference*) yaitu *Data Link Layer*. Sebagai protokol pada lapisan 2 AX.25 bertanggungjawab untuk menbangun *link connection*, menyediakan prosedur *logic* untuk information transfer, dan *link disconnection.* Sehingga AX.25 cukup lengkap untuk dijadikan

contoh implementasi sebuah protokol [16]. Protokol Amatir X.25 (AX.25) adalah protokol radio turunan dari X.25 yang digunakan dalam jaringan paket radio. Untuk membangun hubungan antara dua buah terminal melalui *physical layer* dan lapisan *data link*. Protokol ini akan bekerja pada dua kondisi transmisi yaitu *half duplex* dan *full duplex*. Selanjutnya dua lapisan yang ada pada protokol ini yaitu *physical layer* dan lapisan *data link* dapat dibagi lagi ke dalam beberapa status keadaan seperti yang ditunjukkan pada Gambar 2.15. keadaan yang dimaksudkan adalah mendefinisikan keadaan suatu *link* komunikasi radio untuk *multi link* .



Gambar 2.15 Keadaan Protokol AX.25 untuk *Multi Link* [12]

Dengan *mengacu* Gambar 2.15 pada protokol AX.25 lapisan paling atas dari *layer* 2 adalah *Data Link Access Point* (DLAP). DLAP merupakan lapisan yang akan menyediakan untuk meneruskan paket data ke *layer* 3. Pada saat terjadi transmisi data maka hubungan antara *data link* diberikan oleh lapisan *data link* dengan menggabungkan antara dua atau lebih DLAP. Kemudian *data link* akan memberikan suatu urutan bit yang dipecah menjadi beberapa blok data yang disebut *frame*.

Format protokol AX.25 pada teknologi *packet radio* ditunjukkan pada Gambar 2.16 dengan memiliki maksimum 256 *byte* dalam satu *frame*. Pada pengiriman data kecepatan tinggi dan aplikasi TCP/IP dilakukan beberapa perubahan sehingga dimungkinkan untuk mengirim lebih dari 256 *byte* data dalam satu *frame*.

Flag	AX.25 Transfer Frame Header (128 bits)				Information Field	Frame-Check Sequence	Flag
	Destination Address	Source Address	Control Bits	Protocol Identifier			
8	56	56	8	8	0-2048	16	8

Gambar 2.16 Format *frame* Protokol Link AX.25 yang Digunakan dalam Komunikasi Paket Radio [16]

Frame AX.25 dimulai dan ditutup oleh *flag byte* yang berisi 01111110. *Adress field* berisi alamat tujuan, alamat pengirim paket dan stasiun-stasiun yang berfungsi sebagai *relay*. Dengan menggunakan stasiun lain sebagai *relay*, maka stasiun yang digunakan sebagai *relay* tersebut dapat mengirimkan data ke tempat tujuan. Hal tersebut dikenal sebagai konsep *digipeater* (*digital repeater*). Pada *control field* berisi identifikasi bentuk *frame* AX.25 yang dikirim. Apakah *frame* ini untuk melakukan koneksi (membuka hubungan komunikasi), koreksi (jika ada *frame* AX.25 yang rusak dalam pengiriman), untuk *broadcast* dan sebagainya. *Packet* ID (PID) digunakan untuk memberitahukan jenis data yang dikrim, apakah data berbentuk teks, binary atau protokol lapisan *network*. *Frame Check Sequense* (FCS) digunakan oleh bagian penerima pada proses pendeteksian kesalahan [17].

Protokol AX.25 dalam komunikasi data radio mempermudah pengguna untuk berkomunikasi data secara langsung dengan menggunakan program *hyperterminal* dan pengguna tidak perlu repot dengan masalah *acknowledgement* karena sudah ditangani oleh *terminal node controller* (TNC). Untuk melakukan komunikasi data yang dapat dikontrol secara langsung oleh *software* lebih fleksibel apabila menggunakan protokol lapisan yang lebih bawah. Sebagian besar TNC mendukung penggunaan *keep it simple and stupid* (KISS) sebagai protokol pada lapisan bawah untuk mengirimkan datagram secara langsung dari komputer/mikrokontroler. KISS *frame* ini sudah dilengkapi dengan proses deteksi kesalahan. KISS *frame* ini juga digunakan untuk komunikasi data secara langsung dengan menggunakan protokol TCP/IP [16].

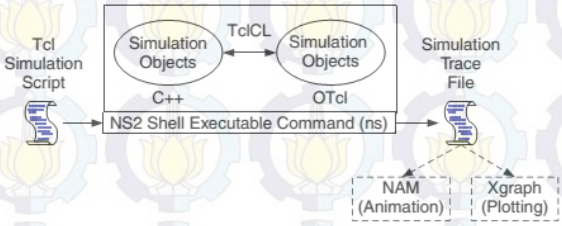
2.8 *Network Simulator 2*

Network Simulator (versi 2), biasa dikenal dengan NS2. Ns merupakan *software* untuk membuat sample simulasi kejadian pergerakan *(event-driven)* yang digunakan untuk mempelajari sifat dinamis dari sebuah jaringan dalam sistem komunikasi, misalnya mensimulasikan fungsi jaringan kabel, nirkabel, dan protokol dalam jaringan seperti algoritma routing, TCP dan UDP dapat disimulasikan menggunakan NS2. Secara umum, NS juga memfasilitasi pengguna misalkan dalam mensimulasikan sebuah protokol dalam jaringan yang sangat sesuai dengan kondisi ideal dan sesuai dengan karakteristiknya [18].

Sejak tahun 1989, NS sudah sangat popular di kalangan komunitas riset dikarenakan kefleksibelannya. Sudah beberapa kali dilakukan revolusi dan revisi terhadap NS disesuaikan dengan keberadaan alat jaringan yang terus berkembang berkat hasil-hasil riset yang terus dilakukan di lapangan, diantaranya adalah yang telah dilakukan oleh *University of California* dan *Cornell University*. Sejak tahun 1995, *Advanced Research Projects Agency (DARPA)* mendukung pengembangan NS melalui proyek *testbed Virtual Internetwork (Vint).* Saat ini *National Science Foundation (NSF)* juga tergabung dalam pengembangan ini, dan masih banyak lagi kelompok peneliti yang mengembangkan NS agar dapat digunakan dalam banyak aspek simulasi.

2.8.1 Arsitektur dasar NS2

Network Simulator (NS) terbangun dari beberapa *script* yang saling memiliki kesenambungan. Bahasa pemprograman yang digunakan untuk membangun NS meliputi TCL, C++, OTCL, NAM, dan Xgraph. Berdasar pada bahasa pemprograman tersebut terbangunlah sebuah *software* simulasi jaringan ini. Kelima bahasa pemprograman tersebut memiliki kesinambungan satu sama lainnya, karena ia akan bekerja secara kolektif untuk satu tujuan. Adapun korelasi dari pemprograman tersebut seperti yang ditunjukkan pada Gambar 2.17.



Gambar 2.17 Arsitektur Dasar pada NS [18]

NS menfasilitasi pengguna untuk merancang sebuah argument atau skenario simulasi, skenario tersebut ditulis dalam *script file* TCL. TCL memungkinkan bagi pengguna untuk membuat animasi dalam simulasi, pengukuran serta plot grafik untuk hasil pengukuran dalam sebuah simulasi [18].

NS terdiri dari dua bahasa pemprograman utama; C++ dan *Object-oriented Tool Command Language* (Otcl). Kedua bahasa pemprogrman tersebut memiliki peran masing-masing. C++ memiliki peran untuk mendefinisikan mekanisme internal dari object simulasi, sedangkan Otcl menyiapkan simulasi dengan merakit dan mengkonfigurasi objek serta penjadwalan peristiwa secara diskrit. C++ dan Otcl digabungkan kinerjanya secara bersamaan menggunakan TclCL [18].

Dalam file TCL, setelah selesai simulasi akan memunculkan file TR atau *trace file* yang berisi *log* kejadian saat simulasi. Semua aktifitas objek saat simulasi terkem secara otomatis dalam *file .tr* tersebut. Dengan *file trace* tersebut kita akan mendapatkan hasil pengujian dari skenario simulasi yang kita buat dalam *file* TCL. Hasil dari *file trace* tersebut bisa kita olah untuk mendapatkan gambaran hasil simulasi dengan menggunakan *tool* yang sudah terpasang pada *ns all in one*, yaitu *Xgraph* dan *Nam*. *Xgraph* digunakan untuk *plot* hasil simulasi kedalam bentuk grafik, sedangkan *Nam* digunakan untuk memberikan gambaran simulasi berupa animasi sesuai dengan skenario yang dibuat pada waktu menuliskan *script* TCL [18].

2.8.2 Pengolahan Data Simulasi

Akhir simulasi pada NS2, akan mengahsilkan keluaran *file* berupa *log* yang mencatat seluruh kejadian saat simulasi, seperti jumlah paket yang dikirim dan diterima, penoromoran node, serta seluruh informasi-informasi yang terjadi saat

simulasi berlangsung. Adapun teknik pengambilan data dari hasil sumulasi adalah sebagai berikut:

2.8.2.1 *File Trace*

File trace merupakan pencatatan seluruh *event* (kejadian) yang dialami oleh node pada saat simulasi berlangsung. Pembuatan *file trace* dilakukan dengan memanggil objek *trace* yang terdapat pada *library* NS. Berikut ini merupakan contoh dari *file trace*.

```
r 1.3556 3 2 ack 40 --------------- 1 3.0 0.0 15 201
+ 2.3556 2 0 ack 40 --------------- 1 3.0 0.0 15 201
- 1.3556 2 0 ack 40 --------------- 1 3.0 0.0 15 201
d 1.35576 2 3 tcp 1000 --------------- 1 0.0 3.0 29 199
```

Gambar 2.18 Contoh *Trace File Output* Simulasi pada Jaringan *Wired*

Contoh file *trace* di atas memilikii format tabel seperti gambar berikut:

Event	Time	From node	To node	Pkt type	Pkt Size	Flags	fid	Scr addr	Dst addr	Seq num	Pkt id

Gambar 2.19 Format Tabel pada File *Trace*

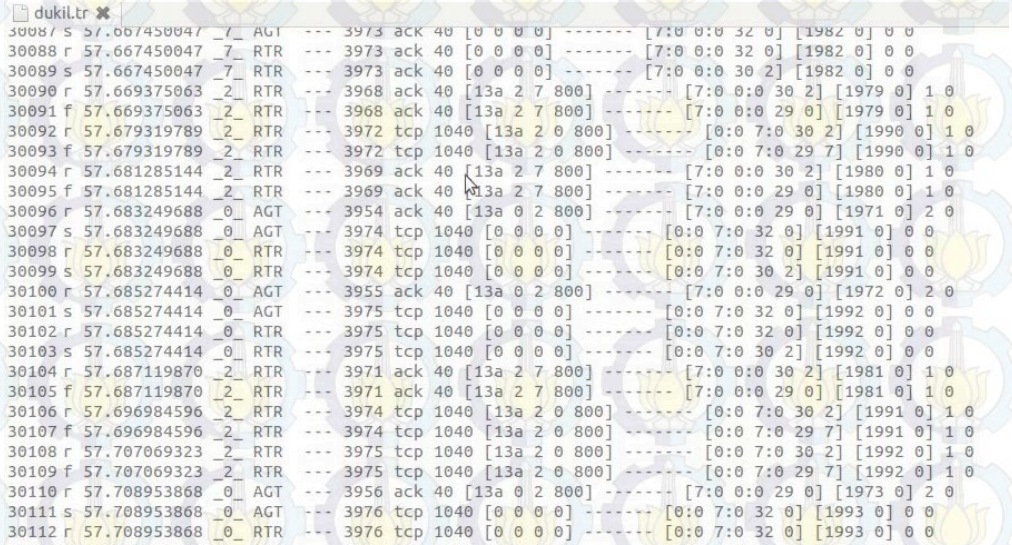
Adapun maksud dari setiap *field* kolom pada Gambar 2.19 diatas adalah sebagai berikut:

- *Event* : Adalah kejadian yang dicatat pada saat simulasi di NS, meliputi:
  r : *receive* (paket yang telah diterima oleh *node dest.*)
  d : *Drop* (sejumlah paket yang mengalami *drop* saat antrian)
  + : *Enter queue* yaitu paket masuk antrian atau keluar dari *node*.
  - : *Leave queue* paket yang keluar dari antrian
- *Time* : Adalah waktu event dalam detik bahkan mili detik
- *From node* : Menyatakan keberadaan paket. Pada saat *event*, paket berada di antrian *from node* atau *to node*..
- *To node* : Menyatakan keberadaan paket
- *Pkt size* : ukuran *(bytes)* dari paket
- *Flag* : Digunakan sebagai penanda. Pada data di atas *flags* tidak digunakan. Adapun macam-macam dari penanda yang dapat digunakan yaitu:

- ✓ E : Digunakan apabila terjadi kongesti *(Congstion Experienced / CE)*
- ✓ N : Digunakan untuk mengindikasi *ECT (ECN-Capable-Transport)* pada *header* IP
- ✓ C : Digunakan untuk *ECN-Echo*
- ✓ A : Digunakan untuk pengurangan *window* kogesti pada *header* TCP
- ✓ P : Untuk prioritas
- ✓ F : Untuk TCP *fast start*

- *Fid* : Merupakan penomoran unik dari tiap aliran data
- *Scr addr* : *Node* asal, memiliki format *node.port*
- *Dst. Addr* : *Node* tujuan, memiliki format *node.port*
- *Seq Num* : Adalah *sequence number* dari paket tersebut
- *Paket id* : Merupakan paket ID dari paket.

Sedangkan *file trace* yang dihasilkan untuk jaringan *wireless*, contohnya seperti Gambar 2.20 berikut:



Gambar 2.20 Contoh *File Trace* Jaringan *Wireless Ad Hoc*

Keterangan:

- *Next Hop Info* berisi *–Hs* dan *– Hd. –Hs* merupakan untuk *node* ini, sedangkan *–Hs* adalah Id untuk *next hop* yang akan melanjutkan paket ke tujuan.

- *Node Property tags* berisi:
  - ✓ Ni : Id pada *node*
  - ✓ Nx : Koordinat X yang dimiliki sebuah *node*
  - ✓ Ny : Koordinat Y yang dimiliki sebuah *node*
  - ✓ Nz : Koorninat Z yang dimiliki sebuah *node*
  - ✓ Ne : Level yang dimiliki *node*
  - ✓ N1 : *Level trace*, seperti AGT, RTP, MAC
- Paket yang dienkapsulasi ke level MAC
  - ✓ Ma : Durasi
  - ✓ Md : Alamat *ethernet* tujuan
  - ✓ Ms : Alamat *ethernet* pengirim
  - ✓ Mt : Tipe dari *ethernet*
- Paket Informasi pada level IP
  - ✓ Is : Alamat pengirim, *port* yang digunakan pengirim
  - ✓ Id : Alamat tujuan dan port yang digunakan
  - ✓ It : Tipe paket
  - ✓ Il : Ukuran paket
  - ✓ If : *Flow Id* (Id jalur)
  - ✓ Ii : ID unik
  - ✓ Lu : Nilai *time to life*

Untuk melakukan analisa terhadap *file trace* yang dihasilkan dari data simulasi, untuk memudahkannya dapat menggunakan *shell programming* atau AWK untuk melakukan penyaringan terhadap informasi parameter yang dibutuhkan saja.

AWK merupakan sebuah program yang biasanya digunakan untuk menfilter teks yang banyak untuk mengambil beberapa yang dibutuhkan saja. AWK dapat pula digunakan untuk mencari bentuk atau model dalam sebuah *file* teks. AWK ini memiliki karakterisrik pemprograman berbasis *shell* atau C. Dengan menggunakan script AWK, dapat memudahkan pemfilteran terhadap data-data yang dibutuhkan saja, misalnya pada tesis ini akan digunakan untuk menggambil data parameter QoS jaringan, seperti *delay* dan *throughput* saja [5].

2.8.2.2 *Parsing*

Parsing merupakan teknik untuk mendapatkan sebuah informasi yang dibutuhkan dari *file trace* yang mencatat seluruh kejadian saat simulasi berlangsung. Untuk men-*parsing file trace*, dibutuhkan *file* dengan *script* tertentu yang disimpan dalam bentuk *awk*. *File awk* digunakan untuk menfilter beberapa data yang dibutuhkan dalam *file trace*, sehingga hanya data-data yang dibutuhkan saja yang dapat dipisahkan. Contoh perintah parsing menggunakan *script awk* ditulis dengan format seperti pada Gambar 2.21 berikut:

```
exec awk -f delay.awk dukil.tr > delay.tr
```

Gambar 2.21 Contoh Perintah *Parsing* pada *File Trace*

Perintah pada Gambar 2.21 dijalankan pada CLI sesuai dengan OS yang digunakan. Jika menggunakan Linux, maka perintah *parsing* tersebut dapat dijalankan di terminal. *Delay.awk* adalah *file* yang berisi *script* parsing terhadap informasi *delay* yang terjadi pada saat simulasi berlangsung yang tersimpan pada *file trace* yang diberi nama *dukil.tr*. Untuk melakukan *parsing*, file awk harus diletakkan pada folder yang sama. Sedangkan *delay.tr* adalah file yang berisi hasil paring berupa *delay* pada saat simulasi berlangsung.

2.9 Pengukuruan *QoS* Jaringan

Untuk mengetahui kualitas dari teknik yang disusun dalam merencanakan suatu sistem, baik sistem simulasi maupun terapan, terdapat beberapa parameter yang dibutuhkan untuk diukur agar dapat menyimpulkan bahwa sebuah sistem dapat dianggap bekerja dengan baik. Dalam sistem jaringan *Mobile Ad Hoc Network (MANET)* terdapat beberapa parameter pengukuran yang dibutuhkan sebagai bahan untuk dapat mengambil kesimpulan, parameter yang diperlukan untuk mengetahui kualitas layanan dalam jaringan salah satunya dengan menggunakan parameter *Delay* dan *Throughput* [5].

2.9.1 Delay

Delay time adalah waktu yang dibutuhkan untuk mengirim sebuah paket data ke node yang lain selama proses simulasi berlangsung. Dari *file trace* yang dihasilkan saat simulasi selesai dapat dihitung *delay time* antara node pengirim dengan node penerima dalam proses transmisi data. *Delay* ini dapat diperoleh dari hasil perhitungan selisih waktu antara waktu paket dikirim dan paket diterima. Penghitungan ini dapat dirumuskan sebagai berikut:

$$waktu\ tunda\ (t) = \frac{Tr-Ts}{Pr} \quad 0 \leq t \leq T \qquad (2.1)$$

Dimana:

Tr : Waktu penerimaan paket
T : Waktu yang digunakan
Ts : Waktu pengiriman paket
t : Waktu pengambilan sampel
Pr : Paket yang diterima (paket)

2.9.2 Throughput

Throughput adalah waktu yang dibutuhkan untuk pengiriman paket dari satu node pengirim ke node tujuan dalam satuan waktu. *Throughput* ada juga yang mendefinisikan sebagai laju data yang terukur pada satuan waktu tertentu. *Throughput* sama halnya dengan konsumsi *bandwith* dimana kecepatan kanal yang diperoleh oleh satu user.

Kendatipun *throughput* memiliki satuan dan rumus yang sama dengan kecepatan laju data, tetapi *throughput* lebih menggambarkan laju data yang sebenarnya pada suatu waktu tertentu dan pada kondisi jaringan internet tertentu yang digunakan untuk mengirim dan menerima suatu data dalam ukuran waktu. Dalam menghitung *throughput* ini dapat menggunakan persamaan berikut:

$$throughput = \frac{Pr}{1\ detik}\ paket\ /\ detik \qquad 0 \leq t \leq T \qquad (2.2)$$

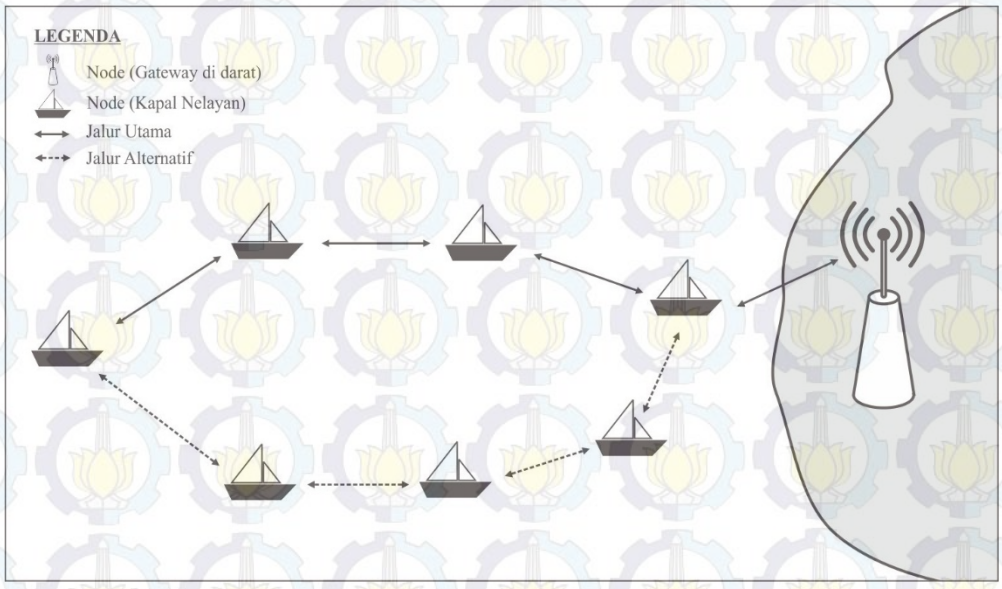
Dimana:

Pr : Paket yang diterima (paket)
T : Waktu yang digunakan (detik)
t : Waktu pengambilan sampel (detik)

BAB 3
METODE PENELITIAN

3.1 Gambaran Umum Sistem

Untuk menganalisa kenerja dari penerapan protokol MANET pada komunikasi VMeS, digunakan protokol manet reaktif yaitu AODV *backup routing* (AODV-BR). AODV-BR merupakan pengembangan dari protokol AODV tradisional yang menawarkan *multiple route* dikarenakan AODV tradisional sering terjadi masalah *route break* saat melakukan komunikasi dan harus melakukan *route discovery* dari awal sehingga membutuhkan waktu yang lebih lama. AODV-BR menawarkan *alternative route* agar pada saat melakukan komunikasi tidak perlu melakukan *route discovery* melainkan memanfaatkan jalur alternatif yang disediakan jika terjadi *route break*. Pemaparan yang jelas untuk skema *routing* dari AODV-BR ini sudah dibahas pada subbab 2.5. Untuk melakukan analisa terhadap kinerja AODV-BR pada VMeS digunakan dua cara, yaitu mensimulasikan kinerja protokol *routing* menggunakan *software Network Simulator 2.35* dan mengimplementasikannya pada testbed menggunakan Laptop untuk menguji kinerja dari protokol AODV-BR secara riil.



Gambar 3.1 Pemanfaatan Jalur Alternatif pada Komunikasi VMeS

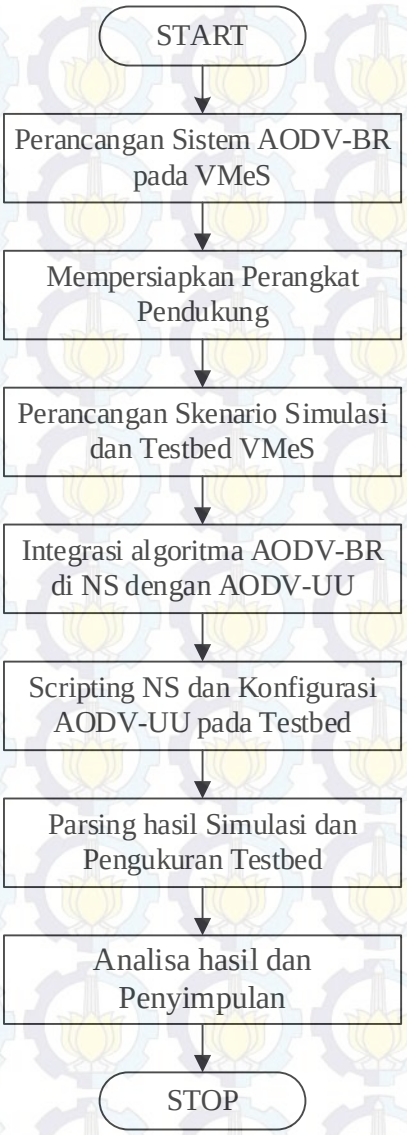
Analisa kinerja protokol routing AODV-BR pada penelitian ini, dilakukan pada protokol *medium access control (MAC)* 802.11 dikarenakan peralatan yang akan digunakan oleh VMeS saat ini masih dalam tahap pengembangan.

3.2 Rancangan Penelitian

Pada penelitian ini akan dibuat rancangan simulasi dan *testbed* menggunakan laptop untuk mengetahui kinerja dari protokol routing AODV *Backup Routing* (AODV-BR) pada sistem komunikasi VMeS secara riil. Pada tahap perancangan ini akan ditentukan skenario simulasi dan implementasi serta teknik pengambilan data serta analisa data. Untuk melakukan simulasi protokol *routing* AODV-BR pada sistem komunikasi VMeS ini digunakanlah *software Network Simulator 2* (NS 2.35) sebagai simulatornya. Untuk implementasi pada laptop digunakan AODV-UU sebagai *software* protokol *routing* yang berjalan pada Linux dengan kernel 2.6.x.

Penelitian ini diawali dengan mempersiapkan perangkat-perangkat berupa laptop yang sudah terinstall OS Linux dengan distro Ubuntu yang didalamnya sudah terpasang perangkat lunak *Network Simulator*. Untuk tahap-tahapan dalam perencanaan simulasi akan dibahas lebih rinci pada sub bab 3.4.1. Untuk uji coba kinerja AODV-BR pada perangkat, dibutuhkan beberapa laptop yang difungsikan sebagai *node*. Untuk langkah-langkah perancangannya akan dibahas pada sub bab 3.4.2. Setelah perangkat yang dibutuhkan untuk analisa kinerja protokol roting AODV-BR sudah siap, maka dilakukan pembuatan skenario simulasi dan implementasi sesuai dengan skenario VMeS dengan mensimulasikan kinerja protokol *routing* AODV-BR menggunakan NS dan meguji coba pada *testbed* laptop menggunakan AODV-UU yang bertujuan untuk memperoleh hasil-hasil pengukuran berdasarkan parameter kinerja yang ditentukan. Selanjutnya akan dilakukan pengolahan hasil simulasi dan hasil uji coba *routing* pada perangkat untuk kemudian dijadikan bahan untuk menganalisa hasil akhir dari penelitian terhadap kinerja protokol *routing* AODV-BR pada komunikasi VMeS.

Berikut ini adalah *flowchart* dari langkah-langkah yang akan dilakukan dalam penelitian buku tesis ini:



Gambar 3.2 *Flowchart* Tahapan Penelitian

Gambar 3.2 merupakan *flowchart* tahapan penelitian yang akan dilakukan untuk menganalisa kinerja protokol routing AODV-BR pada komunikasi VMeS, *flowchart* tersebut dibagi menjadi dua tahapan utama, yaitu tahapan yang dilakukan saat simulasi dan tahapan yang akan dilakukan saat implementasi dan uji coba pada perangkat *testbed*. Adapun tahapan-tahapan yang akan dilakukan pada penelitian ini adalah sebagai berikut:

1. Tahap awal adalah merencanakan sistem yang akan digunakan untuk menganalisa kinerja dari protokol routing AODV-BR, sistem yang akan digunakan meliputi simulasi menggunakan *tool network simulator* dan uji coba protokol untuk implementasi pada *testbed* laptop.
2. Mempersiapkan perangkat pendukung berupa perangkat lunak *(software)* dan perangkat keras *(hardware)* yang akan digunakan untuk menganalisa kinerja dari protokol routing AODV-BR. Untuk hal ini akan dirinci pada sub bab berikutnya.
3. Tahap Simulasi
   - Merancang skenario dan algoritma routing yang digunakan pada NS. Untuk skenario dan algoritma routing dari AODV-BR ini ditulis dalam bahasa pemprograman C++. Untuk penelitian ini, menggunakan modul *routing* yang sudah ada pada NS, setelah dilakukan sedikit modifikasi.
   - Proses *scripting* topologi VMeS di NS, topologi yang digunakan disamakan dengan kondisi riil pada lingkungan jaringan VMeS. Untuk membuat topologi ini dilakukan penulisan *script* agar dapat dapat dieksekusi oleh NS, yaitu menggunakan bahasa pemprograman TCL/OTCL.
   - Parsing Hasil simulasi merupakan tahap pemfilteran data yang tercatat pada *log* yang dihasilkan dari *file trace* akhir setiap simulasi di jalankan. Untuk melakukan ini, peneliti menggunakan *script AWK* untuk melakukan parsing terhadap parameter yang digunakan yaitu meliputi nilai *delay* dan *throughput* pada saat simulasi dijalankan.
4. Tahap Impelementasi
   - Perancangan skenario implementasi diperlukan untuk menentukan perangkat yang dibutuhkan, topologi, dan lokasi yang akan digunakan untuk melakukan uji coba kinerja pada protokol routing AODV-BR ini.
   - Integrasi *script* C++ AODV-BR di NS ke dalam AODV-UU dibutuhkan untuk menyamakan skenario yang sudah dibuat pada NS. AODV-UU merupakan *software* yang terbangun dari bahasa C yang dapat terintegrasi dengan NS.
   - Instalasi dan konfigurasi AODV-UU diinstal pada OS Ubuntu 10.04.1 yang membutuhkan banyak *software* tambahan, seperti *build-essential*, *linux-*

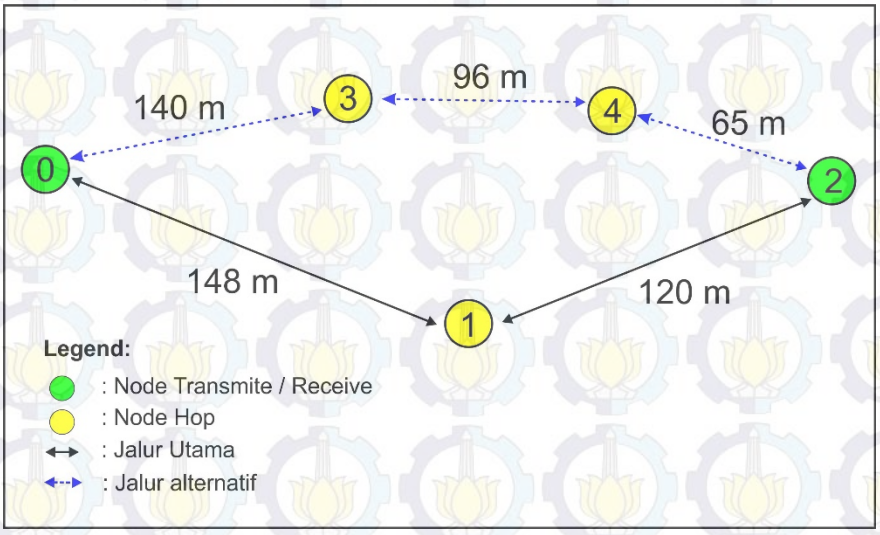
header yang ter-*update*, dan melakukan *compailing* pada AODV-UU agar dapat berjalan pada kernel OS 2.6.28.

- Uji coba dan pengambilan data implementasi dilakukan saat semua konfigurasi perangkat dan *software* yang dibutuhkan sudah siap digunakan. Untuk mengambil data dari uji coba pada *testbed*, digunakan *software Iperf* sebagai alat untuk melakukan testing pengiriman paket yang sudah ditentukan untuk uji coba.

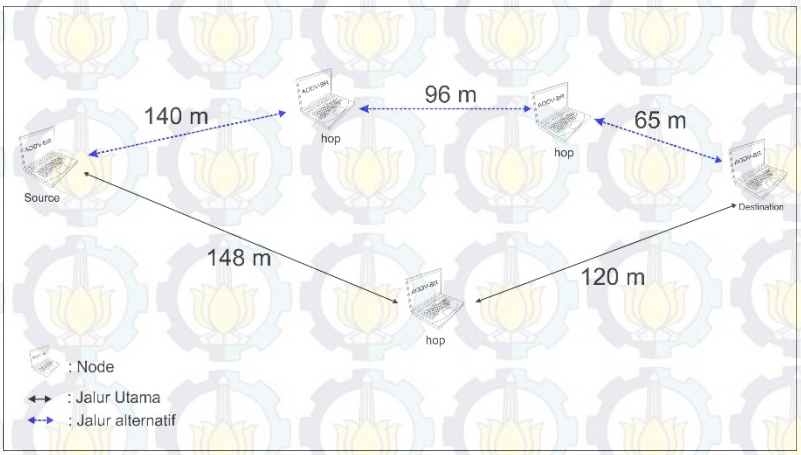
5. Analisa hasil dan penyimpulan dilakukan setelah proses dan tahapan-tahapan dari simulasi dan uji coba pada perangkat sudah dilakukan.

3.3 Desain Topologi Jaringan

Desain topologi yang digunakan pada penelitian ini meliputi dua aspek, yaitu desain topologi pada simulasi dan desain topologi pada *testbed*. Desain dari kedua aspek tersebut disamakan agar dapat dibandingkan kinerja dari masing-masing uji coba dari kinerja protokol routing AODV-*Backup Routing* terhadap komunikasi VMeS. Pada desin topologi *node* terbagi menjadi 3 bagian, yaitu *node source*, *hop*, dan *receiver*. Untuk menguji skema *backup routing* pada AODV dilakukan pada pengaturan *node* yang dijadikan *neighbor* atau *hop*, yaitu dari konfigurasi posisi dan konfigurasi arah pergerakan.

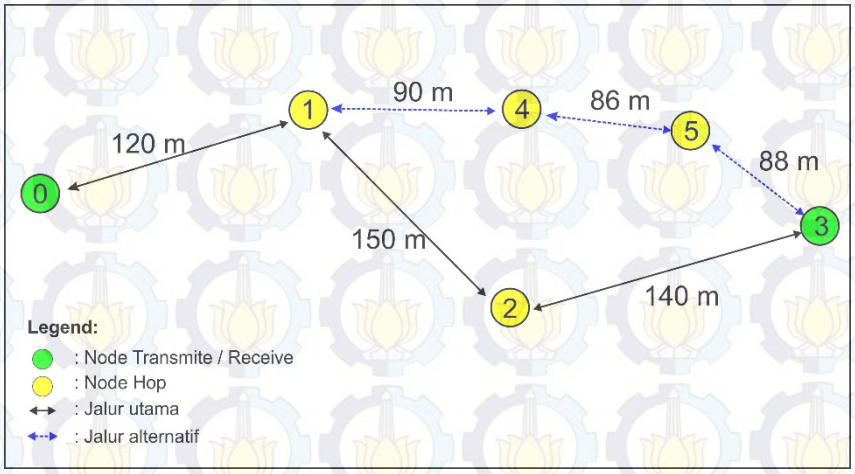


Gambar 3.3 Desain Topologi Simulasi untuk 3 *Hop*

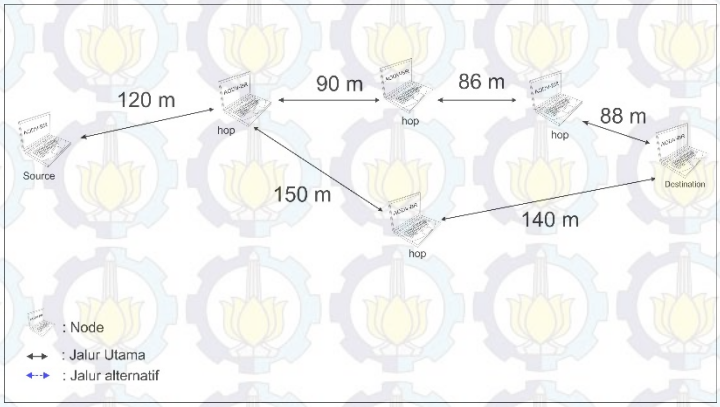


Gambar 3.4 Desain Topologi *Testbed* untuk 3 *Hop*

Pada Gambar 3.3 merupakan desain topologi simulasi untuk 3 *hop*, sedangkan pada Gambar 3.4 desain topologi *testbed* untuk 3 *hop* dimana pada setiap desain memiliki beberapa *node* yang memiliki peran masing-masing. Terdapat *node* yang memiliki peran sebagai pengirim dan *node* sebagai penerima serta beberapa *node* yang dijadikan sebagai loncatan (*hop*). Pada kedua Gambar 3.3 dan 3.4 didefinisikan konfigurasi jarak antar *node* satu dengan lainnya. Desain topologi untuk 3 *hop* dirancang untuk dijadikan pembanding terhadap pengaruh kinerja protokol *routing* dalam sebuah jaringan terhadap perubahan jumlah *node*, yang mana dalam penelitian ini dijadikan sebagai salah satu jenis parameter ukur terhadap kinerja protokol *routing* AODV *backup routing*. Topologi yang direncanakan disesuaikan dengan teknik pengujian yang akan dilakukan terhadap kinerja protokol *routing* AODV-BR untuk komunikasi VMeS.



Gambar 3.5 Desain Topologi Simulasi untuk 4 *Hop*



Gambar 3.6 Desain Topologi *Testbed* untuk 4 *Hop*

Pada Gambar 3.5 merupakan desain topologi simulasi untuk 4 *hop*, sedangkan pada Gambar 3.6 desain topologi *testbed* untuk 4 *hop* dimana pada setiap desain memiliki beberapa *node* yang memiliki peran masing-masing seperti pada desain untuk 3 *hop*. Terdapat *node* yang memiliki peran sebagai pengirim dan *node* sebagai penerima serta beberapa *node* yang dijadikan sebagai loncatan (*hop*). Kedua Gambar 3.5 dan 3.6 tersebut mendefinisikan konfigurasi jarak antar *node* satu dengan lainnya serta model penyusunan *node* jaringan. Desain topologi untuk 4 *hop* dirancang untuk dijadikan pembanding atas 3 *hop* terhadap pengaruh kinerja protokol *routing* dalam sebuah jaringan terhadap perubahan jumlah *node*, yang mana dalam penelitian ini dijadikan sebagai salah satu jenis parameter ukur terhadap kinerja protokol *routing* AODV *backup routing*.

Penelitian ini melakukan simulasi protokol routing dengan jenis topologi jaringan yang ditentukan menggunakan *Network Simulator 2.35* dan merancang testbed menggunakan laptop sebagai pembanding uji kinerja pada simulasi lingkungan ideal dan uji coba kinerja pada kondisi riil VMeS. Alasan utama dibuat dua skenario pengujian pada protokol routing AODV-BR untuk komunikasi VMeS agar sistem dan algoritma *routing* yang direncanakan dapat diadopsi penelitian selanjutnya dalam melakukan impelementasi langsung pada perangkat VMeS yang sebenarnya dengan cara mengacu pada teknik perancangan yang dilakukan pada *testbed*. Pengujian pada simulasi digunakan sebagai langkah awal untuk menguji kinerja protokol *routing* sebelum melakukan impelementasi. Melakukan perbandingan terhadap uji coba simulasi dan *testbed* dilakukan agar dapat

mengetahui kinerja protokol AODV-BR secara konseptual dan secara terapan pada lingkungan nyata.

3.4 Pembuatan Sistem

Pada sub bab ini akan membahas langkah-langlah untuk membuat sistem untuk menganalisa kinerja protokol *routing* AODV-*Backup Routing* terhadap komunikasi VMeS. Terdapat dua langkah yang akan dilakukan, yaitu tahap persiapan terhadap rancangan simulasi dan implementasi *testbed*.

3.4.1 Simulasi

3.4.1.1 Kebutuhan Dasar Simulasi

Dalam proses melakukan simulasi dan analisa kinerja protokol *routing* AODV-BR pada tesis ini dibutuhkan perangkat pendukung yang meluputi:

1. Perangkat keras *(hardware)* yang digunakan.
2. Perangkat lunak *(software)* yang dibutuhkan.

Adapun perangkat keras yang digunakan untuk mensimulasikan AODV-BR ialah menggunakan laptop dengan spesifikasi sebagai berikut:

- Processor : Intel Core i5-2430M 2.4 GHz
- Memory : DDR 3 SDRAM, 4 GB
- VGA : HD Graphics 3000
- HDD : 500 GB

Sedangkan perangkat lunak yang dibutuhkan untuk melakukan simulasi ialah sebagai berikut:

- *Operating System (OS)* yang digunakan adalah Ubuntu 12.04 LTS dengan *code-name Precise Pangolin*.
- *Network Simulator 2.35* sebagai perangkat lunak untuk melakukan simulasi jaringan.
- G-Edit yang akan digunakan untuk *script* editor pada bahasa C dan TCL.
- NAM digunakan untuk menampilkan animasi simulasi.

Selain persiapan secara perangkat, yang perlu dipersiapkan juga adalah skenario dalam perancangan tersebut, adapun hal-hal yang perlu dirancang sebelum melakukan simulasi adalah sebagai berikut:

- Merencanakan simulator objek dan *event-scheduler*
- Merencakanan topologi jaringan yang akan digunakan meliputi node pengirim dan penerima, node yang akan dijadikan loncatan *(hop)* serta rencana *link* antar node.
- Mendefinisikan pola trafik dengan membuat *agent*, *application* dan *flow*.
- Mengatur jalannya skenario simulasi seperti trafik *flow*, *trace* dan *event* lainnya.

3.4.1.2 Konfigurasi dan Instalasi Network Simulator 2 (NS 2.35)

Untuk dapat melakukan simulasi protokol routing AODV *Backup Routing*, yang dilakukan pertama kali adalah melakukan instalasi alat berupa *software* simulator dalam penelitian ini menggunakan *Network Simulator 2.35*. Sebelum melakukan instalasi dan konfigurasi terhadap NS, terdapat beberapa paket yang perlu diinstal agar *network simulator* dapat berjalan dengan sempurna. Adapun paket-paket yang perlu diinstal sebelum melakukan konfigurasi terhadap NS adalah sebagai berikut:

- *tcl8.5-dev, tk8.5-dev*
- *gcc-4.4, g++-4.4*
- *build-essential*
- *autoconf, automake*
- *perl*
- *xgraph*
- *libxt-dev, lib11-dev, dan*
- *libxmu-dev.*

Adapun persiapan serta langkah-langkah instalasi dan konfigurasi terhadap software Network Simulator 2.35 secara detail dipaparkan pada lampiran poin 1 dari buku tesis ini.

3.4.1.3 Konfigurasi Perangkat *Node* pada Simulasi

Node pada simulasi merupakan posisi titik lokasi perangkat komunikasi yang digunakan, baik yang diatur sebagai node *fix* maupun *node mobile* yang dalam penelitian ini merupakan *node* yang dibawa oleh kapal nelayan. Konfigurasi *node* yang akan digunakan pada saat simulasi sudah diatur sesuai dengan skenario pengujian dan kebutuhan sistem. Pengaturan yang disusun dalam simulasi disesuaikan dengan parameter perangkat *testbed* agar dapat dibandingkan hasil kinerja terhadap beberapa parameter ukur. Adapun konfigurasi terhadap parameter *node* pada simulasi adalah pada Tabel 3.1 dibawah ini.

Tabel 3.1 Konfigurasi *Node* pada Simulasi

NO	PARAMETER	VALUE
1	*Channel*	*Wireless Channel*
2	Jenis Propagasi	*TwoRayGround*
3	Antena	*Omni Antenna Directional*
4	Ketinggian Antena	1,5 m
5	Lapisan Fisik	*NetIf*
6	Protokol MAC layer *Datalink*	802.11
7	*Routing* Protokol	AODV-BR / AODV

Pada Tabel 3.1 merupakan konfigurasi perancangan yang dilakukan pada setiap node simulasi. *Channel* merupakan jenis kanal yang digunakan, dalam hal ini diseting menggunakan *wireless* karena jaringannya lebih dinamis. Jenis propagasi yang digunakan adalah *TwoRayGround* dimana setiap *node* dapat saling melihat atau *line of sight (los)* terhadap *node* lainnya atau memantulkan propagasi jika diatas permukaan bumi. Antena yang digunakan adalah *antenna omni directional* dimana pola radiasi yang dipancarkan lebih lebar secara vertikal apabila posisi antena ditegakkan. Ketinggian antena disamakan dengan *testbed* yaitu 1,5 m di atas permukaan bumi. Protokol yang digunakan pada lapisan *datalink* adalah *Medium Access Control* 802.11 sebagai standar yang digunakan oleh Wifi. Protokol *routing* yang digunakan adalah AODV-BR (modifikasi dari AODV) dan AODV tradisional sebagai protokol pembanding kinerja.

3.4.1.4 Konfigurasi Topologi Jaringan pada Simulasi

Pembuatan dan perancangan topologi simulasi mengacu pada Gambar 3.3 dan Gambar 3.5 sebagai patokan perancangan topologi yang direncanakan. Topologi jaringan di *wireless ad hoc network* untuk simulasi yang digunkana pada komunikasi VMeS ini ada dua macam, yaitu *static* dan *dynamic ad hoc network*. Untuk istilah *static ad hoc network* digunakan pada *node* yang tidak bergerak, dalam sistem VMeS digunakan sebagai *gateway* di wilayah darat. Misalnya node_1 (G1) pada awal sampai dengan akhir simulasi berada pada posisi X=10 dan Y=10. Untuk konfigurasi topologi secara keseluruhan terdapat pada Tabel 3.2 berikut.

Tabel 3.2 Konfigurasi *Node* pada Simulasi

NO	PARAMETER	VALUE
1	Jangkauan transmisi max.	150 m
2	Luas topologi	600 x 600 m
3	Jumlah node yang terlibat	5 *node* (3 *hop*) dan 6 *node* (4 *hop*)
4	Kecepatan pergerakan *node*	0.5 m / detik (rata-rata 20 Km/jam)
5	Arah pergerakan	Keluar dari *coverage area*
6	Jumlah *node* yang bergerak	1 *node* bergerak, sisanya statis

Pada Tabel 3.2 dapat dijelaskan parameter yang akan dibuat acuan pengaturan terhadap topologi simulasi. Jangkauan transmisi setiap *node* diatur 150 m, Luas topologi yang digunakan adalah 600 x 600 m, Jumlah node yang terlibat bervariasi tergantung skenario pengujian yang akan dilakukan. Kecepatan pergerakan *node mobile* merupakan kecepatan maksimum mobile node dalam hitungan detik, diseting 0.5 m/detik karena kecepatan kapal laut nelayan dengan ukuran 5 GT kurang lebih 20 km/jam [2]. Arah pergerakan yaitu keluar dari *caverage area* jaringan pada konfigurasi awal posisi *node* yang sudah dibahas pada sub bab 3.3 (Desain Topologi Jaringan). Pada uji coba ini, *node* yang bergerak dengan kecepatan 0.5 m/detik hanyalah satu *node* yang dianggap sebagai kapal nelayan, untuk node lainnya distatiskan. Untuk konfigurasi posisi *node* pada simulasi 3 *hop* dapat dilihat pada Tabel 3.3 dibawah ini.

Tabel 3.3 Konfigurasi *Node* Awal pada Simulasi 3 *hop*

NO	ID Node	Posisi X	Posisi Y	Status
1	Node_(0)	10	10	Statis
2	Node_(1)	158	150	Pindah ke X,Y (158,10)
3	Node_(2)	278	350	Statis
4	Node_(3)	150	290	Statis
5	Node_(4)	246	310	Statis

Pada Tabel 3.3 merupakan konfigurasi posisi *node* yang ditulis dalam *script* TCL yang disusun untuk topologi VMeS. Posisi pada masing-masing *node* ditentukan secara statis dan hanya ada satu *node* yang bergerak. Skenario ini

disusun untuk dapat disesuaikan dengan teknik pengujian AODV *Backup Routing* yaitu dalam teknik memanfaatkan jalur alternatif yang dibuatnya. *Node* yang bergerak yaitu node_(1) yang merupakan *node neighbor*, bergerak dari posisi semula yaitu X (158) dan Y (150) dengan kecepatan 0.5 m/detik menuju posisi X (158) dan Y (10) pada topograpi NS.

Tabel 3.4 Konfigurasi *Node* Awal pada Simulasi 4 *Hop*

NO	ID Node	Posisi X	Posisi Y	Status
1	Node_(0)	10	20	Statis
2	Node_(1)	130	150	Statis
3	Node_(2)	280	320	Pindah ke X,Y (280,30)
4	Node_(3)	420	420	Statis
5	Node_(4)	230	310	Statis
6	Node_(5)	306	330	Statis

Tabel 3.4 merupakan konfigurasi pengaturan terhadap topologi simulasi dengan jarak 4 *hop*. Pada pengaturan tersebut, semua node statis dan hanya node_(2) sebagai *neighbor* / hop dari node_(1) ke node_(3) yang melakukan perpindahan ke arah luar caverage jaringan pada topologi 4 hop. Node_(2) berpindah dari posisi awal X,Y (280, 230) menuju X,Y (280,30) pada topograpi simulasi di Network Simulator 2.35.

3.4.1.5 Konfigurasi Jaringan pada Simulasi

Konfigurasi pada jaringan simulasi digunakan untuk memetakan sistem jaringan yang akan digunakan. Konfigurasi ini meliputi beberapa hal yang memiliki peran dalam seting konfigurasi jaringan, seperti jumlah paket yang akan ditransmisikan, *transport agent* yang digunakan dan lain sebagainya.

Transport agent yang digunakan pada simulasi ini adalah TCP pada sisi pengirim dan TCPSink pada sisi penerima. Aplikasi untuk pengiriman paket adalah *Constant Bit Rate (CBR).* Kapasitas data yang ditransmisikan yaitu sebesar 32, 64, 128 KB sebagai sampel pengiriman paket. Waktu transmisi awal adalah waktu dimana simulai mulai mengirim paket data, konfigurasi yang ditetapkan *node* sender mulai mengirim paket mulai dari milidetik ke 10 sejak dimulainya simulasi dan diakhiri pada milidetik ke 19990 karena total waktu simulasi yang digunakan

20 detik. Adapun konfigurasi yang digunakan pada simulasi sistem protokol *routing* AODV-BR pada komunikasi VMeS ini pada Tabel 3.4 dibawah.

Tabel 3.5 Pengaturan Dasar Jaringan pada Simulasi

NO	PARAMETER	VALUE
1	Transport Agent	TCP/TCPSink
2	Aplikasi Trafik	CBR
3	Kapasitas data	32, 64, 128 KB
4	Waktu awal transmit	10 ms
5	Waktu Akhir Transmit	19990 ms
6	Waktu Simulasi	20 detik

3.4.2 *Testbed*

3.4.2.1 Spesifikasi Peralatan

Dalam proses penelitian untuk menganalisa kinerja dari protokol *routing* AODV-BR secara simulasi pada sistem komunikasi VMeS ini, diperlukan beberapa perangkat pendukung, yaitu berupa perangkat keras dan perangkat lunak. Perangkat keras yang digunakan untuk melakukan analisa dan implementasi protokol *routing* AODV-BR ini membutuhkan beberapa *node*. Setiap *node testbed* dianggap memiliki performasi perangkat nirkabel yang sama, yaitu menggunakan USB Dongle TP-Link TL-WN722N IEEE 802.11 n seperti pada Gambar 3.7.

Gambar 3.7 USB Dongle TP-Link TL-WN722N

Adapun spesifikasi dari USB Dongle TP-Link TL-WN722N tersebut adalah sebagai berikut:

Tabel 3.6 Spesifikasi *Wireless Adapter* yang Digunakan

NO	FITUR	VALUE
1	Port	USB 2.0
2	Dimensi (W x D x H)	3.7 x 1.0 x 0.4 in. (93.5 x 26 x 11mm)
3	Tipe Antena	Detachable Omni Directional (RP-SMA)
4	Gain Antena	4dBi
5	Frekuensi	2.400-2.4835GHz
6	Standar Wireless	IEEE 802.11n, IEEE 802.11g, IEEE 802.11b
7	Jangkauan Area Los	Up to 200 m

Sedangkan *node-node testbed* yang digunakan berupa laptop yang memiliki spesifikasi sebagai berikut:

Tabel 3.7 Spesifikasi *Hardware* yang Digunakan

NODE	PROCESSOR	RAM	OS
1	Intel Core i5-2430M CPU @ 2.40GHz	4 GB	Linux Kernel 2.6.32 (Ubuntu 10.04.1)
2	Intel Atom CPU Z520 @ 1.33GHz	2 GB	Linux Kernel 2.6.32 (Ubuntu 10.04.1)
3	Intel Core 2 Duo CPU E7500 @ 2.93GHz	2 GB	Linux Kernel 2.6.32 (Ubuntu 10.04.1)
4	Intel ® core 2 duo T6600	2 GB	Linux Kernel 2.6.32 (Ubuntu 10.04.1)
5	Intel Core Dual Core, 2.1GHz	2 GB	Linux Kernel 2.6.32 (Ubuntu 10.04.1)
6	Intel atom CPU 2.1 GHz	1 GB	Linux Kernel 2.6.32 (Ubuntu 10.04.1)

Setelah kebutuhan *hardware* sudah terpenuhi, dibutuhkan perangkat lunak untuk menjalankan sistem yang diinginkan, yaitu untuk menganalisa kinerja protokol *routing* AODV-BR pada VMeS. Terdapat beberapa *software* yang akan digunakan, yaitu *source* AODV yang digunakan adalah *aodv-uu-0.9.6.orig.tar.gz*. Selain beberapa kebutuhan *software* utama, terdapat beberapa *software* tambahan yang dibutuhkan yaitu seperti *cpp-4.4, dpkg-dev, fakeroot, g++, libgcc, iptable*

1.36. untuk *tool* pengukuran performasinya jaringan menggunakan *iperf* versi 2.0.2 yang bekerja dibawah terminal pada linux.

3.4.2.2 Instalasi dan Konfigurasi AODV-UU

Untuk melakukan instalasi dan konfigurasi AODV-UU pada operating system Linux dengan kernel 2.6.32 dibutuhkan *source AODV-UU* dengan versi 0.9.6 dan melakukan *updating* terhadap *repository* yang digunakan sebelum melakukan instalasi. Melakukan instalasi beberapa paket *software* yang dibutuhkan sebelum melakukan *compailing* terhadap *source* AODV seperti *gcc, lib gcc*, dll. Melakukan *update* terhadap *header kernel* yang dibutuhkan. Untuk melakukan beberapa hal yang disebutkan tadi, berikut langkah-langkah yang digunakan untuk melakukan konfigurasi dan instalasi AODV-UU.

- Gunakan perintah *apt-get install build-essential* pada terminal untuk melakukan instalasi terhadap beberapa paket *software* yang dibutuhkan sebelum instalasi dan konfigurasi AODV-UU seperti *gcc, lib gcc*, dll.
- Lakukan instalasi linux header seperti yang digunakan pada penelitian ini yaitu *linux-headers-2.6.32-24-generic* dengan mengeksekusi perintah ini melalui terminal *apt-get install linux-headers-$(uname –r).*
- Lakukan ektraksi pada *aodv-uu-0.9.6.orig.tar.gz.* kemudian arahkan posisi direktori pada terminal kedalam *folder* aodv-uu yang sudah diekstrak.
- Gunakan perintah *make* pada terminal untuk melakukan *compailing* pada paket AODV-UU yang sudah diekstrak tadi.
- Jika tidak terjadi *error*, maka proses *compailing* berhasil dilakukan. Kemudian gunakan perintah *make install* untuk melakukan instalasi paket AODV-UU yang sudah berhasil di-*compail*.

3.4.2.3 Instalasi dan Konfigurasi Alat Ukur

Iperf merupakan *tool* performasi pada sebuah jaringan, difungsikan untuk menguji coba pengiriman paket dengan ketentuan kapasitas tertentu. *Iperf* pada penelitian ini ditugaskan untuk menangani kebutuhan pada lapisan aplikasi yaitu digunakan sebagai sarana untuk melakukan perintah mengirim dan menerima paket antar *node*, dan menghitung performa dari jaringan yang digunakan. Untuk

melakukan instalasi dan konfigurasi terhadap *iperf* dilakukan dengan beberapa cara berikut:

- Lakukan instalasi via terminal dengan mengetikkan perintah *apt-get install iperf* dan biarkan OS melakukan instalasi secara *online*.
- Untuk melakukan konfigurasi *iperf* dalam penelitian ini, dilakukan seting yang berbeda antara sisi node pengirim paket dan sisi *node* penerima. Pada sisi penerima, *iperf* distatuskan sebagai penerima dengan mengetikkan perintah:

```
iperf -s
```

- Sedangkan untuk sisi pengirim, perintah yang dieksekusi memiliki format *iperf -c [ip tujuan] –n [kapasitas data yang hendak dikirim misalnya 128K] –I 0.1*, contoh perintah yang digunakan pada terminal:

```
Iperf –c 10.10.10.4 –n 128K –i 0.1
```

3.4.2.4 Konfigurasi Jaringan pada *Testbed*

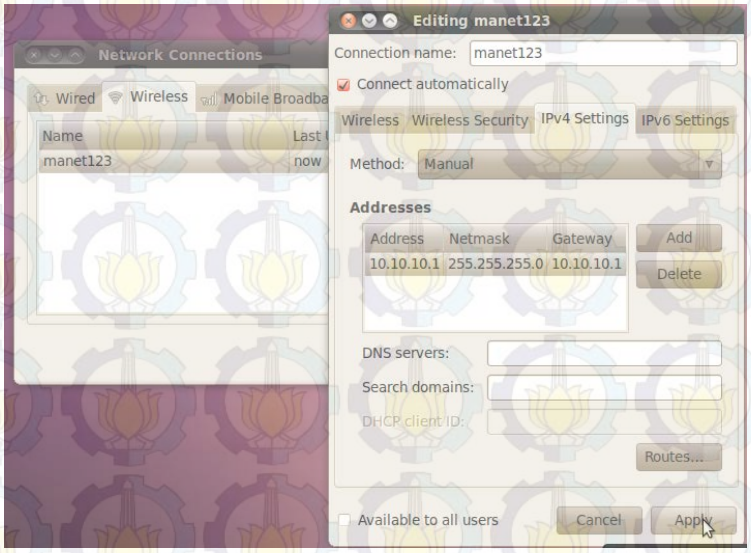
Setting network dalam sistem ini dibutuhkan untuk mengidentitaskan masing-masing *node* agar dapat dikenali oleh *node* yang memiliki pengaturan yang sama. *Setting network* yang dimaksud meliputi; seting IP, seting SSID jaringan yang digunakan, dan melakukan pancaran jaringan *ad hoc* pada masing-masing *node*.

Untuk melakukan pengaturan-pengaturan tersebut, dibutuhkan beberapa langkah sebagai berikut:

- Langkah awal adalah memastikan bahwa OS sudah mendeteksi *hardware* jaringan digunakan, untuk penelitian ini menggunakan *hardware wireless*.
- Membuat SSID dalam penelitian ini SSID yang digunakan adalah “manet123” dan memancarkan sinyal *ad hoc* dari masing-masing *node* dengan cara klik kanan pada *toolbar* jaringan di taskbar OS Ubuntu, kemudian klik *create new wireless*.

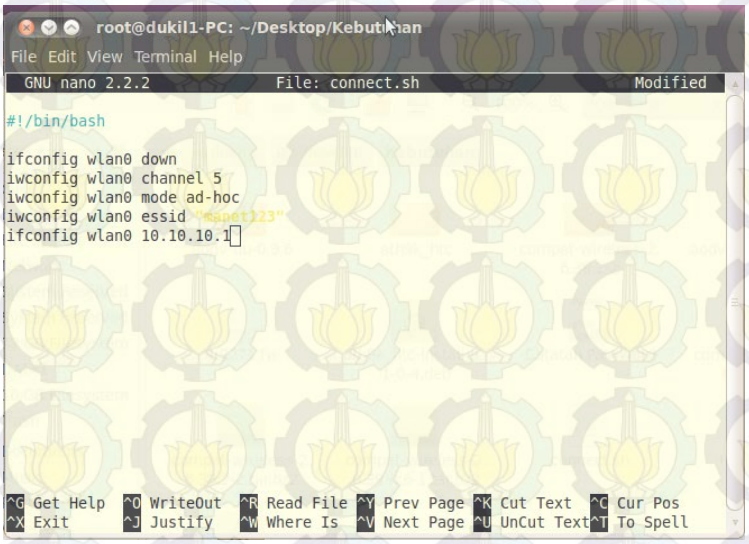
Gambar 3.8 Pembuatan SSID *Ad Hoc*

- Lakukan seting IP dan *subnetmask* yang sekelas, agar saat sebuah node tergabung dalam jaringan yang sama, langsung terkonfigurasi dan terkoneksi secara otomatis.



Gambar 3.9 *Seting IP* pada Jaringan *Ad Hoc* yang Digunakan

Untuk beberapa versi OS lain dari linux seperti Debian (etch), untuk melakukan konfigurasi ini dapat dilakukan dengan sekali cara, yaitu dengan membuat *file.sh* yang nantinya akan dieksekusi satu kali dan otomatis semua konfigurasi seperti pembuatan SSID jaringan *ad hoc* dan seting IP akan berubah dengan satu kali langkah, berikut tampilan *file* yang berisi *script* dan perintah yang digunakan untuk langkah ini:



Gambar 3.10 *Script Config Wireless Ad Hoc* Menggunakan *File.sh*

Adapun seting dan konfigurasi yang dilakukan pada setiap *node* yang digunakan untuk analisa kinerja protokol routing AODV *backup routing* pada *testbed* adalah pada Tabel 3.8 berikut:

Tabel 3.8 Konfigurasi *Node* pada *Testbed*

NODE	IP	SSID	MODE
1	10.10.10.1	manet123	AdHoc
2	10.10.10.2	manet123	AdHoc
3	10.10.10.3	manet123	AdHoc
4	10.10.10.4	manet123	AdHoc
5	10.10.10.5	manet123	AdHoc
6	10.10.10.6	manet123	AdHoc

3.5 Skenario Pengujian Sistem

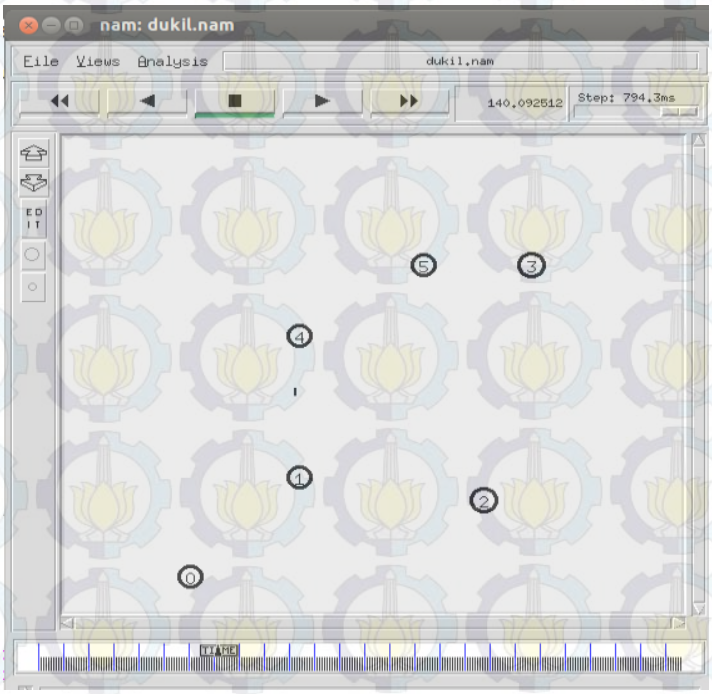
Skenario pengujian merupakan langkah-langkah yang akan dilakukan untuk menguji kinerja dari protokol routing yang ditentukan. Skenario pengujian ini meliputi pengujian pada saat simulasi dan pengujian pada *testbed*. Adapun skenario dibagi menjadi sebagai berikut:

3.5.1 Simulasi

3.5.1.1 Pengujian Simulasi pada NS

Pada proses pengujian simulasi pada NS, digunakan bahasa pemprograman TCL untuk pengaturan topologi jaringan, data yang dikirim, dan beberapa ketentuan lain yang tentunya disesuaikan dengan sistem yang diinginkan untuk

bahan pengujian kinerja dari algoritma protokol AODV-BR pada komuinkasi VMeS dapat berjalan dengan baik, sehingga dari hasil simulasi diharapkan dapat menarik kesimpulan tentang kinerja dari protokol *routing* AODV-BR pada komunikasi VMeS.



Gambar 3.11 Pengujian simulasi yang ditampilkan dalam file NAM

3.5.1.2 Pengukuran Parameter QoS

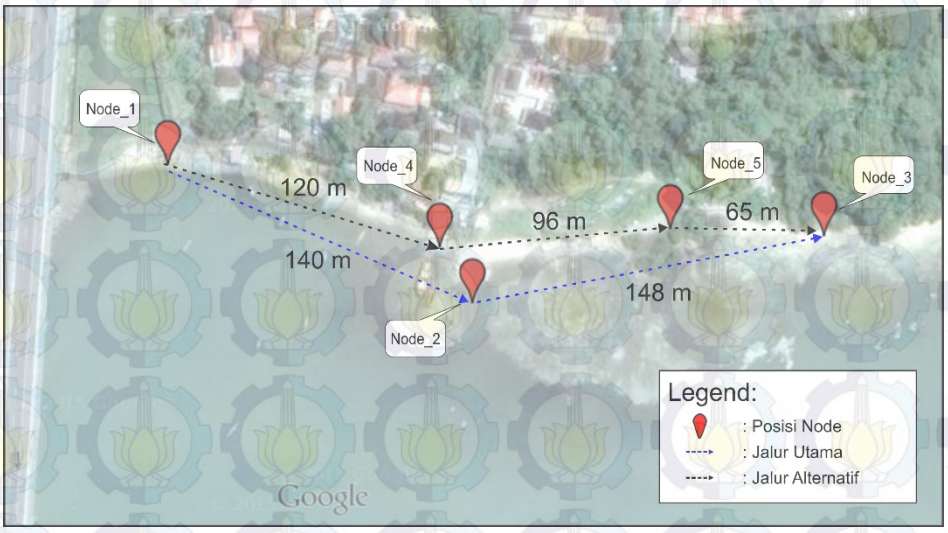
Pada pengujian *Quality of Service (QoS)* pada jaringan *ad hoc* yang dibuat untuk komunikasi VMeS, perlu ditentukan parameter yang akan diukur untuk menganalisa sebuh kinerja dari protokol routing AODV-BR. Parameter-parameter yang dibutuhkan untuk analisa yaitu *delay* dan *throughput*. Untuk mendapatkan nilai dari hasil kinerja pada saat simulasi, digunakan *script awk* untuk menparsing beberapa data yang terdapat pada *file trace* di akhir simulasi.

3.5.2 *Testbed*

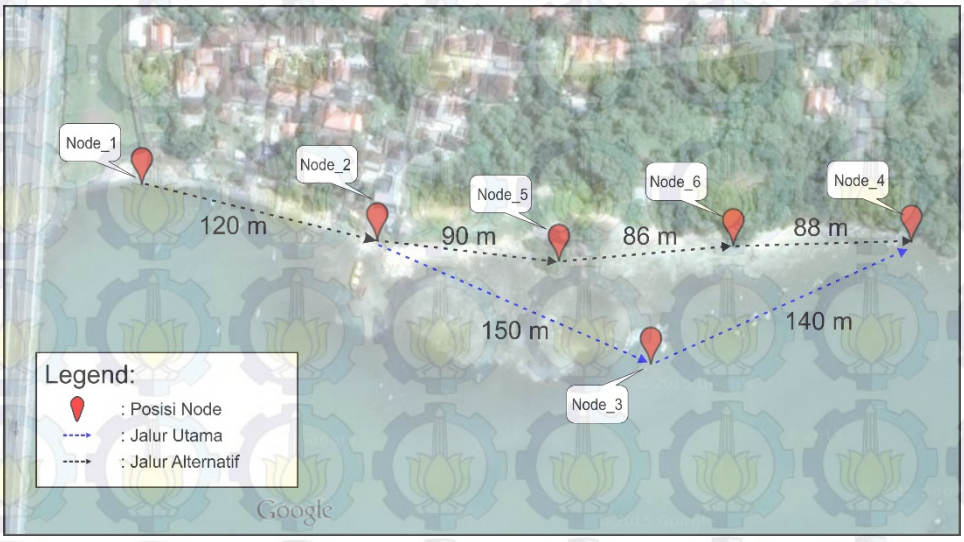
3.5.2.1 Lokasi Pengujian

Lokasi pengujian terhadap testbed yang sudah berhasil dijadikan *router* dalam *ad hoc* VMeS ini diuji coba di salah satu pantai Madura berdekatan dengan lokasi jembatan Suramadu. Lokasi diasumsikan *line of sight* tanpa adanya penghalang (*obstacle* yang tentunya memiliki pengaruh besar) karena antar *node*

diseting saling terhubung langsung dengan jarak bervariasi seperti yang sudah diskenariokan. Pengujian diharuskan *los* karena sistem komunikasi VMeS diterapkan pada wilayah laut yang sudah dipastikan antar node tidak terdapat penghalang apapun. Adapun denah lokasi pengukuran dan posisi node ditunjukkan pada Gambar 3.12 dan Gambar 3.13 berikut:



Gambar 3.12 Lokasi Pengukuran dan Konfigurasi Posisi 3 *Hop*



Gambar 3.13 Lokasi Pengukuran dan Konfigurasi Posisi 4 *Hop*

3.5.2.2 Pengujian Koneksi antar *Node*

Sebelum melakukan uji kemampuan *routing* pada *node* yang sudah ditanamkan didalamnya routing AODV-BR, terlebih dahulu perlu diuji konektivitas tanpa menjalankan *routing*. Untuk melakukan uji konektivitas pada setiap *node*, *node* dipastikan secara bersama memancarkan sinyal *ad hoc* dengan SSID yang sama dan IP *address* yang berbeda. Setelah saling terkoneksi dengan jaringan yang dipancarkan satu *node* dengan lainnya, maka lakukan PING di terminal terhadap IP node yang sedang terkoneksi. Perintah yang dieksekusi melalui terminal misalnya hendak melakukan PING pada IP 10.10.10.6 adalah seperti

```
Ping 10.10.10.6
```

Dalam penelitian ini, PING juga difungsikan sebagai cara untuk konfigurasi posisi masing-masing *node* dan pengambilan data. Dengan cara menjalankan perintah ping terhadap *node* tetangga kemudian menggeser *node* hingga ujung *caverage area*. Untuk mengetahui ujung *caverage area* masing-masing dari jangkauan transmisi node bisa diketahui dari proses ping yang awalnya berjalan menghitung *seq, ttl*, dan *time* berubah status menjadi *destination host unreachable*.

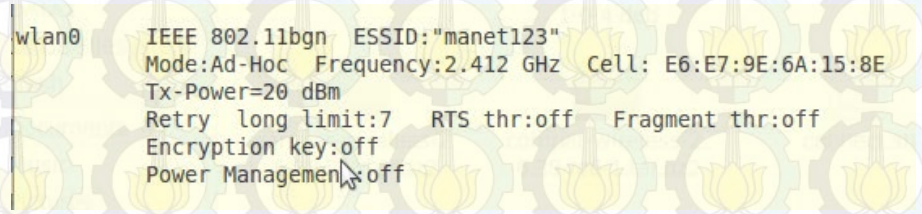
3.5.2.3 Pengujian jaringan yang digunakan

Untuk melakukan seting dan konfigurasi terhadap jaringan yang digunakan, perlu dikoreksi terlebih dahulu pengaturan jaringan yang hendak digunakan. Uji coba ini, hanya dibutuhkan untuk memastikan agar tidak terjadi kesalahan seperti perbedaan SSID jaringan, dan memastikan untuk tidak terjadinya IP ganda pada *node* yang terhubung.

Untuk melakukan *cheking* terhadap status seting dan konfigurasi jaringan yang sudah dilakukan, ketikkan perintah berikut pada terminal:

```
iwconfig
```

Maka anda dapat mengkoreksi status dari seting dan konfigurasi jaringan yang hendak digunakan, seperti pada Gambar 3.14 berikut:



Gambar 3.14 Seting Konfigurasi *Wireless* yang Digunakan

3.5.2.4 Pengujian protokol *routing* AODV-BR

Pengujian kinerja protokol *routing* AODV-BR dilakukan untuk mengetahui kinerja dari protokol ini. Pengujian dilakukan dengan cara mengecek jalur dari *node sender* ke *node destination* setelah protokol *routing* dijalankan. Untuk menjalankan routing AODV-BR ini dapat dilakukan melalui terminal dengan perintah.

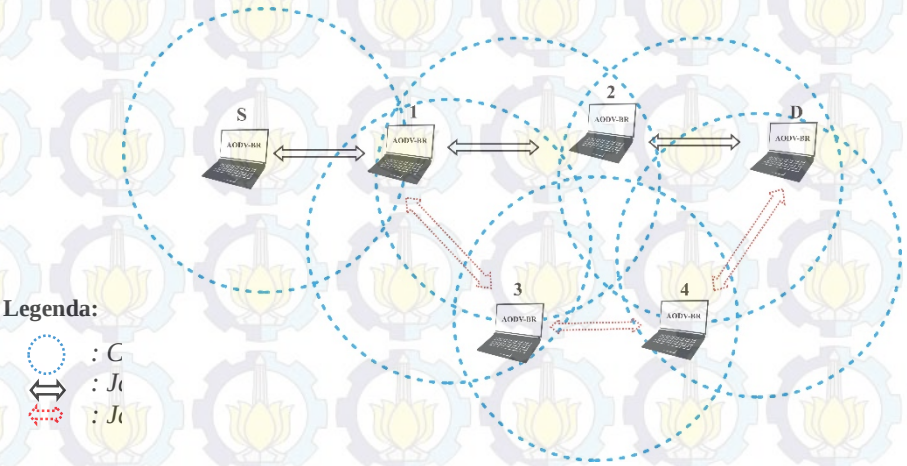
```
aodvd -D
```

Setelah memastikan setiap *node* yang tergabung dalam jaringan *ad hoc* aplikasi protokol routingnya jalan secara keseluruhan. Langkah selanjutnya untuk melakukan uji coba apakah jalur sudah terbentuk atau tidak, maka lakukan dengan cara *thresould* terhadap IP tujuan atau lakukan dengan cara PING dengan ekstensi –R terhadap IP tujuan, maka akan tampak beberapa jalur berupa identitas IP perangkat yang akan dilalui untuk menuju ke *node* destinasi.

3.5.2.5 Skenario Pengujian pada *Testbed*

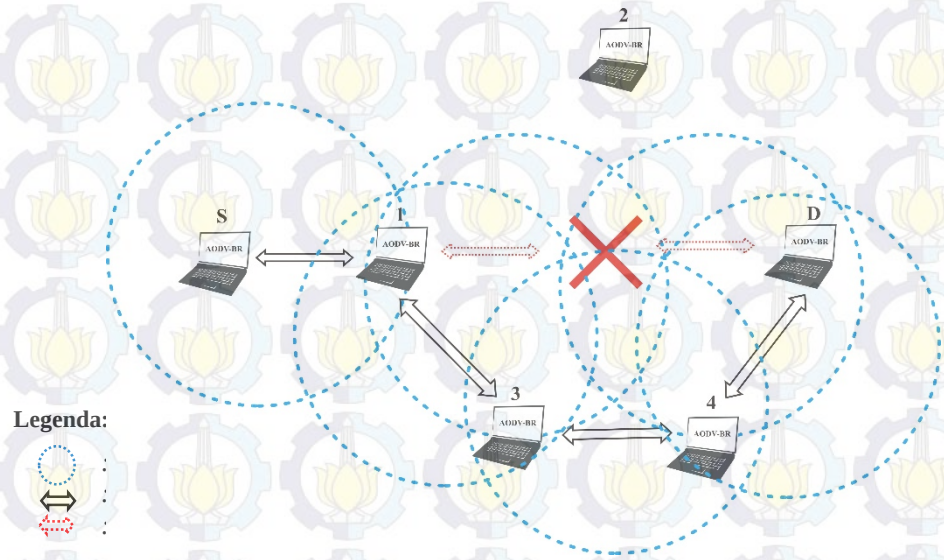
Skenario pengujian meliputi topologi yang digunakan saat uji coba protokol *routing* AODV-BR pada *node* yang berupa laptop. Posisi antar *node* dalam penelitian ini diasumsikan berada pada ujung *caverage* masing-masing *node* yang lain, artinya tidak diperbolehkan dalam satu *caverage* memiliki dua jangkaun terhadap *node* yang dijadikan *neighbor* kecuali *node* yang menjadi *hop* sebelumnya dan *hop* setelahnya. Misalkan dalam sebuah jaringan terdapat tiga *node* yaitu node_1, node_2, dan node_3. Node_1 hanya dapat menjangkau node_2 dan tidak boleh menjangkau node_3. Sedangkan node_2 yang menjadi *hop* dari *node* node_1 diharuskan menjangkau node_1 dan node_3. Node_3 hanya diperbolehkan menjangkau node 2. Alasan diskenariokan seperti ini, dikarenakan konsep protokol *routing* yang digunakan adalah protokol *routing* AODV dimana *routing algoritm* yang diadopsi oleh AODV adalah *routing* yang berdasarkan jarak *hop count* dan *sequence number*. Pada prakteknya, AODV tidak dapat menjadikan node *hop*

kendatipun posisi *node* sangat dekat jika node *destination* yang diinginkan masih dalam *coverage area node sender*. *Node sender* tidak akan memilih *node hop* melainkan langsung men-*direct* ke *destination*. Jadi, sebuah *node* diskenariokan tidak diperbolehkan untuk mencakup lebih dari satu node dalam *coverage area* yang dimilikinya, kecuali *node hop* yang mencakup dua *node* pada posisi kanan dan kirinya. Adapun topologi yang digunakan pada skenario pengujian kinerja protokol *routing* AODV-BR pada implementasi keperangkat ini adalah diitunjukkan pada Gambar 3.15 berikut:



Gambar 3.15 Topologi Pengujian Sistem *Routing* dengan 4 *Hop*

Pada Gambar 3.15 merupakan skenario pengujian yang dilakukan untuk mengetahui kinerja dari protokol routing AODV-BR, dimana masing-masing *node* hanya dapat menjangkau satu *node* yang akan dijadikan *next hop* atau *neighbor*. *Node* diletakkan sesuai skenario untuk menguji *backup routing* dari AODV-BR. Skenario pengujian yang akan dilakukan adalah pada saat *node* melakukan pengiriman data dari *node* ujung ke *node* penerima yang mana pada Gambar 3.15 *node* sumber ditandai dengan (S) dan node penerima ditandai dengan (D) salah satu *node* yang dijadikan *hop* dari pengirim ke penerima akan dipindahkan untuk keluar dari *coverage area* atau dengan cara mematikan *node* untuk menguji jalur alternatif yang dimiliki oleh AODV-BR.



Gambar 3.16 Kondisi Jalur pada AODV-BR Saat *Node* Dikeluarkan dari *Caverage Area*

Pada Gambar 3.16 merupakan skenario pengujian mengeluarkan satu *hop* yang dijadikan jalur utama oleh AODV-BR *caverage area* node yang lain agar AODV-BR dapat melakukan *switch* untuk merubah jalur yang akan dilalui paket ke jalur alternatif tanpa melakukan *route discovery* kembali.

3.6 Proses Penyimpulan Hasil Penelitian

Dari seluruh proses yang dilakukan pada penelitian ini yang meliputi tahap perancangan, implementasi dan pengujian, akan didapatkan beberapa kesimpulan yang dapat diambil. Kesimpulan diambil berdasarkan analisa data dengan beberapa parameter kinerja dari protokol *routing* AODV-BR secara keseluruhan.

Analisa kinerja yang akan dilakukan adalah berdasarkan nilai yang dihasilkan dari pengukuran performasi jaringan meliputi parameter *delay* menggunakan persamaan 2.1 dan *throughput* pada jaringan menggunakan persamaan 2.2 dengan protokol *routing* yang sudah ditentukan, yaitu AODV *Backup Routing* dan AODV tradisional sebagai protokol pembanding.

BAB 4
HASIL DAN ANALISA DATA

Pada bab ini akan dibahas mengenai hasil dari simulasi AOV-BR pada NS dan *testbed* menggunakan laptop. Pada akhir simulasi, diperoleh *file trace* yang mencatat seluruh aktivitas kejadian selama simulasi berlangsung. *File trace* ini, akan dijadikan bahan dasar untuk menfilter data dengan melakukan parsing menggunakan *script awk*. Pada simulasi jumlah *node* yang diuji coba sama halnya dengan yang digunakan saat implementasi pada alat. Sedangkan untuk *testbed*, teknik pengambilan data menggunakan *tool iperf* yang dapat menguji performasi jaringan dengan protokol *routing* yang sudah ditentukan, adapun jumlah *node* yang digunakan untuk *testbed* adalah 6 *node* dikarenakan keterbatasan alat saat melakukan uji coba *testbed*.

Protokol *Medium Access Control (MAC)* pada lapisan *datalink* yang digunakan VMeS, semestinya adalah protokol AX.25 sebagai protokol radio disebabkan prangkat pada lapisan fisik yang diinginkan harus mendukung frekuensi yang digunakan oleh komunikasi VMeS, yaitu HF dan VHF. Namun pada penelitian ini, masih menggunakan lapisan *Datalink* dan Fisik 802.11 baik pada sisi simulasi maupun implementasi testbed, dikarekan *tool* simulasi yang digunakan yaitu *Network Simulator 2.35* belum mendukung protokol *Medium Access Control (MAC)* AX.25 yang digunakan sebagai protokol MAC pada VMeS. Untuk uji coba kinerja protokol *routing* terhadap perangkat juga memanfaatkan *Medium Access Control (MAC)* 802.11 disebabkan perangkat yang mendukung protokol AX.25 masih dalam tahap pengembangan.

Sebagai pembanding terhadap analisa kinerja dari AODV-BR peneliti juga melakukan simulasi dan uji coba menggunakan AODV tradisional yang masih belum memiliki *multipath routing*. Parameter yang digunakan untuk analisa terhadap kinerja AODV-BR adalah menguji performa terhadap *QoS* jaringannya, yaitu *delay* yang dibutuhkan dan *throughput* yang dihasilkan, adapun parameter pengujian ditunjukkan pada Tabel 4.1 dibawah.

Tabel 4.1 Parameter Uji Coba untuk Analisa

PARAMETER UJI	SIMULASI	TESTBED
Jarak 3 Hop	√	√
Jarak 4 Hop	√	√
Throughput	√	√
Delay	√	√
Variasi Data	√	√

4.1 Hasil Pengujian Simulasi

Analisa pada hasil simulasi dilakukan untuk mengetahui performa *routing* dari *AODV-Backup Routing* dengan AODV berdasarkan parameter *Quality of Service (QoS)* pada jaringan, dalam penelitian ini parameter yang digunakan untuk dianalisa berdasarkan pada hasil *delay* dan *throughput* yang dihasilkan terhadap jumlah hop yang terlibat pada jaringan berdasarkan variasi data yang ditentukan, yaitu 32 KB, 64 KB, 128 KB.

4.1.1 *Delay*

Delay merupakan interval atau selisih waktu yang perlukan untuk pengiriman sebuah paket dari *node* pengirim ke *node* penerima. Pengukuran terhadap *delay* yaitu membandingkan waktu yang dibutuhkan oleh AODV-BR dengan AODV dalam sekenario pengukuran yang sama. Hasil simulasi menegenai parameter *delay* ini berdasarkan jumlah *hop* yang digunakan saat simulasi adalah sebagai berikut:

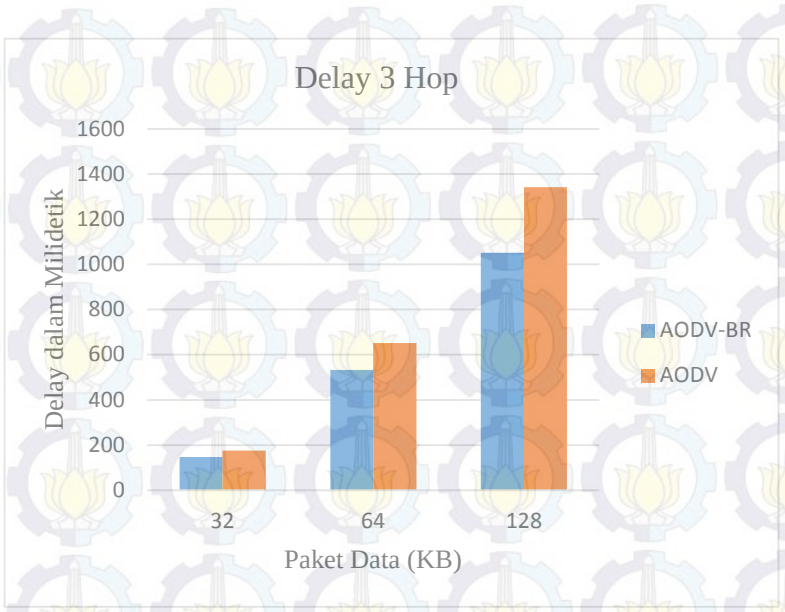
4.1.1.1 *Delay* untuk 3 hop

Untuk pengukuran *delay* yang dibutuhkan dalam pengiriman paket dengan jumlah 4 *hop*, dimana terdapat 4 *node* yang akan dilalui oleh paket data, yaitu dikirim dari *node* 1 ke *node* 4 atau sebaliknya yang melalui *node* 3 dan *node* 2 sebagai *intermediate node*. Data yang diukur dibandingkan antara AODV-*backup routing* dengan AODV tradisional dengan pengambilan sampel besaran data yang bervariasi yaitu 32KB, 64 KB, dan 128 KB.

Tabel 4.2 Delay yang Dibutuhkan untuk 3 *Hop* AODV-BR dan AODV

Perc. Ke	32 KB		64 KB		128 KB	
	AODV-BR	AODV	AODV-BR	AODV	AODV-BR	AODV
1	142	172	523	642	1077	1301
2	147	181	510	644	1067	1300
3	148	169	532	650	1000	1376
4	151	175	541	669	1010	1389
5	152	168	513	641	1045	1320
6	151	177	545	661	1002	1334
7	134	181	525	654	1082	1308
8	144	180	511	640	1100	1340
9	150	171	521	653	1092	1350
10	150	174	610	659	1120	1398
Rata-rata	**147**	**175**	**533**	**651**	**1051**	**1342**

Pada Tabel 4.2 merupakan hasil simulasi terhadap *delay* yang dibutuhkan untuk simulasi 3 *hop* pada protokol *routing* AODV-BR dan AODV tradisional. Perbedaan *delay* yang dibutuhkan antara AODV-BR dan AODV dengan besaran paket data yang bervariasi yaitu 32 KB, 64 KB, dan 128 KB dari 10 kali percobaan pada skenario yang sama, dapat disimpulkan kedalam hasil rata-rata dari sekian percobaan. Untuk variasi paket data 32 KB, rata-rata AODV-BR membutuhkan waktu pengiriman paket 147 milidetik sedangkan AODV 175 milidetik. Untuk 64 KB, AODV-BR membutuhkan 533 milidetik sedangkan AODV 651 milidetik. Untuk 128 KB, rata-rata AODV-BR membutuhkan 1051 milidetik sedangkan AODV membutuhkan 1342 milidetik. Untuk memudahkan menganalisa hasil, hasil data rata-rata yang diperoleh dapat diplot ke dalam grafik. Grafik rata-rata dari waktu yang dibutuhkan dalam pengiriman paket oleh AODV-BR dan AODV untuk 3 *hop* dapat dilihat pada Gambar 4.1 dibawah.



Gambar 4.1 Perbadingan Rata-rata *Delay* 3 *Hop*

Berdasarkan Gambar 4.1 diatas, terlihat perbedaan dalam kebutuhan *delay* terhadap variasi data yang dikirim terlihat begitu signifikan. AODV-BR membuktikan kebutuhan *delay* yang lebih rendah disetiap macam variasi data yang dikirim. Hal ini dikarenakan AODV-BR tidak membutuhkan *route discovery* kembali disaat terjadi *route break* melainkan secara otomatis memanfaatkan *routing* alternatif yang dibuat saat proses *setup route.* Sedangkan AODV membutuhkan *route discovery* dan *setup route* dari awal dikarenakan tidak adanya jalur alternatif dalam AODV. Dari jumlah 3 *hop* ini, estimasi waktu yang dibutuhkan AODV-BR lebih sedikit dari pada AODV tradisional.

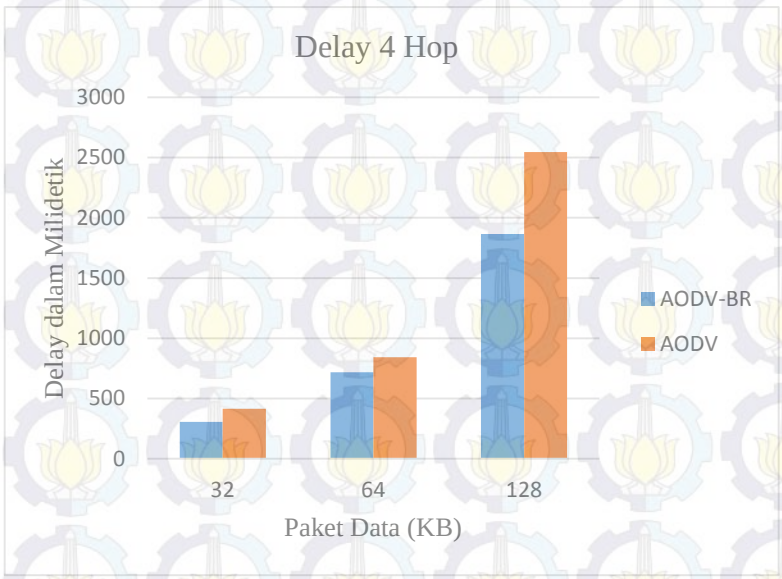
4.1.1.2 *Delay* untuk 4 *Hop*

Untuk pengukuran *delay* yang dibutuhkan dalam pengiriman paket dengan jumlah 4 *hop*, dimana terdapat 5 *node* yang akan dilalui oleh paket data, yaitu dikirim dari *node* 1 ke *node* 5 atau sebaliknya yang melalui *node* 4, *node* 3, node 2 sebagai *intermediate node*. Data yang diukur dibandingkan antara AODV-*Backup Routing* dengan AODV tradisional dengan pengambilan sampel besaran data yang bervariasi yaitu 32KB, 64 KB, dan 128 KB.

Tabel 4.3 *Delay* yang Dibutuhkan untuk 4 *Hop* AODV-BR dan AODV

Perc. Ke	32 KB		64 KB		128 KB	
	AODV-BR	AODV	AODV-BR	AODV	AODV-BR	AODV
1	200	390	661	812	1800	2510
2	232	395	665	822	1867	2598
3	220	399	687	831	1851	2577
4	312	407	691	832	1899	2501
5	323	413	703	833	1910	2489
6	331	417	714	843	1934	2563
7	335	422	737	851	1854	2570
8	351	427	765	852	1881	2562
9	376	431	773	876	1841	2542
10	382	436	782	881	1812	2550
Rata-rata	**306**	**414**	**718**	**843**	**1865**	**2546**

Pada Tabel 4.3 merupakan hasil simulasi terhadap *delay* yang dibutuhkan untuk simulasi 4 *hop* pada protokol *routing* AODV-BR dan AODV tradisional. Perbedaan rata-rata *delay* yang dibutuhkan antara AODV-BR dan AODV dengan besaran paket data yang ditentukan yaitu 32 KB, 64 KB, dan 128 KB dari 10 kali percobaan pada skenario pengujian yang sama, dapat diambil nilai rata-rata dari sekian percobaan untuk memudahkan dalam menyimpulkan hasil. Untuk variasi paket data 32 KB, rata-rata AODV-BR membutuhkan waktu pengiriman paket 306 milidetik sedangkan AODV 414 milidetik. Untuk 64 KB, AODV-BR membutuhkan 718 milidetik sedangkan AODV 843 milidetik. Untuk 128 KB, rata-rata AODV-BR membutuhkan 1865 milidetik sedangkan AODV membutuhkan 2546 milidetik. Untuk memudahkan proses analisa, hasil data rata-rata yang diperoleh dapat diplot ke dalam grafik. Grafik rata-rata dari waktu yang dibutuhkan dalam pengiriman paket oleh AODV-BR dan AODV untuk 5 *hop* dapat dilihat pada Gambar 4.2 dibawah.



Gambar 4.2 Perbadingan Rata-rata *Delay* 4 *Hop*

Berdasarkan Gambar 4.2 diatas, terlihat perbedaan dalam kebutuhan *delay* terhadap variasi data yang dikirim terlihat begitu signifikan. AODV-BR membuktikan kebutuhan *delay* yang lebih rendah disetiap macam variasi data yang dikirim. Hal ini dikarenakan AODV-BR tidak membutuhkan *setup route* dari awal disaat terjadi *route break* karana AODV-BR menyediakan alternatif *route*, sedangkan AODV membutuhkan *route discovery* dan *setup route* dari awal dikarenakan tidak adanya jalur arternatif dalam AODV. Dari jumlah 5 *hop* ini, estimasi waktu yang dibutuhkan AODV-BR lebih sedikit dari pada AODV tradisional.

4.1.2 *Throughput*

Throughput merupakan jumlah data yang sukses terkirim dalam ukuran detik melalui sebuah sistem atau media komunikasi. Untuk hasil simulasi, *throughput* didapatkan dari hasil *parsing* data pada akhir simulasi. Pengukuran *throughput* dilakukan untuk mengetahui kinerja dari protokol *routing* yang digunakan. *Throughput* berbanding lurus dengan *delay* yang dibutuhkan untuk pengiriman paket. Pengukuran terhadap *throughput* ini untuk mengetahui performa *routing* dari AODV-BR dan AODV tradisional yang tentunya dalam skenario

pengujian yang sama. Perbandingan untuk menganalisa *throughput* yang didapatkan berdasarkan jumlah *hop* yang digunakan waktu simulasi.

4.1.2.1 *Throughput* untuk 3 hop

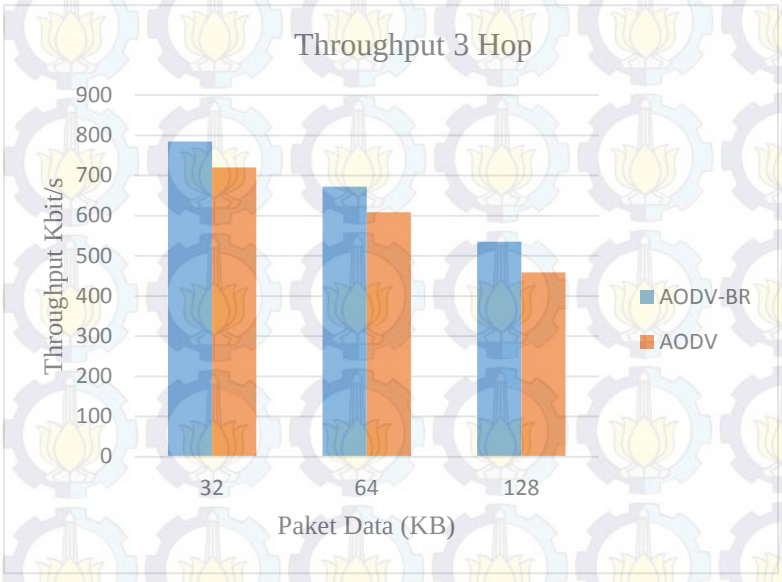
Pengukuran terhadap *throughput* paket per-detiknya dibutuhkan untuk menganalisa performa dari sebuah jaringan. Pada pengukuran 3 hop, terdapat 4 node yang akan dilalui data. Pada simulasi, diasumsikan semua *node* memiliki performa yang sama karena dalam kondisi ideal. Semakin tinggi *delay* yang dibutuhkan, otomatis nilai *throughput* akan semakin rendah. Untuk besaran variasi data yang digunakan dalam mengukur *throughput* sama besarnya saat pengukuran *delay* yaitu 32 KB, 64 KB, dan 128 KB. *Throughput* dan *delay* didapat secara bersamaan dalam setiap kali uji coba.

Tabel 4.4 *Throughput* yang Didapatkan untuk Simulasi 3 *Hop* AODV-BR dan AODV

Perc. Ke	32 KB		64 KB		128 KB	
	AODV-BR	AODV	AODV-BR	AODV	AODV-BR	AODV
1	800	729	663	640	554	467
2	792	701	654	613	556	462
3	791	731	693	616	571	452
4	768	726	651	598	518	443
5	767	732	678	596	531	480
6	785	724	667	613	532	448
7	807	701	676	615	551	450
8	798	704	699	623	508	465
9	770	730	650	582	524	463
10	770	726	690	590	501	461
Rata-rata	**785**	**720**	**672**	**609**	**535**	**459**

Pada Tabel 4.4 merupakan hasil pengukuran dalam simulasi terhadap *throughput* yang dihasilkan sesuai dengan *delay* pengiriman paket yang dibutuhkan. Rata-rata *throughput* yang dihasilkan pada pengujian simulasi 3 *hop* terhadap protokol routing AODV-BR memiliki nilai *throughput* yang lebih tinggi dari pada AODV tradisional. Hal ini dipicu oleh faktor *delay* yang dibutuhkan AODV saat terjadi *route break* pada waktu pengiriman paket data berlangsung.

Dari sekian data hasil pengujian simulasi yang bervariasi, untuk meudahkan dalam penyimpulan dibutuhkan nilai rata-rata terhadap *throughput* yang dihasilkan. Untuk variasi paket 32 KB, AODV-BR menghasilkan *throughput* rata-rata 785 Kbit/s sedangkan AODV 720 Kbit/s. Untuk 64 KB, AODV-BR menghasilkan *throughput* rata-rata sebesar 672 Kbit/s sedangkan AODV 609 Kbit/s. Untuk 128, AODV-BR menghasilkan *throughput* rata-rata sebesar 535 Kbit/s, sedangkan AODV 459 Kbit/s. Dari rata-rata yang didapatkan dapat diplot ke dalam grafik. Grafik rata-rata *throughput* yang dihasilkan oleh proses simulasi terhadap kinerja protokol *routing* AODV-BR dan AODV untuk 3 *hop* dapat dilihat pada Gambar 4.3 dibawah.



Gambar 4.3 Perbadingan Rata-rata *Throughput* 3 Hop

Berdasarkan Gambar 4.3 perbandingan *throughput* yang dihasilkan dari uji coba simulasi 3 *hop* pada AODV-BR menghasilkan nilai *throughput* yang lebih tinggi dari AODV. Hal ini dipicu oleh *delay* yang dibutuhkan AODV-BR lebih sedikit dari pada AODV tradisional.

4.1.2.2 *Throughput* untuk 4 hop

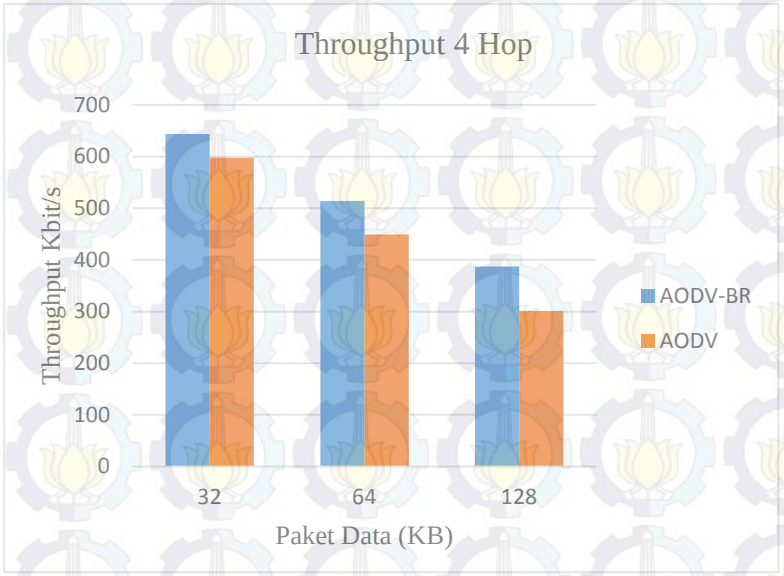
Pengukuran terhadap *throughput* yang dihasilkan oleh simulasi dengan 4 *hop* memiliki skenario yang sama dengan pengujian 3 *hop*. Perbedaan jumlah *node* yang awalnya 3 *hop* menjadi 4 *hop* memberikan kontribusi nilai hasil yang berbeda,

karena dipengaruhi oleh jumlah *node* yang terlibat. Adapun hasil pengukuran *throughput* dari hasil simulasi terhadap performa AODV-BR dan AODV tradisional terdapat pada Tabel 4.4 dibawah.

Tabel 4.5 *Throughput* yang Didapatkan untuk Simulasi 4 *hop* AODV-BR dan AODV

Perc. Ke	32 KB		64 KB		128 KB	
	AODV-BR	AODV	AODV-BR	AODV	AODV-BR	AODV
1	680	620	540	470	410	320
2	668	613	537	463	397	287
3	672	616	536	460	399	307
4	640	618	520	458	370	312
5	637	601	512	456	360	325
6	633	596	510	449	348	231
7	631	593	502	443	398	298
8	628	585	498	442	378	309
9	625	570	496	426	400	311
10	623	564	490	420	405	310
Rata-rata	**644**	**598**	**514**	**449**	**387**	**301**

Dari beberapa hasil uji coba simulasi terhadap performa routing AODV-BR dan AODV yang ditunjukkan pada Tabel 4.5 diatas dapat diambil nilai rata-rata dikarenakan data yang dihasilkan tidak sama dalam setiap percobaan disebabkan pergerakan *node* yang sifatnya acak dan untuk lebih mudahnya dalam menyimpulkan sebuah hasil. Hasil simulasi AODV-BR dan AODV tradisional yang melibatkan jumlah 4 *hop* atau 5 *node* dengan variasi sampel pengiriman data 32 KB, 64 KB, dan 128 KB. Untuk pengrimana data 32 KB, AODV-BR menghasilkan rata-rata *throughput* sebesar 644 Kbit/s, sedangkan AODV 598 Kbit/s. Untuk 64 KB, AODV-BR menghasilkan rata-rata *throughput* 514 Kbit/s, sedangkan AODV 449 Kbit/s. Untuk 128 KB, AODV-BR menghasilkan rata-rata *throughput* 387 Kbit/s, sedangkan AODV 301 Kbit/s. Untuk lebih mudahnya dalam proses analisa terhadap nilai rata-rata dari sepuluh percobaan pengambilan data, nilai rata-rata dapat diplot ke dalam grafik. Grafik rata-rata *throughput* yang dihasilkan dapat dilihat pada Gambar 4.4 dibawah.



Gambar 4.4 Perbadingan Rata-rata *Throughput* dalam 4 *Hop*

Berdasarkan grafik pada Gambar 4.4, AODV-BR memiliki hasil *throughput* yang lebih tinggi dari pada AODV tradisional dari tiga variasi data yang sudah ditentukan. Simulasi AODV-BR memiliki kinerja *routing* yang lebih baik dengan adanya penambahan *multiple route* atau *alternative route* pada AODV tradisional.

4.2 Hasil Pengujian *Testbed*

Hasil pengujian terhadap kinerja protokol *routing* AODV-BR pada VMeS menggunakan *testbed* laptop dengan memanfaatkan *wireless* IEEE 802.11. Proses pengujian sesuai dengan skenario yang sudah dibahas pada sub bab 3.4. Pengujian *testbed* dilakukan dengan tujuan untuk mengetahui kinerja protokol routing AODV-BR tidak hanya dengan hasil simulasi menggunakan *software* simulator saja, melainkan mengetahui secara riil performa dari protokol ini setelah diimplementasikan pada sebuah perangkat. Parameter *Quality of Service (QoS)* yang diukur terhadap performasi jaringan pada *testbed* disesuaikan dengan simulasi, yaitu mengukur nilai parameter *delay* yang dibutuhkan dan *throughput* yang dihasilkan.

4.2.1 *Delay*

4.2.1.1 *Delay* untuk 3 *hop*

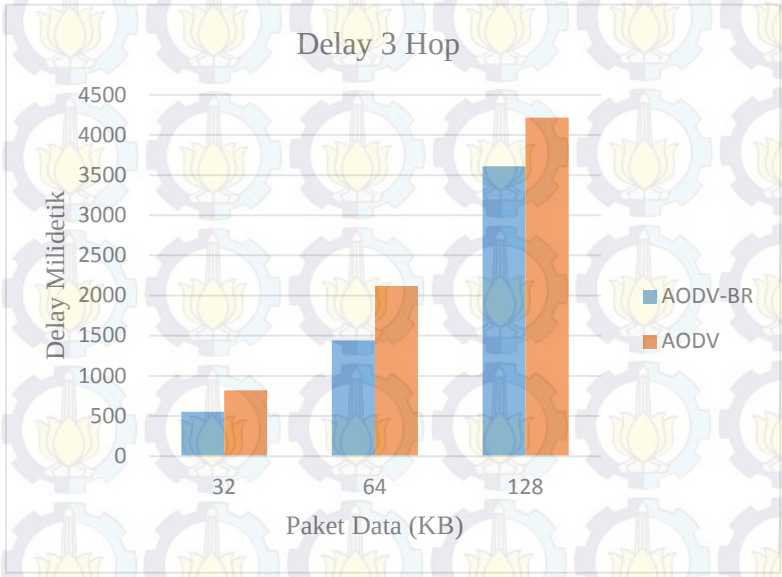
Untuk pengukuran *delay* yang dibutuhkan dalam pengiriman paket dengan jumlah 3 *hop*, dimana terdapat 4 *node* yang akan dilalui oleh paket data pada skema 4 *hop* yang akan dijadikan sebagai jalur alternatif. Paket dikirim dari *node* 1 (10.10.10.1) ke *node* 3 (10.10.10.3) atau sebaliknya yang melalui *node* 2 (10.10.10.2) sebagai *intermediate node* untuk *primary route*. Untuk pengiriman data melalui alternatif *route*, *node* 1 (10.10.10.1) ke *node* 3 (10.10.10.3) akan melalui *node* 4 (10.10.10.4) dan *node* 5 (10.10.10.5) sebagai *intermediate node*. Data *delay* yang diukur dibandingkan antara AODV-*Backup Routing* dengan AODV tradisional dengan pengambilan sampel besaran data yang bervariasi yaitu 32 KB, 64 KB, dan 128 KB.

Tabel 4.6 *Delay* untuk 3 *Hop* AODV-BR dan AODV pada *Testbed*

Perc. Ke	32 KB		64 KB		128 KB	
	AODV-BR	AODV	AODV-BR	AODV	AODV-BR	AODV
1	510	865	1401	2000	3156	4654
2	507	754	1400	2103	3231	4100
3	545	776	1476	2034	3547	4643
4	589	852	1489	2187	4721	4653
5	565	823	1420	2174	3800	3540
6	511	857	1434	2032	4752	4431
7	576	759	1408	2100	2439	3760
8	571	843	1440	2123	4100	5551
9	586	832	1450	2057	3587	3700
10	591	832	1498	2400	2770	3124
Rata-rata	**555**	**819**	**1442**	**2121**	**3610**	**4215**

Pada Tabel 4.6 merupakan hasil pengukuran yang dilakukan terhadap *delay* yang butuhkan pada saat uji coba pada perangkat dengan jumlah 3 *hop*. Perbedaan *delay* yang dibutuhkan dibedakan antara AODV-BR dan AODV tradisional. Besaran paket yang digunakan yaitu 32 KB, 64 KB, dan 128 KB. Uji coba dilakukan dalam skenario yang sama sampai 10 kali percobaan. Pada Tabel 4.6 terlihat perbedaan antara percobaan satu dengan lainnya, hal ini disebabkan pengaruh perubahan konfigurasi posisi dan lingkungan pada saat uji coba. Dari 10

kali percobaan yang dilakukan diambil nilai rata-rata untuk memudahkan dalam analisa. Hasil pengujian pada perangkat, untuk variasi paket data 32 KB, rata-rata AODV-BR membutuhkan waktu pengiriman paket 555 milidetik sedangkan AODV 819 milidetik. Untuk 64 KB, AODV-BR membutuhkan 1442 milidetik sedangkan AODV 2121 milidetik. Untuk 128 KB, rata-rata AODV-BR membutuhkan 3610 milidetik sedangkan AODV membutuhkan 4215 milidetik. Untuk memudahkan dalam analisa, hasil data rata-rata yang diperoleh dapat diplot ke dalam grafik. Grafik rata-rata dari waktu yang dibutuhkan dalam pengiriman paket oleh AODV-BR dan AODV untuk 3 *hop* dapat dilihat pada Gambar 4.5 dibawah.



Gambar 4.5 Perbadingan rata-rata *delay* dalam 3 *hop*

Pada Gambar 4.5 merupakan grafik perbandingan waktu yang diperlukan untuk pengiriman data pada 3 *hop* uji coba *routing* pada testbed. AODV-BR memberikan performasi waktu yang lebih rendah dari pada AODV tradisional. Pada grafik tersebut menunjukkan, semakin besar sampel data yang dikirim semakin besar pula *delay* yang dibutuhkan. Pada pengiriman data dengan kapasitas data 128 KB, membutuhkan *delay* 4-6 detik untuk 3 *hop* atau 4 *node* yang dilalui.

4.2.1.2 Delay untuk 4 *Hop*

Untuk pengukuran *delay* yang dibutuhkan dalam pengiriman paket dengan jumlah 4 *hop*, dimana terdapat 5 *node* yang terlibat dalam jaringan pada skema 4 *hop* yang akan dijadikan sebagai jalur alternatif. Sesuai dengan skenario pengujian, paket dikirm dari *node* 1 (10.10.10.1) ke *node* 4 (10.10.10.4) atau sebaliknya yang melalui *node* 2 (10.10.10.2) dan *node* 3 (10.10.10.3) sebagai *intermediate* node untuk *primary route*. Untuk pengiriman data melalui alternatif *route*, *node* 1 (10.10.10.1) ke *node* 4 (10.10.10.4) akan melalui node 2 (10.10.10.2), *node* 5 (10.10.10.5), dan *node* 6 (10.10.10.6) sebagai *intermediate node*. Data *delay* yang diukur dibandingkan antara AODV-*backup routing* dengan AODV tradisional dengan pengambilan sampel besaran data yang bervariasi yaitu 32 KB, 64 KB, dan 128 KB.

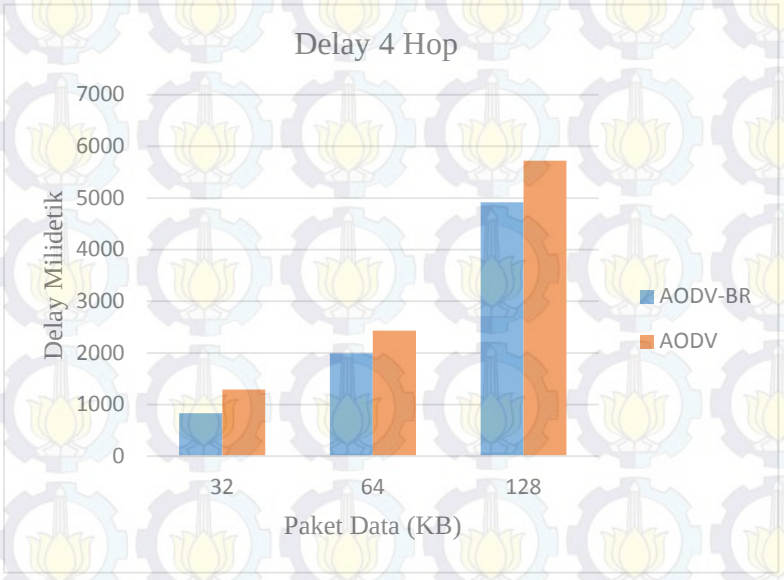
Tabel 4.7 *Delay* yang Dibutuhkan untuk 4 Hop AODV-BR dan AODV

Perc. Ke	32 KB		64 KB		128 KB	
	AODV-BR	AODV	AODV-BR	AODV	AODV-BR	AODV
1	800	1230	2000	2400	5654	5120
2	812	1283	1800	2430	5100	6044
3	801	1301	1832	2471	5643	6321
4	832	1324	2201	2429	4653	6542
5	854	1287	1983	2466	5540	5871
6	832	1293	2000	2412	4431	6232
7	866	1243	1832	2422	4760	5989
8	841	1382	2110	2462	4551	5821
9	841	1300	2265	2410	4700	4761
10	830	1280	1863	2420	4124	4541
Rata-rata	**831**	**1292**	**1989**	**2432**	**4916**	**5724**

Pada Tabel 4.7 merupakan hasil pengukuran yang dilakukan terhadap *delay* yang butuhkan pada saat uji coba pada perangkat dengan jumlah 4 *hop*. Perbedaan *delay* paket yang dibutuhkan dibedakan antara AODV-BR dan AODV tradisional. Besaran paket yang digunakan yaitu 32 KB, 64 KB, dan 128 KB. Uji coba dilakukan dalam skenario yang sama pada 10 kali percobaan. Pada Tabel 4.7 terlihat perbedaan antara percobaan satu dengan lainnya, hal ini disebabkan pengaruh perubahan konfigurasi posisi dan pengaruh lingkungan transmisi dari

setiap *node* yang berbeda-beda. Hasil pengujian pada perangkat, untuk variasi paket data 32 KB, rata-rata AODV-BR membutuhkan waktu pengiriman paket 831 milidetik sedangkan AODV 1292 milidetik. Untuk 64 KB, AODV-BR membutuhkan 1989 milidetik sedangkan AODV 2432 milidetik. Untuk 128 KB, rata-rata AODV-BR membutuhkan 4916 milidetik sedangkan AODV membutuhkan 5724 milidetik. Dari hasil pengukuran, tampak semakin bertambahnya node yang telibat untuk jadi hop dari *source* ke *destination* akan mempengaruhi terhadap *delay* yang dibutuhkan untuk setiap variasi data yang digunakan.

Untuk memudahkan dalam analisa, hasil data rata-rata yang diperoleh dapat diplot ke dalam grafik. Grafik rata-rata dari waktu yang dibutuhkan dalam pengiriman paket untuk menguji performa routingnya oleh AODV-BR dan AODV untuk 4 *hop* dapat dilihat pada Gambar 4.6 dibawah.



Gambar 4.6 Perbadingan Rata-rata *Delay* dalam 4 *Hop*

Pada Gambar 4.6 merupakan grafik perbandingan waktu yang diperlukan untuk pengiriman data pada 4 *hop* uji coba *routing* pada *testbed*. AODV-BR memberikan performasi waktu yang lebih rendah dari pada AODV tradisional. Pada grafik tersebut menunjukkan, semakin besar sampel data yang dikirim semakin besar pula *delay* yang dibutuhkan. Pada pengiriman data dengan kapasitas data 128 KB, membutuhkan *delay* 14-16 detik untuk 4 Hop atau 5 node yang dilalui.

4.2.2 *Throughput*

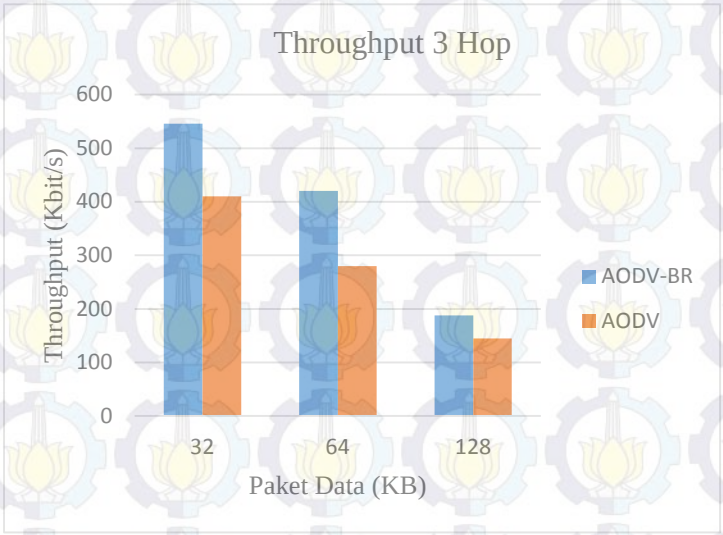
4.2.2.1 *Throughput* untuk 3 *hop*

Untuk hasil data pengukuran terhadap *throughput* yang dihasilkan dari uji coba implementasi pada *testbed*. *Throughput* yang dihasilkan sesuai dengan *delay* yang dibutuhkan, semakin tinggi *delay* pada proses pengiriman data, maka *throughput* yang dihasilkan akan semakin rendah. Untuk hasil pengukuran terhadap *throughput* untuk jarak 3 *hop* pada uji coba kinerja routing AODV-BR dan AODV terdapat pada Tabel 4.8 dibawah.

Tabel 4.8 *Throughput* yang Didapatkan untuk 3 *Hop* AODV-BR dan AODV

Perc. Ke	32 KB		64 KB		128 KB	
	AODV-BR	AODV	AODV-BR	AODV	AODV-BR	AODV
1	591	388	430	320	201	141
2	598	429	432	292	192	155
3	571	418	409	301	185	142
4	518	394	415	279	180	142
5	531	410	427	281	172	147
6	532	412	423	206	179	149
7	586	423	429	295	188	138
8	508	407	420	285	215	146
9	524	400	419	298	183	140
10	501	415	400	243	180	152
Rata-rata	**546**	**410**	**420**	**280**	**188**	**145**

Pada Tabel 4.8 merupakan data pengukuran terhadap *throughput* yang dihasilkan pada saat uji coba pada perangkat pada jarak 3 *hop* atau 4 *node* yang akan dilalui data dari sumber ke penerima. Dari 10 kali percobaan pengambilan data menunjukkan hasil yang berbeda, untuk memudahkan dalam mengambil kesimpulan maka perlu diambil nilai rata-rata dari setiap percobaan. Pada variasi data sebesar 32 KB, AODV-BR mengahasilkan rata-rata throughput 546 Kbit/s, sedangkan AODV 410 Kbit/s. Untuk 64 KB AODV-BR menghasilkan rata-rata *throughput* 420 Kbit/s, sedangkan AODV 280 Kbit/s. untuk 128 KB, AODV-BR menghasilkan rata-rata throughput 188 Kbit/s, sedangkan AODV 145 Kbit/s. Dari nilai rata-rata diatas dapat diplot ke dalam grafik pada Gambar 4.7.



Gambar 4.7 Perbadingan Rata-rata *throughput* dalam 3 *Hop*

4.2.2.2 *Throughput* untuk 4 hop

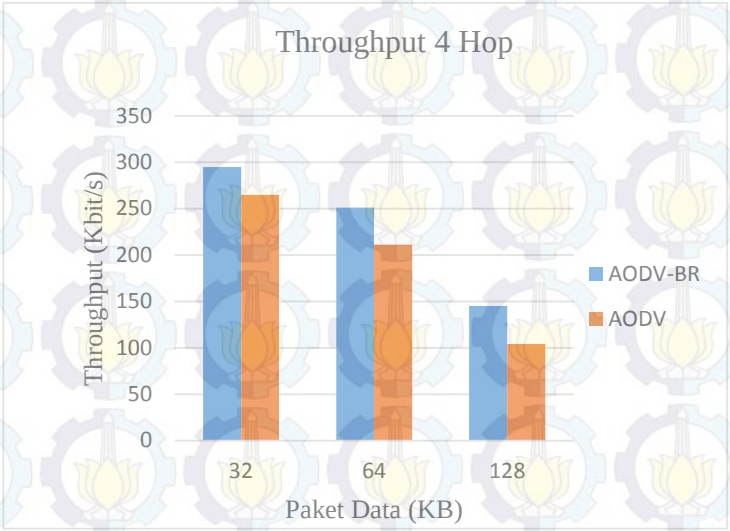
Untuk hasil pengukuran terhadap *throughput* untuk jarak 4 *hop* pada uji coba kinerja *routing* AODV-BR dan AODV pada *testbed* laptop terdapat pada Tabel 4.9 dibawah.

Tabel 4.9 *Throughput* yang Didapatkan untuk 4 *Hop* AODV-BR dan AODV

Perc. Ke	32 KB		64 KB		128 KB	
	AODV-BR	AODV	AODV-BR	AODV	AODV-BR	AODV
1	312	226	255	217	141	107
2	307	223	263	213	155	116
3	311	317	261	201	142	105
4	296	267	235	214	142	100
5	280	320	258	202	147	98
6	298	219	255	216	149	103
7	275	224	261	215	138	114
8	285	312	245	203	146	98
9	285	217	229	216	140	99
10	298	321	250	214	152	100
Rata-rata	**295**	**265**	**251**	**211**	**145**	**104**

Pada Tabel 4.9 merupakan data pengukuran terhadap *throughput* yang dihasilkan pada saat uji coba pada perangkat pada jarak 4 *hop* atau 5 *node* yang akan dilalui data dari sumber ke penerima. Dari 10 kali percobaan pengambilan data

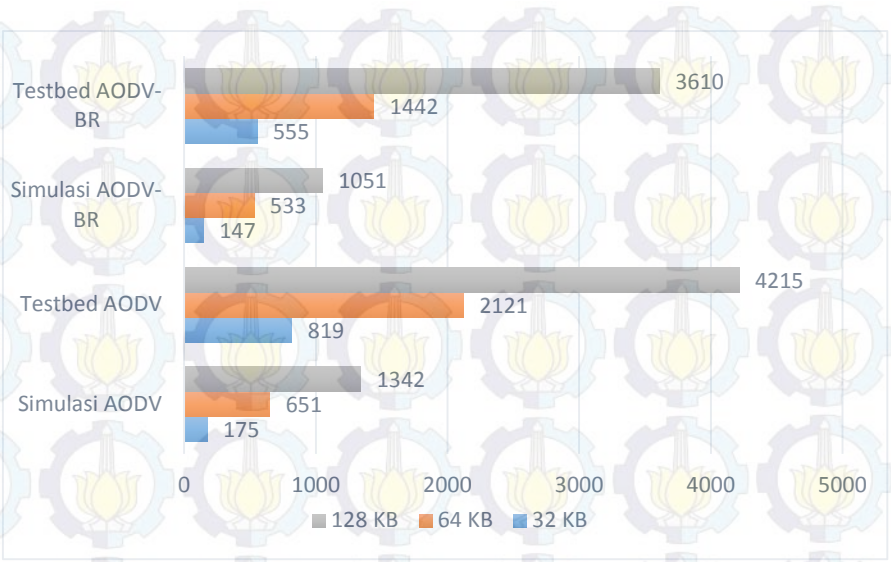
menunjukkan hasil yang berbeda, untuk memudahkan dalam mengambil kesimpulan maka perlu diambil nilai rata-rata dari setiap percobaan. Pada variasi data sebesar 32 KB, AODV-BR menghasilkan rata-rata *throughput* 295 Kbit/s, sedangkan AODV 265 Kbit/s. Untuk 64 KB AODV-BR menghasilkan rata-rata *throughput* 251 Kbit/s, sedangkan AODV 211 Kbit/s. Untuk 128 KB, AODV-BR menghasilkan rata-rata throughput 145 Kbit/s, sedangkan AODV 104 Kbit/s. Dari nilai rata-rata diatas dapat diplot ke dalam grafik pada Gambar 4.8 dibawah.



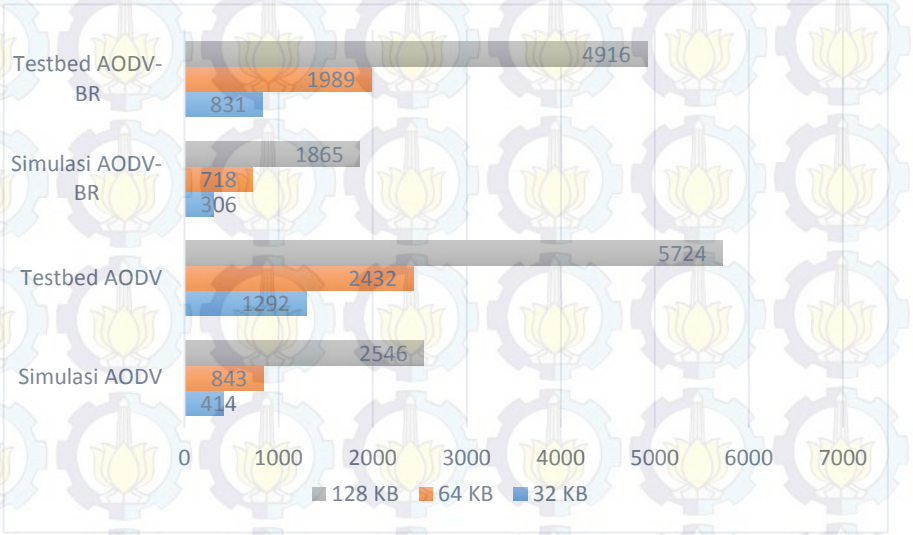
Gambar 4.8 Perbadingan Rata-rata *throughput* dalam 3 *Hop*

4.3 Analisa dan Pembahasan

Untuk mengetahui dan memaparkan hasil analisa terhadap data-data hasil pengukuran simulasi maupun uji coba pada *testbed* dapat dibandingkan, agar dapat menganalisa kinerja dari protokol AODV-BR terhadap pengaruh antara pertambahan jumlah *node* dengan jumlah data yang ditransmisikan. Untuk melakukan analisa terhadap hasil pengukuran pada parameter *QoS*, dipaparkan hasil perbandingann antara hasil pengukuran simulasi pada *network simulator* dengan hasil pengukuran pada *testbed* laptop.



Gambar 4.9 Perbandingan Rata-rata *Delay* Simulasi dan *Testbed* AODV-BR dengan AODV untuk 3 *Hop*



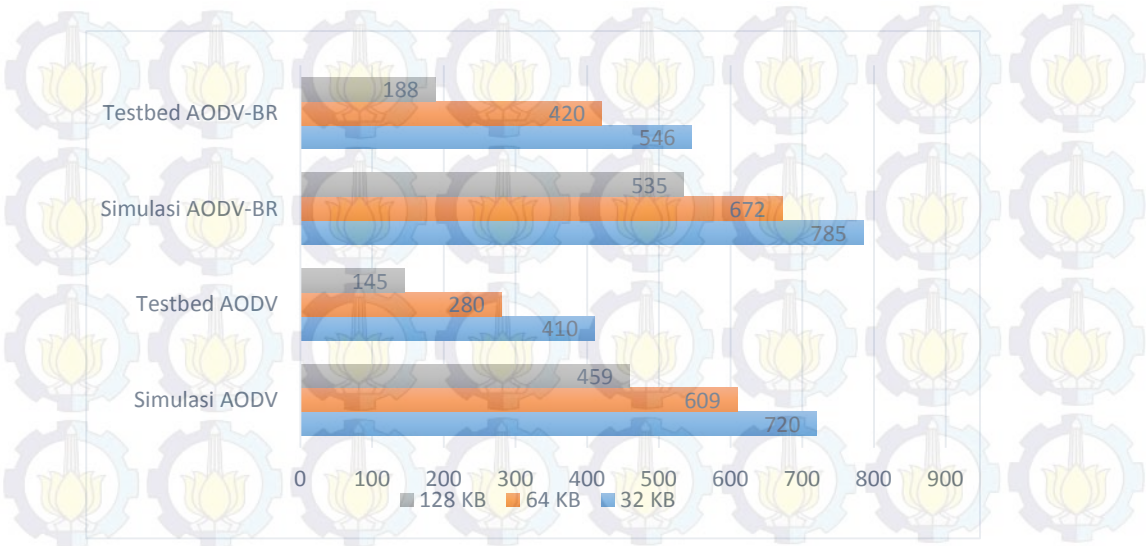
Gambar 4.10 Perbandingan Rata-rata *Delay* Simulasi dan *Testbed* AODV-BR dengan AODV untuk 4 *Hop*

Pada Gambar 4.9 Menunjukkan grafik perbandingan *delay* antara hasil simulasi dan hasil *testbed* untuk 3 hop dan pada Gambar 4.10 menunjukkan grafik perbandingan *delay* antara hasil simulasi dan hasil *testbed* untuk 4 *hop*. Perbedaan *delay* yang sangat signifikan terjadi pada pengiriman paket data sebesar 128 KB, dari masing-masing variasi jumlah *hop* yang diuji coba baik pada AODV-BR maupun pada AODV pada hasil *testbed*. *Delay* yang tinggi pada *testbed* dipengaruhi oleh faktor lingkungan pada saat melakukan uji coba *routing* dan faktor

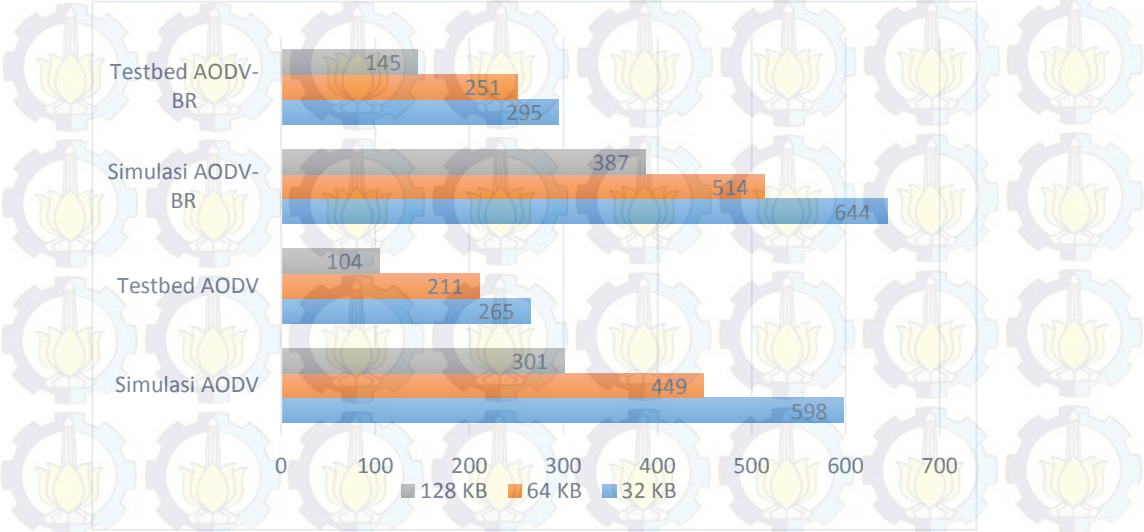
perangkat yang digunakan. Setiap *node* yang digunakan memiliki performa yang berbeda, sehingga mengakibatkan *delay* yang tinggi untuk *testbed* 3 *hop* hingga 3610 milidetik untuk AODV-BR dan 4215 milidetik untuk AODV, dan untuk *testbed* 4 *hop* juga mengalami peningkatan yang jauh lebih besar hingga 4916 milidetik untuk AODV-BR dan 5724 milidetik untuk AODV. Kebutuhan *delay* yang besar dipengaruhi oleh faktor besaran data yang ditransmisikan serta beberapa faktor lain seperti faktor *delay* perangkat yang digunakan serta proses *route discovery* pada masing-masing protokol *routing*.

Untuk rata-rata pada pengujian *testbed* 3 – 4 hop, AODV-BR memiliki estimasi waktu yang lebih rendah dari pada AODV, disebabkan pada waktu pengujian diskenariokan jalur *primary* putus agar *routing* AODV-BR dapat secara otomatis memanfaatkan jalur alternatif yang dijadikan kontribusi perbaikan terhadap AODV tradisional. Skenario pengujian yang sama juga diterapkan pada AODV tradisional agar dapat membandingkan hasil kinerja apabila terjadi *route break* pada saat melakukan komunikasi. AODV membutuhkan *delay* yang lebih lama walaupun dalam kondisi skenario pengukuran yang sama, disebabkan AODV tradisional perlu melalukan *setup route* kembali saat jalur yang digunakan mengalami *route break*. Berdasarkan grafik yang ditampilkan pada Gambar 4.9 dan Gambar 4.10 AODV-BR memiliki estimasi waktu $\pm$ 10% s/d 15% lebih kecil dari pada *delay* yang dibutuhkan oleh AODV tradisional.

Untuk delay yang dihasilkan simulasi yang ditunjukkan dalam grafik pada gambar rata-rata antara tiga jenis variasi yang digunakan masih dalam kondisi stabil dikarenakan pada simulasi, perangkat yang digunakan diasumsikan sama, dan delay pada setiap node diasumsikan 0. Delay yang berpengaruh pada simulasi hanya delay yang disebabkan oleh *route discovery* dan waktu pengiriman paket.



Gambar 4.11 Perbandingan Rata-rata *Throughput* Simulasi dan *Testbed* AODV-BR dengan AODV untuk 3 *Hop*



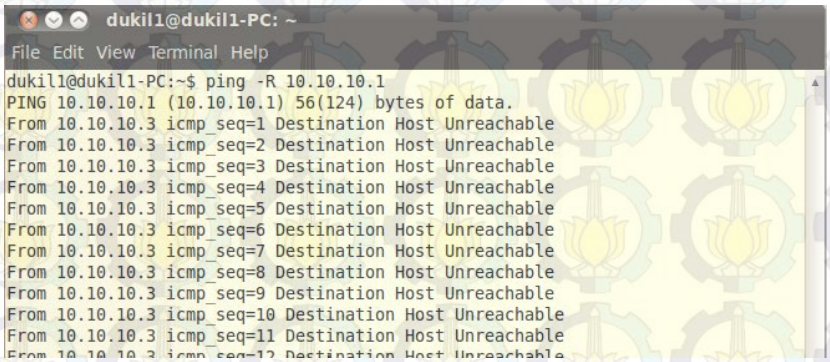
Gambar 4.12 Perbandingan Rata-rata *Throughput* Simulasi dan *testbed* AODV-BR dengan AODV untuk 4 *Hop*

Pada Gambar 4.11 dan Gambar 4.12 menunjukkan grafik perbandingan terhadap rata-rata *throughput* yang dihasilkan dari keseluruhan pengujian terhadap kinerja protokol routing AODV-BR dan AODV tradisional. *Throughput* yang dihasilkan berbading lurus dengan *delay* yang dibutuhkan. Semakin besar *delay* yang dibutuhkan, maka *throughput* yang dihasilkan semakin rendah. Pada hasil *testbed*, rata-rata antara uji coba pada 3 *hop* dan 4 *hop* mengalami penurunan dalam setiap variasi kapasitas data yang dikirimkan, baik pada AODV-BR maupun AODV

tradisional. *Throughput* tertinggi yang dihasilkan dari pengukuran terhadap kinerja *routing* AODV-BR maupun AODV tradisional yaitu pada hasil simulasi, dikarenakan pada simulasi *delay* yang dibutuhkan sangat rendah. *Throughput* tertinggi yang dihasilkan oleh AODV-BR mencapai 785 Kbit/s pada uji coba simulasi 3 *hop* dan 644 Kbit/s pada uji coba simulasi 4 *hop* untuk variasi data 32 KB. Sedangkan *throughput* tertinggi yang dihasilkan AODV mencapai 720 Kbit/s pada uji coba 3 hop dan 598 Kbit/s pada uji coba simulasi 4 hop untuk variasi data 32 KB.

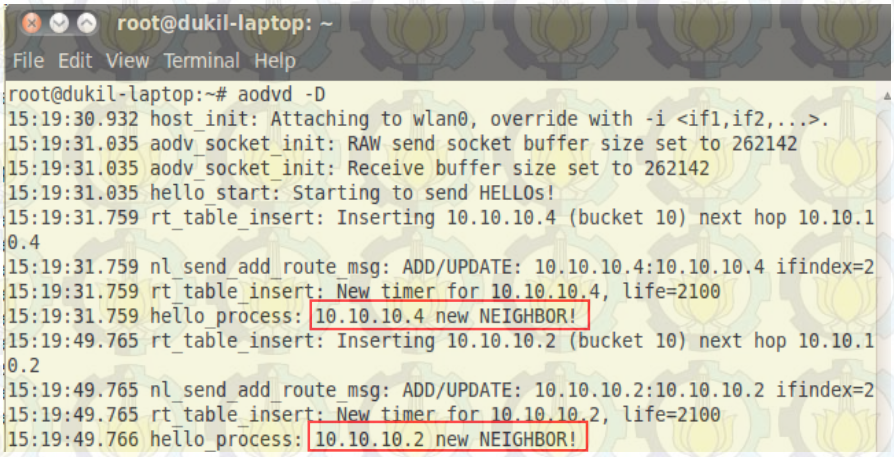
4.4 Uji Kemampuan *Routing Testbed*

Pengujian ini dilakukan setelah semua *node* yang akan dilibatkan dalam pengukuran yang terdiri dari 3 dan 4 *hop* ditata sesuai topologi jaringan untuk pengujian. Sebelum melakukan *cheking* terhadap jalur yang akan dilalui, setiap *node* harus menjalankan AODV-BR secara serempak untuk dapat melakukan uji coba *routing*. Pertama dilakukan uji coba koneksi dari *node* 3 ke *node* 1 tanpa menjalankan *routing* protokol.



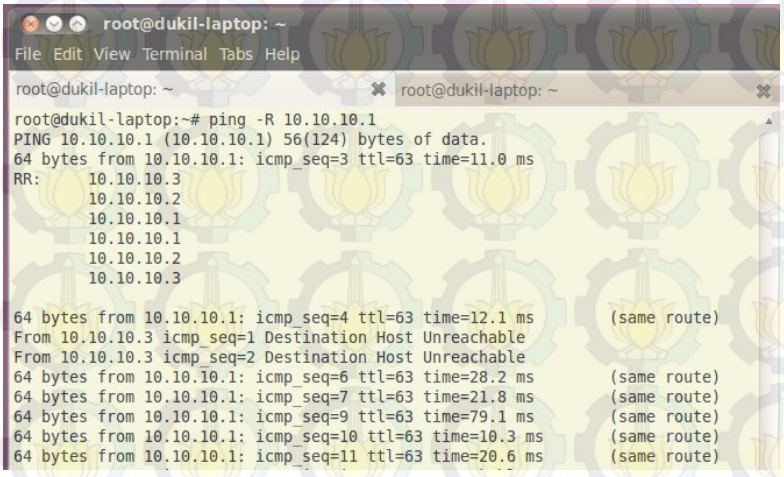
Gambar 4.13 Tidak Ada Koneksi dari *Node* 3 ke 1

Gambar 4.13 di atas menampilkan tidak adanya koneksi antara *node* 3 (10.10.10.3) ke *node* 1 (10.10.10.1) disebabkan *node* 2 (10.10.10.2) yang dijadikan *next hop* oleh *node* 3 menuju node 1 sebagai jalur *primary* tidak terditeksi adanya protokol *routing*. Atau melewati *node* 5 (10.10.10.5) dan *node* 4 (10.10.10.4) sebagai jalur alternatif.



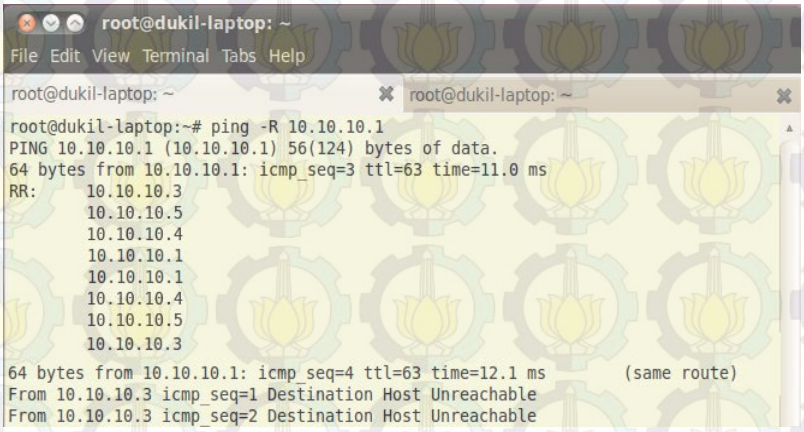
Gambar 4.14 Protokol *Routing* Mendeteksi *Node* Tetangga *(neighbor)*

Pada Gambar 4.14 protokol *routing* AODV-BR mendeteksi *node* tetangga dan melakukan *update table routing* secara berkala pada setiap *node*. *Broadcast* pesan *hello* digunakan untuk mendeteksi *node* tetangga untuk dijadikan *next hop*. Secara *default*, *broadcast hello* dilakukan secara periodik per 2100 milidetik. Setelah masing-masing *node* melakukan *update routing* pada setiap *node*. Selanjutnya, dilakukan pengujian *routing* menggunakan perintah *ping* dengan ekstensi –R untuk melihat rute yang akan dilalui.



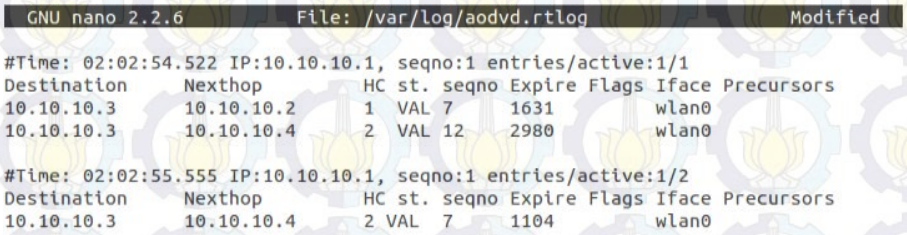
Gambar 4.15 Uji *Routing* untuk 3 *Hop Primary Route*

Pada Gambar 4.15 melakukan *Ping* dengan ekstensi –R dari *node* 3 ke *node* 1 dengan menampilkan *next hop* yang akan dilalui paket yaitu *node* 2. Mengacu pada skenario pengujian, *primary route* akan diputus atau dikeluarkan dari *coverage area* agar secara otomatis AODV-BR memanfaatkan jalur alternatif.



Gambar 4.16 Uji *Routing* untuk 3 *hop* Jalur Alternatif

Pada Gambar 4.16 menampilkan jalur yang akan dilalui dalam pengujian 3 *hop*, dimana jalur alternatif memiliki jumlah *next hop* yang lebih besar dari pada jalur *primary*. Untuk pengujian pada 4 *hop*, akan dilakukan pada skenario yang sama. Pengujian koneksi *routing* ini dilakukan pada saat proses pengiriman sampel paket data *dummy* menggunakan *tool iperf* dengan besaran paket yang sudah ditentukan.



Gambar 4.17 Tabel *Routing* AODV-BR pada *Node* 1

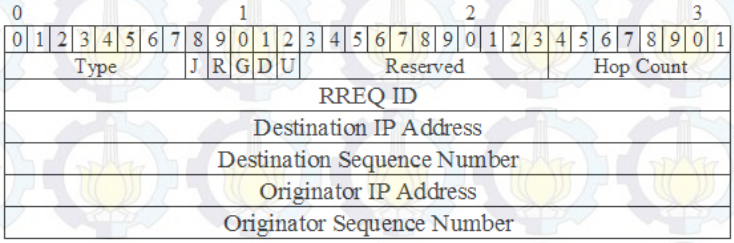
AODV-BR menyediakan alternatif *route*. alternatif *route* akan difungsikan sesuai dengan skenario pengujian. AODV-BR akan memanfaatkan *primary route* jika tidak terjadi *route break*. Proses yang terlihat pada Gambar 4.17 beberapa detik kemudian, AODV-BR menggeser *alternative route* ke urutan utama dan menghapus jalur *primary* dan tetap melanjutkan pengiriman paket data dikarenakan *node* jalur utama dikeluarkan dari *caverage area*. AODV-BR dalam menentukan jalur utama dan jalur alternatif disesuaikan dengan jumlah *hop count* yang akan dilalui. Jumlah *hop cont* yang lebih sedikit akan diprioritaskan.

4.5 Pembahasan Rekomendasi Routing Protokol

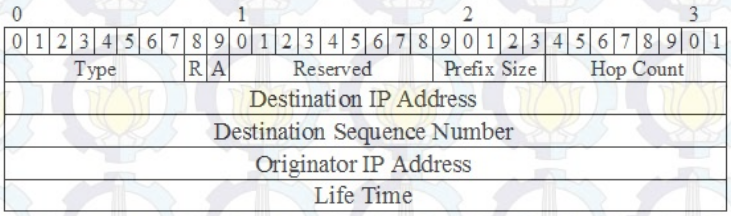
Hasil pengujian kinerja terhadap protokol AODV-BR dibandingkan dengan AODV tradisional menunjukkan AODV-BR lebih efektif diterapkan untuk VMeS berdasarkan hasil parameter *delay* yang dihasilkan dari pengukuran. AODV-BR mampu mereduksi *delay* yang digunakan oleh AODV, delay ditimbulkan berdasarkan skenario pengukuran disebabkan oleh pemodelan pengukuran, bukan dikarenakan kapasitas *overhead*. *Delay* yang kecil cocok diterapkan pada data rate yang rendah seperti HF/VHF yang hanya memiliki *data rate* 1200 bit [4].

Penelitian ini menggunakan MAC protokol 802.11 pada lapisan *datalink*-nya, dimana MAC protokol 802.11 memiliki kapasitas *frame* yang tinggi hingga 2346 *byte* (2,3 KB). Sedangkan perangkat VMeS yang akan dirancang pada protokol MAC-nya menggunakan Amatir AX.25 (AX.25) disebabkan lapisan fisik yang digunakan adalah perangkat yang mendukung frekuensi HF/VHF seperti *handy talky* dll. AX.25 memiliki kapasitas *frame* yang lebih pendek dari pada protokol MAC 802.11, yaitu 256 *byte* (0,25 KB). Kapasitas *frame* yang lebih sedikit diperlukan optimasi pada setiap kapasitas paket yang dikirimkan oleh lapisan *Network* melalui lapisan *Data Link* agar dapat mengoptimalkan kinerja dari suatu layanan sistem telekomunikasi.

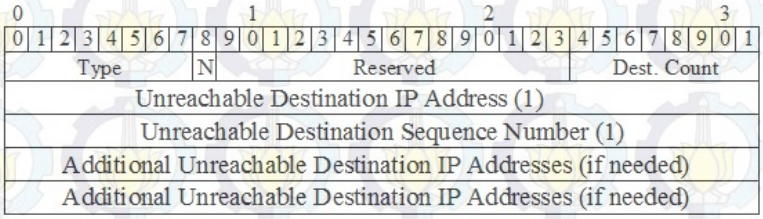
Pembahasan ini berdasarkan referensi standar protokol *routing* dan protokol MAC yang sudah digunakan. Protokol routing AODV yang digunakan memiliki tiga komponen paket utama dalam proses penentuan jalur, yaitu RREQ, RREP, dan RRER. Masing-masing paket memiliki total *bit* yang berbeda, hal ini dapat dianalisa dengan membandingkan dengan kapasitas *frame* yang dimiliki oleh protokol MAC yang digunakan.



Gambar 4.18 Format Pesan *RREQ (Route Request)* [14]

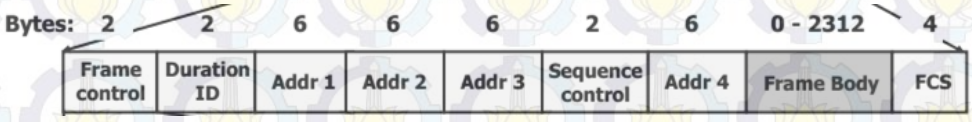


Gambar 4.19 Format Pesan *RREP (Route Reply)* [14]



Gambar 4.20 Format Pesan *RRER (Route Error)* [14]

Gambar 4.18 merupakan format RREQ yang difungsikan untuk permintaan pembuatan jalur memiliki panjang 32 bit, dimana total dari keseluruhan paket adalah 24 *Byte*. Gambar 4.19 merupakan format RREP yang digunakan untuk *setup route*, memiliki panjang 32 bit dengan total kapasitas paket 20 *Byte*. Gambar 4.20 merupakan format RRER yang digunakan informasi apabila terjadi *link break* atau *route error*, memiliki panjang 32 bit dengan total kapasitas paket 20 *Byte*.



Gambar 4.21 Fromat *Frame* 802.11 [15]

Gambar 4.21 merupakan format *frame* pada protokol MAC 802.11 layer data link. Dapat dilihat kapasitas dari *field frame body* memiliki pajang dari 0 –

2312 byte. *Frame body* merupakan alokasi kapasitas *byte* yang dikhususkan untuk kapasitas paket yang akan ditransmisikan dalam bentuk *frame*. RREQ memliki kapasitas 24 *Byte*, RREP memiliki kapasitas 20 Byte, dan RRER memiliki kapasitas 20 *Byte*. Kapasitas 24, 20, 20 *Byte* ditransmisikan satu-satu sesuai dengan kebutuhan sistem pada setiap *frame* pada layer *data link*. Kapasitas paket 24, 20 *Byte* jika dibandingkan dengan panjang dari *frame* yang disedikan protokol MAC 802.11 hanya digunakan 1 % saja dari total keseluruhan kapasitas dalam satu *frame*. Sehingga beban penggunaan data paket RREQ, RREP, dan RERR AODV pada *medium access control* 802.11 tidak memiliki pengaruh yang siginifikan terhadap beban kapasitas *frame*.

Flag	AX.25 Transfer Frame Header (128 bits)				Information Field	Frame-Check Sequence	Flag
	Destination Address	Source Address	Control Bits	Protocol Identifier			
8	56	56	8	8	0-2048	16	8

Gambar 4.22 Fromat *frame* AX.25 [17]

Gambar 4.21 merupakan format *frame* yang dimiliki oleh AX.25, pada *field info* memiliki kapasitas maksimal 256 *byte*. Kapasitas paket yang akan dikirimkan dari Layer Network akan diletakkan dalam *frame Information Field*, kapasitas paket apabila memiliki kapasitas melebihi 256 *byte* akan dipecah menjadi beberapa *frame* pada saat ditransmisikan sesuai dengan standar yang dimiliki oleh protokol lapisan *Data Link*. Kapasitas paket yang akan dikirimkan pada AODV adalah 20 – 24 *byte* untuk pembentukan suatu jalur sesuai standar protokol *routing* yang digunakan. Penggunaan *frame* untuk *overhead* digunakan 9,5 % dari total keseluruhan kapasita *frame* pada *field* informasi.

AODV dalam melakukan proses penentuan jalur pada layer *network*, melakukan pengriman paket pembentukan jalur secara periodik sesuai kebutuhan. Data yang dikirimkan dalam AODV tidak disertakan dengan paket penemuan jalur. Paket penemuan jalur dikatakan sebagai *overhead* apabila disertakan dengan data setelah melalui proses enkapsulasi.

Pada protokol MAC 802.11 penggunaan paket data pesan RREQ, RREP, dan RERR hanya menggunakan 1% dari keseluruhan *frame*, sedangkan pada

Protokol MAC AX.25 menggunakan 10% dari keseluruhan *frame*. Tidak perlu dilakukan modifikasi terhadap standar paket pesan dari protokol routing AODV disebabkan *overhead* tidak dikirimkan bersamaan dengan data seperti pada protokol TCP IP. Pada AODV data selain *overhead* dikirimkan setelah jalur terbentuk sempurna.

Jika pada penelitian selanjutnya hendak memodifikasi pada setiap jenis paket pesan pada AODV, dapat dilakukan kalkulasi terhadap kapasitas paket, mengurangi kapasitas *bit* setiap *field*, dan membuang *field* yang tidak terlalu digunakan yang ada pada paket pesan standar AODV sehingga bisa mencapai 5% dari penggunaan alokasi bit *frame* pada protokol AX.25. Rekomendasi ini, dapat dijadikan referensi untuk perancangan protokol *routing* pada penelitian selanjutnya yang akan diterapkan pada sistem dengan menggunakan protokol MAC AX.25.

DAFTAR PUSTAKA

[1] M. Ardita and A. Affandi, “Perancangan Terminal Komunikasi Data Terintegrasi untu Jaringan Ad-Hoc Vessel Messagging System (VMes)”, Surabaya: Thesis JTE-ITS, 2010.

[2] A. Affandi, "Sistem Komunikasi Data Terpadu Armada Perahu Nelayan Menggunakan Kanal Frekuensi Tinggi (VMeS-Vessel Messagging System)," in Hibah Pasca (HPTP), ITS Surabaya, 2007.

[3] A. Boukerche, “Algorithms and Protocols for Wireless and Mobile Adhoc Network”, Canada: A John Wiley & Sons, Inc. Publication, 2009.

[4] B. Y. Pratomo and A. Affandi, “Implementasi Protokol Jaringan Ad Hoc Pada Teminal Komunikasi Data untuk Sistem Komunikasi Kapal Laut”, Surabaya: JTE-ITS, 2011.

[5] K. Syarif, A. Affandi and D. S. Rahardjo, “Analisa Kinerja Protokol Routing Ad-Hoc on Demand Distance Vector (AODV) Pada Komunikasi VMeS”, Surabaya: JTE ITS, 2012.

[6] C. Perkins and E. M. Royer, "Ah-hoc On-demand Distance Vector Routing," in WMCSA'99 Second IEEE Workshop, New Orleans, LA, 1999.

[7] E. B. Royer, C. Perkins and S. Das, “Ad hoc On-Demand Distance Vector (AODV) Routing”, Internet Society, 2003.

[8] M. G. SJ Lee, "AODV -BR: Backup Routing in Ad hoc Networks," in IEEE, Chicago, IL, 2000.

[9] W. K. Lai, S. Y. Hsiao and Y. C. Lin, "Adaptive backup routing for ad-hoc networks," Elsevier, vol. 30, no. Computer Communications, pp. 453-464, 2007.

[10] F. P. T. A. A. Andhika, "Protokol Interchangeable Data pada VMeS (Vessel Messaging System) dan AIS (Automatic Identification System)," in Jurnal Teknik ITS, September 2012.

[11] A. A. Fuad Ahmad, “Mekanisme Komunikasi Data Gateway Multi-Terminal VMeS jaringan ad hoc Menggunakan Konsep Metode Akses Protokol MAC 802.11”, Surabaya: Tesis JTE-FTI-ITS, 2012.

[12] S. K. Sarkar, T. G. Basavaraju and C. Puttamadappa, “Ad Hoc Mobile Wireless Networks: Principles, Protocols and Applications”, France: Auerbach Publications, 2007.

[13] X. Hong, K. Xu and M. Gerla, "Scalable Routing Protocols for Mobile Ad Hoc Networks," IEEE, vol. 16, no. 4, pp. 11 - 21 , 2002.

[14] E. B. R. C. Perkins, “Ad hoc On-Demand Distance Vector (AODV) Routing”, RFC 3561, 2003.

[15] N. Pietro, "IEEE 802.11 frame format," www.studioreti.it, 2005.

[16] D. W. S. S. H. A. Prasetyo, "Perancangan dan Impelementasi Protokol AX.25 (Simplefield) pada MCS-51," Prosiding Seminar Nasional Sistem dan Informatika, 2007.

[17] N. D. E. T. J. B. A. William, “AX.25 Link Access Protokol for Amateur Packet Radio”, Tucson Amateur Packet Radio Corporation, 1997.

[18] T. Issariyakul and E. Hossain, “Introduction to NS 2”, Springer Science, 2009.

BAB 5
PENUTUP

5.1 Kesimpulan

Berdasarkan hasil analisa dan pengujian sistem yang dilakukan pada penelitian ini, yaitu mengenai analisa kinerja protokol *routing* AODV-BR pada komunikasi VMeS dapat diambil kesimpulan sebagai berikut:

1. Hasil pengukuran menunjukkan, perbandingan delay antara AODV-BR lebih kecil dari pada AODV tradisional dengan berbagai jenis variasi data yaitu 23, 64, 128 KB dengan jumlah hop yang sudah ditentukan.
2. *Throughput* yang dihasilkan oleh AODV-BR lebih besar dari pada AODV disebabkan AODV-BR tidak terjadi *routing loop* saat terjadi *route break* sehingga data yang ditransmisikan dengan teknik penambahan skema *backup routing* lebih banyak dari pada AODV tradisional.
3. Protokol routing AODV-BR memberikan teknik skema routing pada *layer network* untuk mengembangkan skema routing pada AODV tradisional dengan menambah *multipath routing* untuk mengatasi masalah *route break* yang berpengaruh terhadap meningkatnya *delay* yang dibutuhkan.
4. Hasil simulasi menunjukkan perbedaan yang signifikan dibandingkan dengan hasil testbed terhadap kinerja dari protokol routing yang dilakukan uji coba, dikarenakan pada testbed, faktor eksternal seperti lingkungan dan perangkat masih dipertimbangkan.
5. Penambahan jumlah node yang terlibat dalam komunikasi dan bertambahnya variasi data memiliki pengaruh terhadap *delay* yang dibutuhkan dan *throughput* yang dihasilkan terhadap AODV-BR dan AODV tradisional. Semakin bertambah jumlah node yang terlibat dan bertambahnya kapasitas data yang ditransmisikan, maka *delay* yang dibutuhkan semakin meningkat, dan *throughput* yang dihasilkan semakin menurun.

6. Hasil pengujian yang dilakukan didapatkan perbadingan kinerja dari AODV-BR pada VMeS berdasarkan *delay* yang dibutuhkan lebih sedikit dari pada AODV tradisional ± 10-15 % untuk hasil uji coba *testbed*.
7. AODV-BR dapat diimplementasikan di VMeS dikarenakan konsep yang ditawarkan pada AODV-BR sama dengan AODV tradisional, dimana teknik routing yang digunakan berdasarkan *on-demand*. AODV-BR akan bekerja saat dibutuhkan saja dan sangat cocok untuk sistem komunikasi yang memiliki *data rate* rendah seperti VMeS.

5.2 Saran

Berdasarkan hasil perancangan dan pengujian terhadap kinerja dari protokol routing AODV-BR, dapat diberikan beberapa saran yang berguna untuk penelitian selanjutnya pada komunikasi VMeS.

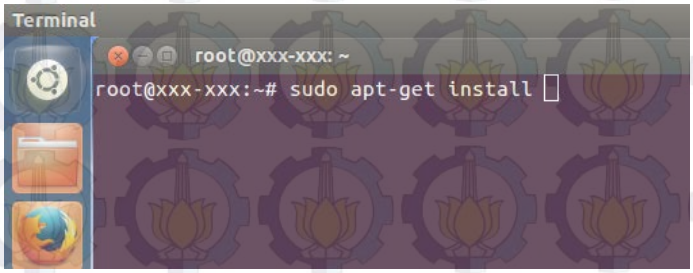
1. Skema routing yang diuji coba pada komunikasi VMeS ini menggunakan protokol MAC pada lapisan datalink IEEE 802.11 dengan skema *Distributed Coordination Function (DFC)* untuk *medium access control*-nya. Perlu dirancang dan diuji coba pada AX.25 dengan mencoba menerapkan mikanisme *Request To Send / Clear to Send (RTS/CTS)* yang ada pada 802.11 terhadap AX.25.
2. Uji coba pada MAC 802.11 sangat efektif, dengan penggunaan *frame* 1 % daru total *frame* pada 802.11, dan AX.25 10%. Untuk penelitian selanjutnya tidak perlu mereduksi paket utama pada protokol AODV-BR dengan alasan tidak dibutuhkannya kapasitas *frame* yang tinggi untuk jalankan protokol AODV-BR pada AX.25.
3. Penelitian terhadap protokol routing AODV-BR ini masih sampai pada simulasi secara tool simulasi dan *testbed*, dan masih sangat perlu dilakukan penelitian secara riil pada perangkat VMeS yang sebenarnya.
4. Untuk analisa kinerja pada protokol *routing* AODV-BR selanjutnya, perlu penambahan parameter *QoS* yang dijadikan parameter ukur agar dapat memberikan kontribusi analisa yang lebih relevan terhadap kinerja dari protokol AODV-BR ini.

LAMPIRAN

1. *Steep by steep* Instalasi NS di Ubuntu 12.04

Dibawah ini adalah langkah-langkah melakukan instalasi dan konfigurasi dari *Network Simulator 2.35* yang digunakan untuk simulasi Protokol AODV-BR. Untuk langkah-langkah detailnya adalah sebagai berikut:

- Pastikan PC atau laptop sudah terinstal Sistem Operasi Linux Ubuntu 12.04 atau versi di atasnya seperti 12.10, 13.04, 13.10, 14.04, 14.10, dan saat ini sudah rilis 15.04. Repository dari OS tersebut sudah terupdate, bisa saja menggunakan official servernya Ubuntu atau pada server mirror lokal yang ada di Indonesia.
- Perangkat harus terhubung dengan jaringan internet pada saat proses instalasi, dikarenakan terdapat beberapa software pendukung yang dilakukan instalasi secara online. Untuk melakukan instalasi secara online digunakan script dasar pada Ubuntu seperti *$sudo apt-get install <nama software>* yang dapat dieksekusi melalui terminal (ctrl + T) pada Ubuntu.



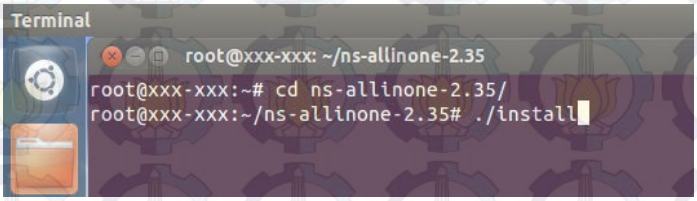
Gambar 7.1 Script dasar Ubuntu untuk melakukan instalasi software

- Sebelum menginstall NS 2.35, lakukan instalasi software atau tool tambahan agar semua fungsi pada NS 2.35 dapat berjalan secara efektif pada Ubuntu 12.04. Sotware yang perlu diinstal sebelum melakukan instalasi NS adalah seperti; *tcl8.5-dev, tk8.5-dev, gcc-4.4, g++-4.4, build-essential, autoconf, automake, perl, xgraph, libxt-dev, lib11-dev,* dan *libxmu-dev.*
- Siapkan software *ns-allinone-2.35.tar.gz* yang bisa diunduh secara gratis pada situs resminya NS yaitu pada alamat www.nsnam.isi.edu.

- Letakkan file software NS pada direktori /Home, kemudian lakukan proses ekstraksi dari paket software NS dengan mengeksekusi perintahnya melalui terminal (ctrl + T). Ketikkan secara beruntun script berikut ini;
    - sudo –s
    - tar -zxvf ns-allinone-2.35.tar.gz

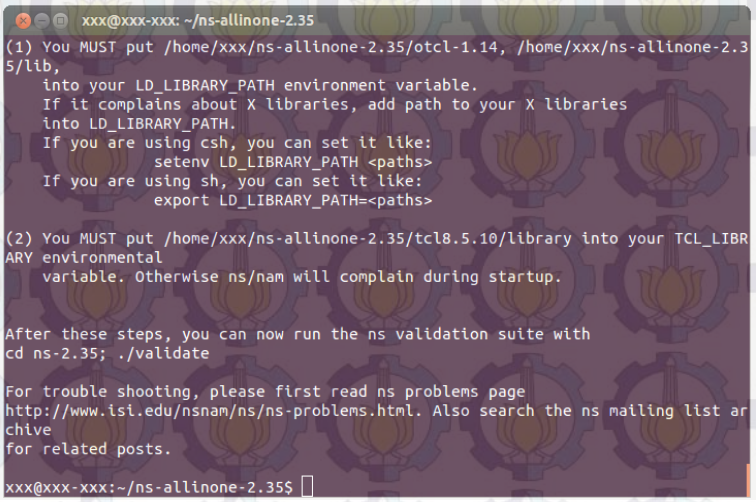
Gambar 7.2 Proses ekstraksi paket NS 2.35

- Setelah proses ekstraksi sudah selesai, arahkan direkrori di terminal pada folder hasil ekstrak dari ns-allinone-2.35, kemudian lakukan instalasi dengan mengeksekusi perintah berikut di terminal.
    - cd ns-allinone-2.35
    - ./install



Gambar 7.3 Proses instalasi NS 2.35

- Apabila proses instalasi tidak terjadi error, maka pada akhir *log* dari proses instalasi yang muncul diterminal akan memberikan informasi rekomendasi untuk mengubah variable LD_LIBRARY_PATH. Ini bertujuan untuk mengarahkan software atau tool yang tadinya diinstall sebelum melakukan instalasi NS, akan disingkronkan dan diarahkan ke direktori dimana NS terinstall. Untuk melakukan perubahan tersebut, dapat dilakukan dengan cara masuk ke folder /HOME, kemudian tekan tombol ctrl+h untuk membuka file yang disembunyikan oleh sistem operasi pada direktori tersebut. File yang disembunyikan ini diperlukan untuk mengubah alamat dari direktori yang direkomendasikan oleh NS.



Gambar 7.4 *Log* akhir dari instalasi NS

- Cari dan buka file dengan nama *.bashrc* yang secara default akan dibuka menggunakan editor gedit pada Ubuntu.
- Kemudian tambahkan script variabel berikut pada akhir dari file *.bashrc* tadi.

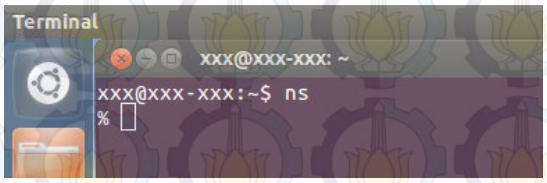
```
PATH=$PATH:/home/xxx/ns-allinone-2.35/bin:/home/xxx/ns-allinone-2.35/tcl8.5.10/unix:/home/xxx/ns-allinone-2.35/tk8.5.10/unix

LD_LIBRARY_PATH=$LD_LIBRARY_PATH: /home/xxx/ns-allinone-2.35/lib

TCL_LIBRARY=$TCL_LIBRARY:/home/xxx/ns-allinone-2.35/tcl8.5.10/library

export PATH
export LD_LIBRARY_PATH
export TCL_LIBRARY
```

- Edit nama user sesuai dengan user yang digunakan. Pada penelitian ini digunakan nama user “xxx”. Setelah selesai melakukan pengeditan, simpanlah hasil edit yang telah dilakukan.
- Langkah selanjutnya adalah melakukan testing terhadap NS, apakah NS sudah berhasil diinstal atau tidak. Langkah ini bisa dilakukan dengan cara ketik NS pada terminal, jika hasilnya “%” berarti NS sudah terinstall dan bisa dijalankan.



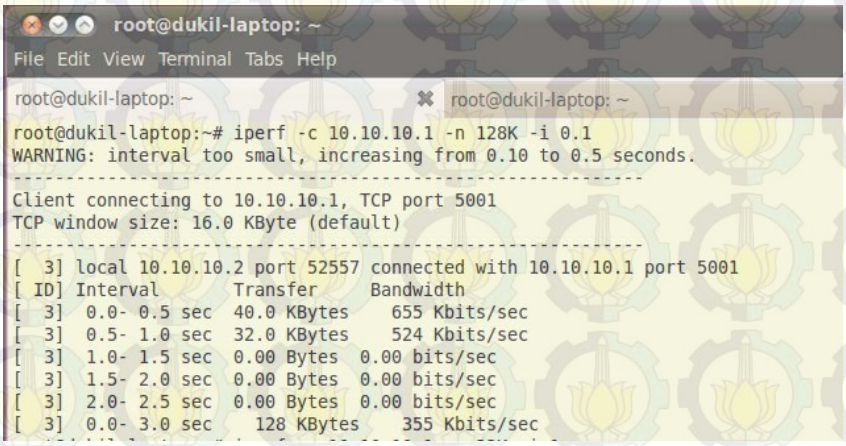
Gambar 7.5 *Testing* NS

- Proses akhir dari instalasi dan konfigurasi dari Network Simulator 2.35 ini adalah melakukan validasi terhadap seluruh perubahan yang dilakukan saat instalasi serta untuk mensinkronkan antar *tool* yang dibutuhkan untuk membangun script pada NS. Langkah ini dapat dilakukan dengan cara mengarahkan direktori terminal kedalam direktori NS *$cd ns-allinone-2.35/ns-2.35* kemudian tekan enter maka secara langsung direktori yang terdapat pada terminal akan mengarah pada folder /ns-2.35. Untuk melakukan proses validasi terhadap NS, ketiklah perintah *$./validate* pada terminal, maka dengan perintah tersebut secara otomatis akan melakukan validasi terhadap semua *tool* yang digunakan oleh NS. Proses validasi membutuhkan waktu yang lumayan lama, bisa diperkirakan 30 menit atau bisa jadi 60 menit sesuai dengan performa PC atau laptop yang digunakan.

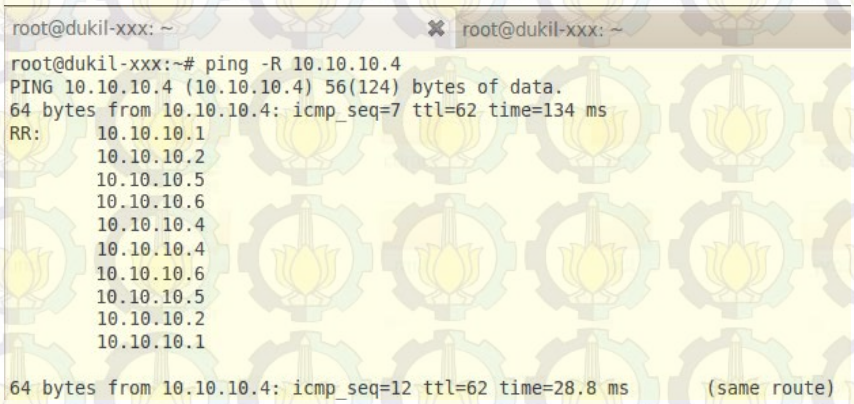
2. Gambar tampilan Iperf pada sisi penerima.

```
^Croot@dukil-xxx:~# iperf -s
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[  4] local 10.10.10.1 port 5001 connected with 10.10.10.2 port 42062
[ ID] Interval       Transfer     Bandwidth
[  4]  0.0- 2.3 sec  32.0 KBytes    114 Kbits/sec
[  5] local 10.10.10.1 port 5001 connected with 10.10.10.2 port 45501
[  5]  0.0- 8.8 sec  32.0 KBytes  29.7 Kbits/sec
[  4] local 10.10.10.1 port 5001 connected with 10.10.10.2 port 45502
[  4]  0.0- 1.5 sec  32.0 KBytes    176 Kbits/sec
[  5] local 10.10.10.1 port 5001 connected with 10.10.10.2 port 45503
[  5]  0.0- 2.4 sec  32.0 KBytes    109 Kbits/sec
[  4] local 10.10.10.1 port 5001 connected with 10.10.10.2 port 45504
[  4]  0.0-17.9 sec  32.0 KBytes  14.7 Kbits/sec
[  5] local 10.10.10.1 port 5001 connected with 10.10.10.2 port 45505
[  5]  0.0- 1.8 sec  32.0 KBytes    144 Kbits/sec
[  4] local 10.10.10.1 port 5001 connected with 10.10.10.2 port 45506
[  4]  0.0- 1.3 sec  32.0 KBytes    199 Kbits/sec
[  5] local 10.10.10.1 port 5001 connected with 10.10.10.2 port 45507
[  5]  0.0- 3.5 sec  32.0 KBytes  75.9 Kbits/sec
[  4] local 10.10.10.1 port 5001 connected with 10.10.10.2 port 45508
```

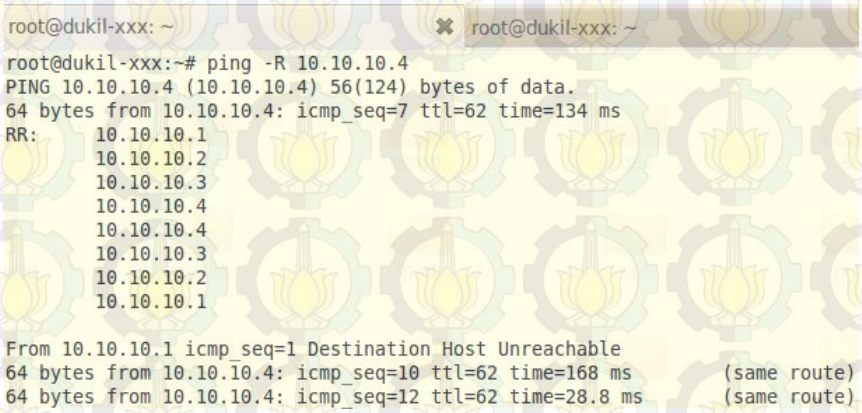
3. **Gambar tampilan Iperf pada sisi pengirim**



4. **Ping –R untuk 4 hop alternatif *route***



5. **Ping –R untuk 4 hop *primary route***



6. Script TCL Topologi VMeS

```
# Simulasi 4-5 node Pada Protokol Routing AODV-BR

# Define options
set val(chan)     Channel/WirelessChannel    ;# Type Chanel
set val(prop)     Propagation/TwoRayGround   ;# Model radio-propagation
set val(netif)    Phy/WirelessPhy            ;# Jenis Interface Jaringan
set val(mac)      Mac/802_11                 ;# Protokol MAC / Datalink
set val(ifq)      Queue/DropTail/PriQueue    ;# Type Interface Antrian
set val(ll)       LL                         ;# Type Layer Link
set val(ant)      Antenna/OmniAntenna        ;# Model Antena
set val(ifqlen)   30                         ;# Max. Paket pada ifq
set val(nn)       5                          ;# Jumlah Mobile Node
set val(rp)       AODV                       ;# Jenis Protokol Routing
set val(x)        500                        ;# Dimensi X pada Pemetaan
set val(y)        450                        ;# Dimensi Y pada Pemetaan
set val(stop)    500                ;# Batas Waktu Simulasi Selesai (ms)

      set ns           [new Simulator]
      set tracefd      [open dukil.tr w]
      set windowVsTime2 [open dukil.tr w]
      set namtrace     [open dukil.nam w]

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

      # Mengatur Pemetaan Objek
      set topo       [new Topography]

      $topo load_flatgrid $val(x) $val(y)

      create-god $val(nn)

      #
      #  membuat mobile node nn dan melampirkan pada saluran
      #

      # Konfigurasi Node
            $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) \
                  -channelType $val(chan) \
                  -topoInstance $topo \
                  -agentTrace ON \
                  -routerTrace ON \
                  -macTrace OFF \
                  -movementTrace ON

          for {set i 0} {$i < $val(nn) } { incr i } {
            set node_($i) [$ns node]
```

```
                $node_($i) set X_ [ expr 10+round(rand()*490) ]
                $node_($i) set Y_ [ expr 10+round(rand()*390) ]
                $node_($i) set Z_ 0.0
            }

            for {set i 0} {$i < $val(nn) } { incr i } {
                $ns at [ expr 1+round(rand()*60) ] "$node_($i) setdest [
expr  10+round(rand()*490)  ]  [  expr  10+round(rand()*390)  ]  [  expr
2+round(rand()*15) ]"
            }

        #Konfigurasi Posisi Awal node mobile
        $node_(0) set X_ 10.0
        $node_(0) set Y_ 10.0
        $node_(0) set Z_ 0.0


        $node_(1) set X_ 75.0
        $node_(1) set Y_ 75.0
        $node_(1) set Z_ 0.0

        $node_(2) set X_ 150.0
        $node_(2) set Y_ 150.0
        $node_(2) set Z_ 0.0

        $node_(3) set X_ 225.0
        $node_(3) set Y_ 225.0
        $node_(3) set Z_ 0.0

        $node_(4) set X_ 300.0
        $node_(4) set Y_ 300.0
        $node_(4) set Z_ 0.0


        # Arah pergerakan Node
        $ns at 10.0 "$node_(0) setdest 10.0 10.0 0.0"
        $ns at 12.0 "$node_(1) setdest 100.0 90.0 3.0"
        $ns at 12.0 "$node_(2) setdest 125.0 140.0 3.0"
        $ns at 12.0 "$node_(3) setdest 260.0 240.0 3.0"
        $ns at 15.0 "$node_(4) setdest 300.0 300.0 10.0"
        #$ns at 60.0 "$node_(5) setdest 150.0 70.0 2.0"
        #$ns at 90.0 "$node_(6) setdest 380.0 150.0 8.0"
        #$ns at 42.0 "$node_(7) setdest 200.0 100.0 15.0"
        # $ns at 55.0 "$node_(8) setdest 50.0 275.0 5.0"
        # $ns at 19.0 "$node_(9) setdest 250.0 250.0 7.0"
        # $ns at 90.0 "$node_(10) setdest 150.0 150.0 20.0"

        # Mengatur Koneksi menggunakan agent TCP antar node pengirim dan
penerima
        set tcp [new Agent/TCP/Newreno]
        $tcp set class_ 2
        set sink [new Agent/TCPSink]
        $ns attach-agent $node_(0) $tcp
        $ns attach-agent $node_(4) $sink
        $ns connect $tcp $sink
        set cbr [new Application/Traffic/CBR]
```

```
        $cbr attach-agent $tcp
        $cbr set packetSize_ 128K
        $cbr set random_ false
        $ns at 10.0 "$cbr start"
        $ns at 480.0 "$cbr stop"

        #set tcp [new Agent/TCP/Newreno]
        #$tcp set class_ 2
        #set sink [new Agent/TCPSink]
        #$ns attach-agent $node_(5) $tcp
        #$ns attach-agent $node_(0) $sink
        #$ns connect $tcp $sink
        #set ftp [new Application/FTP]
        #$ftp attach-agent $tcp
        #$ns at 10.0 "$ftp start"

        # Printing the window size
        proc plotWindow {tcpSource file} {
        global ns
        set time 0.1
        set now [$ns now]
        set cwnd [$tcpSource set cwnd_]
        puts $file "$now $cwnd"
        $ns at [expr $now+$time] "plotWindow $tcpSource $file" }
        $ns at 10.1 "plotWindow $tcp $windowVsTime2"

        # Mendefinisikan posisi node pada ANIMASI (nam)
        for {set i 0} {$i < $val(nn)} { incr i } {

        # Memberikan ukuran besar 30 pada node di file nam
        $ns initial_node_pos $node_($i) 30
        }

        # Konfigurasi akhir sumulasi
        for {set i 0} {$i < $val(nn) } { incr i } {
            $ns at $val(stop) "$node_($i) reset";
        }

        # Konfigurasi akhir simulasi dan animasi
        $ns at $val(stop) "$ns nam-end-wireless $val(stop)"
        $ns at $val(stop) "stop"
        $ns at 490 "puts \"end simulation\" ; $ns halt"
        proc stop {} {
            global ns tracefd namtrace
            $ns flush-trace
            close $tracefd
            close $namtrace
            #exec nam dukil.nam &
            #exit 0
        }

$ns run
```

TESIS – TE142599

PERFORMANCE ANALYSIS OF AODV BACKUP ROUTING ON VMES COMMUNICATION SYSTEM AS A COMMUNCATION TOOL OF FISHING VESSEL

MASDUKIL MAKRUF
2213203010

Supervisors
Dr. Ir. Achmad Affandi, DEA
Dr. Istas Pratomo, ST, MT.

MAGISTER PROGRAME
MULTIMEDIA TELECOMMUNICATION
ELECTRICAL ENGINEERING DEPARTMENT
FACULTY OF INDUSTRIAL TECHNOLOGY
SEPULUH NOPEMBER INSTITUTE OF TECHNOLOGY
SURABAYA
2015

ANALISA KINERJA *AODV BACKUP ROUTING* PADA SISTEM KOMUNIKASI VMES SEBAGAI ALAT KOMUNIKASI KAPAL NELAYAN

Nama Mahasiswa : Masdukil Makruf
NRP : 2213203010
Pembimbing I : Dr. Ir. Achmad Affandi, DEA
Pembimbing II : Dr. Istas Pratomo, ST, MT

ABSTRAK

Indonesia sebagian besar wilayahnya terdiri dari wilayah maritim. Wilayah laut yang luas tentunya memiliki potensi yang besar dan dapat dimanfaatkan oleh banyak kalangan khusunya para nelayan. Dibutuhkan sistem telekomunikasi yang memadai untuk memaksimalkan potensi yang ada. Sistem telekomunikasi alternatif yang dikembangkan sebelumnya untuk kapal nelayan dikenal dengan *Vessel Messaging System (VMeS)*. VMeS merupakan sistem komunikasi yang digunakan untuk bertukar informasi antar kapal atau ke *basestation* di darat dengan menggunakan sistem *ad hoc*. *Ad hoc* yang diadopsi pada VMeS adalah *Mobile Ad hoc Network (MANET)* dimana masing-masing *node* (kapal nelayan) dapat bergerak secara acak. MANET membutuhkan protokol *routing* untuk memandu pengiriman paket data. Protokol *routing* yang digunakan adalah AODV. AODV sudah mampu memberikan mekanisme *routing* yang tepat pada komunikasi VMeS, akan tetapi masih sering terjadi *route break* saat terjadi mobilitas node. Untuk mengatasi masalah ini, digunakan protokol AODV *Backup Routing* pada komunikasi VMeS. Pada penelitian ini, dilakukan analisa terhadap kinerja protokol *routing* AODV-BR pada komunikasi VMeS dengan melakukan simulasi pada *Network Simulator* 2.35 dan *testbed* menggunakan protokol *Medium Access Control* 802.11. Dari hasil analisa simulasi menggunakan *Network Simulator 2* didapatkan rata-rata *delay* 1869 ms untuk AODV-BR sedangkan pada AODV 2546 ms pada variasi data 128 KB dengan 4 *hop*. Kemudian hasil simulasi dibandingkan dengan hasil pengujian pada *testbed*, didapatkan rata-rata delay 4916 ms untuk AODV-BR sedangkan AODV 5724 ms pada skenario yang sama.

Kata Kunci: *MANET, AODV, Backup Routing, VMeS*

PERFORMANCE ANALYSIS OF AODV BACKUP ROUTING ON VMES COMMUNICATION SYSTEM AS A COMMUNCATION TOOL OF FISHING VESSEL

Name	: Masdukil Makruf
NRP	: 2213203010
Supervisor	: Dr. Ir. Achmad Affandi, DEA
Co-Supervisor	: Dr. Istas Pratomo, ST, MT

ABSTRACT

Indonesian territory consists largely of maritime territory. Vast sea area certainly has great potential and can be used by many people but especially the fishermen. Alternative telecommunications system needs to maximize the potential. Alternative telecommunications systems previously developed to fishing boats known as Vessel Messaging System (VMeS). VMeS is a communication system that is used to exchange information between ships or to a base station on land by using ad hoc system. Ad hoc VMeS was adopted on Mobile Ad hoc Network (MANET) where each node (fishing boat) can move randomly. MANET requires a routing protocol to guide the delivery of data packets. Routing protocol used is AODV. AODV has been able to provide the appropriate routing mechanisms on communication VMeS, but still common route break occurs when the node mobility. To resolve this problem, use Backup Routing AODV protocol on communication VMeS. In this study, an analysis of the performance of the routing protocols AODV-BR on VMeS communication by performing simulations on Network Simulator 2.35 and testbed using the 802.11 Medium Access Control Protocols. From the analysis of simulation using Network Simulator 2 obtained an average delay of 1869 ms for AODV-BR whereas in AODV 2546 ms at 128 KB of data variation with 4 hops. Then the simulation results compared with the results of testing on testbed, obtained an average delay 4916 ms for AODV-BR AODV 5724 ms while at the same scenario.

Keyword: *MANET, AODV, Back-up Routing, VMeS*

KATA PENGANTAR

Dengan Nama Allah Yang Maha Pengasih lagi Maha Penyayang.

Segala puja dan puji syukur kepada Allah SWT atas segala rahmat dan karunia yang telah dilimpahkan, sehingga penulisan tesis dengan judul :

"ANALISA KINERJA *AODV BACKUP ROUTING* PADA SISTEM KOMUNIKASI VMES SEBAGAI ALAT KOMUNIKASI KAPAL NELAYAN"

dapat diselesaikan dengan baik. Buku tesis ini disusun untuk memenuhi salah satu syarat memperoleh gelar Magister pada Program Studi Teknik Elektro, Bidang Keahlian Telekomunikasi Multimedia, Institut Teknologi Sepuluh Nopember.

Pada kesempatan ini penulis sampaikan terima kasih yang sedalam-dalamnya kepada :

1. Kedua orangtuaku Ayahanda Hanari dan Ibunda Samarni tercinta yang telah mendidik penulis dari kecil hingga menjadi orang sukses.
2. Bapak Dr. Ir. Achmad Affandi, DEA dan Dr. Istas Pratomo, ST., MT. atas bimbingan, kesabaran dan pendorong semangat dalam menyelesaikan tesis ini.
3. Bapak Dr. Ir. Suwadi, MT., selaku dosen wali yang telah membimbing selama perkuliahan.
4. Bapak Dr. Ir. Wirawan, DEA., Bapak Eko Setidjadi, ST., MT., P.hD., dan Ibu Dr. Ir. Titiek Suryani, MT., yang telah sudi menguji dan mengkoreksi isi dan penyusunan buku tesis ini.
5. Bapak dan Ibu dosen S2 dan S1 terima kasih atas bimbingan dan ilmu pengetahuan yang diberikan selama masa perkuliahan.
6. Rekan-rekan S2 dan S1 di lab Jaringan Telekomunikasi B301, lab Pengolahan Sinyal Digital B304, dan lab Antena dan Propagasi 306, terima kasih atas kebaikan dan kerjasamanya dalam penelitian ini.

7. Terimakasih khusus untuk saudara Afredo Agung Randita dan sudara Fanush Sofi Akbar yang telah membantu dan mengarahkan proses penyusunan tesis ini.
8. Terimakasih khusus untuk Sarindani Oktarina yang telah memberikan dukungan penuh baik secara emosi, mental, dan dukungan semangatnya pada saat proses pembuatan tesis ini.

Penulis menyadari bahwa dalam penulisan thesis ini masih jauh dari sempurna, untuk itu demi perbaikan dan penyempurnaan tesis, maka kritik dan saran sangat diharapkan. Besar harapan penulis bahwa buku thesis ini dapat memberikan informasi dan manfaat bagi pembaca pada umumnya dan mahasiswa Jurusan Teknik Elektro pada khususnya.

Surabaya, 20 Juni 2015

Penulis

Masdukil Makruf
2213203010

DAFTAR ISI

DAFTAR TABEL

BAB 3

BAB 4

!!-- Halaman ini sengaja dikosongkan --!!

DAFTAR GAMBAR

BAB 2

BAB 3

BAB 4

RIWAYAT HIDUP

Masdukil Makruf, lahir di Pamekasan, 02 Juni 1990, merupakan putra pertama dari pasangan Bpk Hanari dan Ibu Samarni.

Menyelesaikan pendidikan jenjang S1

Di Universitas Islam Madura Pamekasan

Jurusan Teknik Informatika

Lulus tahun 2012

Terdaftar sebagai mahasiswa Program Pasca Sarjana

Program Strata Dua Tahun 2013 pada Bidang Studi Telekomunikasi Multimedia

di Institut Teknologi Sepuluh Nopember Surabaya

Email: masdukil.makruf13@mhs.ee.its.ac.id

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Negara Kesatuan Republik Indonesia (NKRI) sebagian besar terdiri dari kepulauan yang dikelilingi oleh lautan. Potensi laut Indonesia yang melimpah ruah dapat dimanfaatkan oleh banyak kalangan khususnya para nelayan. Teknologi telekomunikasi dan informasi sangat membantu terhadap kinerja manusia, yang difungsikan sebagai sarana informasi baik untuk kepentingan keamanan maupun lainnya. Pada wilayah darat, melakukan komunikasi jarak jauh sudah didukung oleh banyak infrastruktur telekomunikasi sehingga memudahkan masyarakat untuk merasakan manfaat dari teknologi ini degan mudah. Beda halnya dengan wilayah laut, yang tidak mungkin dapat membangun infrastruktur tetap (lokasi fix).

Minimnya infrastruktur telekomunikasi pada wilayah laut mengakibatkan susahnya para nelayan untuk melakukan koordinasi, mendapatkan atau mengirim informasi disaat terjadi suatu insiden. Sebenarnya sudah ada alat telekomunikasi untuk wilayah maritim yang memanfaatkan layanan satelit, namun hal ini membutuhkan *budget* yang sangat besar sehingga tidak memungkinkan para nelayan tradisional dapat memanfaatkan alat tersebut [1].

Saat ini, sudah ada penelitian di Lab Jaringan Telekomunikasi ITS tentang *Vessel Messagging System (VMeS)* yang didesain secara *lowcost* dengan harapan dapat memberikan solusi atas keterbatasan alat komunikasi bagi nelayan. VMeS merupakan alternatif solusi untuk alat komunikasi kapal nelayan dibawah ukuran 30 GT. VMeS merupakan pengembangan teknologi dari *Vessel Monitoring System (VMS)* yang digunakan untuk memonitor kapal-kapal penangkap ikan untuk mencegah penangkapan ikan secara ilegal. Kebutuhan sistem komunikasi bagi nelayan merupakan suatu hal yang sangat vital. Dengan adanya sistem komunikasi seperti VMeS, akan memberikan kemudahan pada nelayan untuk memberikan dan mendapatkan informasi dari pusat kontrol atau dari nelayan lain yang juga sedang melakukan pelayaran. VMeS dapat dimanfaatkan oleh nelayan untuk memberikan laporan kepada pusat kontrol mengenai situasi dan kondisi lingkungan, atau

dimanfaatkan untuk menerima informasi dari pusat kontrol misalnya mengenai informasi lokasi yang strategis, peringatan, dan batas-batas wilayah [2].

Mobile Ad Hoc Networks (MANETs) merupakan teknologi telekomunikasi yang sudah dikembangkan penelitiannya untuk mengatasi sulitnya infrastruktur pada suatu wilayah terutama pada wilayah laut. MANET juga merupakan salah satu teknologi yang dikembangkan pada VMeS. MANET tidak mebutuhkan infrastruktur terpusat dikarenakan setiap *node*-nya dirancangan sebagai *router* bergerak secara acak yang dilengkapi dengan *transceiver wireless*. Untuk memungkinkan terjadinya komunikasi antar *node* dibutuhkan protokol *routing* untuk memetakan jalur pengiriman paket data [3] [4].

Adhoc On-demand Distance Vector (AODV) merupakan salah satu protokol *routing* reaktif berbasis *on-demand* dari salah satu ketiga protokol reaktif lainnya yaitu DSDV dan DSR. AODV memiliki performasi yang lebih baik dari pada protokol reaktif yang lain berdasarkan konsumsi delay yang lebih sedikit [5]. AODV yang memiliki ciri utama menjaga *timer-based state* pada setiap node sesuai dengan penggunaan tabel routing. Jalur yang tercatat dalam tabel *routing* akan kadaluarsa apabila tidak ada permintaan. AODV memiliki komponen *Route Discovery* dan *Route Maintenance* sebagai proses pembuatan jalur. *Route Discovery* berupa pesan *Route Request (RREQ)* dan *Route Reply (RREP)*, sedangkan *Route Maintenance* berupa pesan *Route update* dan *Route Error (RRER)* [6]. Protokol AODV lebih efektif dibandingkan protokol MANET lainnya dikarenakan AODV sangat menjaga kualitas jalur dengan meng-*update* tabel *routing* secara periodik sehingga cocok untuk diterapkan pada *node* yang memiliki mobilitas tinggi [7].

Pengembangan penelitian terkait protokol routing AODV sudah banyak dilakukan, salah satunya oleh S.J Lee dan M. Gerla dari University of California – Los Angeles yang telah mengembangkan sekema *flooding* pada AODV menjadi AODV *Backup Routing* (AODV-BR) [8], mereka mengemukakan dua alasan atas dikembangkannya AODV menjadi AODV-BR. Pertama, AODV-BR akan menggunakan jalur alternatif untuk mengirimkan paket data jika jalur terputus, yang mana AODV tradisional membuangnya dikarenakan AODV bukan protokol *multipath*. Kedua, ketika terdapat multi jalur alternatif, ia membuat redundansi dan

meningkatkan kualitas jumlah data transmisi. AODV-BR dalam proses pembuatan jalur alternatif memasrahkan sepenuhnya pada proses penyebaran pesan RREP dan tidak dibutuhkan pesan tambahan lainnya. Dengan teknik seperti ini AODV-BR lebih mampu menstabilkan koneksi jalur dibandingkan dengan AODV [9].

Permasalahan ini menjadi urgen pada MANET dikarenakan sering terjadi putusnya jalur *(route break)* akibat mobilitas *node*. Hal ini dipicu oleh topologi jaringan yang dinamis disebabkan *node* yang selalu bergerak sewaktu-waktu dengan arah yang tidak bisa ditentukan. Putusnya jalur akan berpengaruh pada jumlah waktu yang dibutuhkan untuk pengiriman paket, jumlah kesuksesan pengiriman paket, dan kualitas dari protokol *routing* itu sendiri. Peneltian terdahulu, AODV sudah berhasil disimulasikan dan diuji coba pada VMeS. Namun, dikarenakan sering terjadinya *route break* akibat AODV tradisional merupakan protokol *unipath*, maka sangat perlu untuk menerapkan AODV dengan skema perbaikan *Backup Routing* untuk mengatasi permasalahan tersebut sehingga apabila terjadi putusnya jalur akibat mobilitas node, node sumber atau *next hop* tidak perlu membangun jalur kembali melainkan memanfaatkan *route backup* yang sudah terbentuk saat *route discovery* [8].

Pada penelitian ini, akan menganalisa kinerja protokol routing AODV-BR pada komunikasi VMeS dengan melakukan simulasi menggunakan perangkat lunak *Network Simulator 2.35* dan melakukan uji coba kinerja protokol *routing* pada perangkat *testbed* berupa laptop dengan menggunakan protokol *Medium Access Control (MAC)* 802.11 untuk mengetahui efektifitas kinerja protokol *routing* AODV-BR pada lingkungan VMeS. Untuk dapat menganalisa kinerja dari protokol *routing* AODV-BR pada komunikasi VMeS ini, dibutuhkan hasil pengukuran terhadap performa jaringan berdasarkan beberapa parameter *Quality of Service (QOS)* yaitu *average delay* dan *average throughput* agar dapat menganalisa dan mengambil sebuah kesimpulan.

1.2 Rumusan Masalah

Berdasarkan latar belakang diatas dapat dirumuskan permasalahan sebagai berikut:

1. Bagaimana mensimulasikan dan menguji kinerja protokol *routing* AODV-BR pada komunikasi VMeS?
2. Bagaimana analisa perbandingan terhadap kinerja protokol routing AODV tradisional dengan AODV *Backup Routing* pada komunikasi VMeS?
3. Apakah protokol *routing* AODV-BR dapat memberikan kontribusi mekanisme *routing* yang tepat untuk VMeS?
4. Bagaimana mengimplementasikan *routing* AODV-BR pada VMeS?

1.3 Batasan Masalah

Batasan masalah dari penelitian terhadap analisa kinerja protokol routing AODV-BR pada Komunikasi VMeS, adalah sebagai berikut:

1. AODV-BR disimulasikan dan diimplementasikan pada *testbed* dengan menggunakan *Medium Access Control* 802.11.
2. Protokol yang digunakan adalah AODV *Backup Routing*
3. *Software* simulasi yang digunakan adalah *Network Simulator* versi 2.35
4. Implementasi AODV-BR diuji pada *testbed* laptop menggunakan AODV-UU.
5. Membandingkan kinerja AODV-BR dengan AODV.
6. Untuk menguji *backup routing* diasumsikan sebagian node bergerak dinamis dengan kecepatan tertentu tanpa keluar dari *caverage area*, dan beberapa node pada jalur utama bergerak hingga terjadi *route break.*
7. Pengujian unjuk kerja pada testbed dilakukan di daerah pesisir dan laut.
8. Parameter yang diukur untuk analisa kinerja adalah *average delay* dan *average throughput*.

1.4 Tujuan Penelitian

Tujuan yang akan dicapai dari penelitian ini adalah sebagai berikut:

1. Menguji kinerja protokol AODV-BR dengan menambahkan skema *Backup Routing* untuk mengembangkan *routing* AODV yang sesuai dengan kebutuhan sistem.
2. Untuk menganalisa kinerja protokol routing AODV-BR pada VMeS agar teknologi MANET yang ada pada komunikasi VMeS lebih efektif kinerjanya dengan melakukan simulasi dan *testbed* pada laptop terlebih dahulu sebagai acuan pokok rekomendasi untuk pengembangan protokol pada penelitian VMeS selanjutnya.
3. Dengan adanya pengembangan skema routing pada protokol routing MANET di VMeS ini dapat memberikan kontribusi terhadap sebagian pengembangan teknologi VMeS yang masih membutuhkan pengembangan lebih besar.

1.5 Manfaat Penelitian

Dari usulan penelitian tesis ini diharapkan dapat memberikan kontribusi keilmuan bagi pembaca pada umumnya, bagi penulis pada khususnya mengenai kinerja protokol *routing* AODV *Backup Routing* pada VMeS serta berharap dapat dijadikan bahan acuan untuk penelitian selanjutnya dalam tahapan-tahapan atau proses pengembangan protokol *routing* pada sistem komunikasi VMeS.

1.6 Metodologi

Metodologi penelitian dalam menganalisa kinerja protokol *routing* AODV-BR pada Komunikasi VMeS ini meliputi tahapan-tahapan berikut:

1. Perumusan Masalah

   Langkah awal yang dilakukan untuk memulai penelitian pada Tesis ini adalah studi literatur. Dari hasil studi litertur, dapat dirumuskan beberapa permasalah sebagai materi penelitian pada tesis ini.

2. Perancangan Sistem

   Pada tahap ini dilakukan perancangan untuk menganalisa kinerja protokol *routing* AODV *Backup Routing* pada sistem komunikasi VMeS menggunakan *software Network Simulator* 2 dan merancang *testbed* protokol *routing* menggunakan laptop untuk melakukan uji coba secara riil.

3. Implementasi Sistem

   Dari hasil perancangan sistem tersebut, dibuat implementasi jaringan *ad hoc* pada VMeS diantaranya mengimplementasikan secara simulasi dan *testbed* menggunakan laptop.

4. Pengujian dan Analisa

   Dari hasil implementasi, akan dilakukan uji coba kinerja dengan mengukur beberapa parameter QoS seperti *delay* dan *throghput* berdasarkan banyak variasi *hop* yang sudah ditentukan. Kemudian, hasil pengukuran akan dianalisa untuk mendapatkan sebuah kesimpulan.

5. Kesimpulan dan Saran

   Berisi kesimpulan tentang hasil yang sudah didapatkan, dan memberikan saran untuk penelitian selanjutnya.

1.7 Relevansi

Hasil yang diperoleh dari tesis ini diharapkan dapat memberikan manfaat sebagai berikut:

1. Menghasilkan tulisan ilmiah mengenai Analisa Kinerja Protokol AODV-BR pada komunikasi VMeS.
2. Memberikan rekomendasi dalam memilih mekanisme *routing* yang tepat untuk penelitian selanjutnya.
3. Memberikan ulasan mengenai penggunaan AODV pada *Medium Access Control (MAC)* dengan protokol AX.25.

BAB 2
TINJAUAN PUSTAKA

2.1 Sistem Komunikasi VMeS

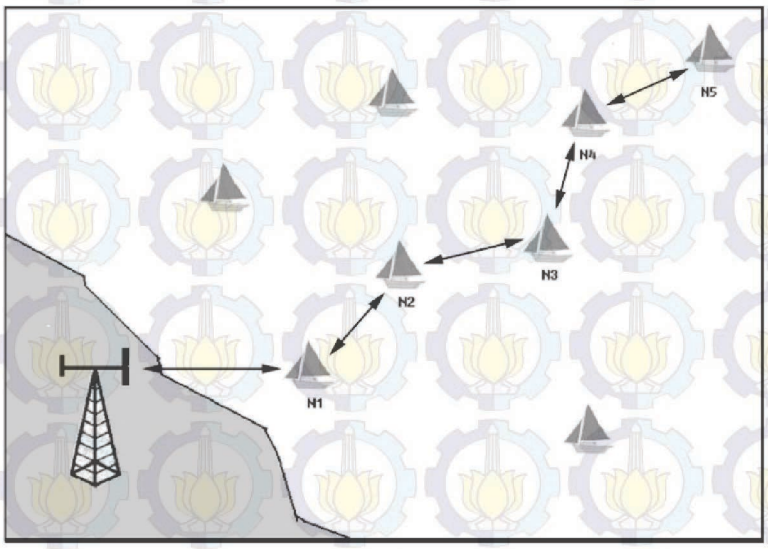
Sistem komunikasi VMeS *(Vessel Messaging System)* merupakan salah satu dari projek penelitian yang saat ini sedang dikembangkan di Laboratorium Jaringan Telekomonukasi Elektro ITS. VMeS merupakan sistem komunikasi laut berbasis *wireless* yang terdiri dari satu *node gateway* dan node tersebar secara *mobile* dan acak. VMeS dikembangkan dari sistem komunikasi setelit yang disebut dengan VMS *(Vessel Monitoring System)*. VMS dirancang khusus untuk memonitoring posisi kapal pada wilayah laut, namun sistem ini dianggap tidak efisien apabila digunakan sebagai alat komunikasi pada kapal nelayan, dikarenakan sistem ini membutuhkan biaya yang sangat besar dan konsumsi daya yang juga besar. Oleh sebab itu, timbulah inisiatif untuk membuat sebuah sistem komunikasi alternatif untuk kapal nelayan yang murah dan terjangkau [2].

VMeS dirancang dengan menggunakan gelombang radio pada frekuensi VHF [10] untuk menyampaikan informasi secara dua arah. Informasi yang dikirimkan dapat berupa data kapal, data kondisi lingkungan di sekitar kapal, dan data peringatan tanda bahaya. Dengan demikian para nelayan bisa memberikan informasi tentang kejadian-kejadian di lingkungan sekitar kapal yang sedang mereka gunakan, misalnya terjadi kecelakaan serta adanya tindak pencurian ikan oleh kapal nelayan asing. Informasi yang diberikan oleh nelayan tersebut bisa diteruskan ke pihak yang berwenang sehingga apabila terjadi hal-hal yang tidak diinginkan dapat cepat direspon dan ditanggulangi.

Banyak hal yang sudah dikembangkan dalam VMeS, baik dari segi sistem komunikasi, teknik pengiriman data, bahkan perangkat yang digunakanpun disesain agar semakin murah. Sistem komunikasi yang digunakan pada VMeS adalah sistem *ad hoc*, dikarena sistem *ad hoc* mampu melakukan komunikasi dengan memanfaatkan jangkauan perangkat lain sehingga data yang hendak dikirimkan dapat sampai pada tujuan [1].

Alasan dipilihanya *ad hoc* sebagai sistem komunikasi yang dimanfaatkan oleh sistem VMeS ialah dikarenakan VMeS dirancang khusus untuk diterapkan pada suatu wilayah yang tidak dapat dibangun infrastruktur komunikasi sama sekali yaitu pada wilayah laut. Tidak ada pilihan sistem komunikasi lain yang dapat digunakan selain menggunakan sistem *ad hoc*, dikarekan jika menggunakan layanan satelit, perangkatnya dapat dibilang mahal apalagi untuk kalangan nelayan. Semua sistem yang diterapkan dalam sistem *ad hoc* di VMeS adalah sama dengan sistem *ad hoc* yang ada pada sistem komunikasi lain, misalnya dari segi teknik *routing* dari sistem ini. Semua teknik *routing* yang digunakan dalam VMeS juga memiliki peran penting dalam melakukan komunikasi, seperti pemetaan jalur pengiriman paket data berupa informasi dan lainnya [11].

Node yang tersebar dalam sistem VMeS berupa kapal nelayan sama halnya dengan *node* yang bergerak dalam sistem *ad hoc*. Pergerakan yang terjadi secara acak merupakan salah satu teknik dalam *ad hoc* yang memiliki *node* bergerak, hal tersebut biasa disebut dengan *Mobile Ad Hoc Network (MANET)*. MANET memiliki topologi jaringan dinamis yang selalu berubah tanpa dapat ditentukan. MANET diterapkan pada VMeS, disesuiakan dengan kondisi lingkungan pada VMeS yang memiliki node bergerak dan susahnya mendirikan infrastruktur fix dan terpusat pada wilayah laut [4].



Gambar 2.1 Ilustrasi VMeS dengan Teknologi *Ad Hoc* [1]

2.2 *Mobile Adhoc Network (MANET)*

Jaringan *wireless* terdiri dari dua model yaitu *fixed wireless*, dan *mobile wireless*. Jaringan *fixed wireless* tidak mendukung *mobility* karena infrastrukturnya terbangun dengan paten pada suatu titik lokasi, dan kebanyakan jaringan ini adalah *point to point*. Lain halnya dengan *mobile wireless* yang dapat bergerak namun tetap bisa berkomunikasi meskipun dengan topologi yang dinamis dan sangat dibutuhkan oleh user bergerak. Jaringan *mobile* terbagi menjadi dua kategori utama, yaitu jaringan yang memiliki infrastruktur dan jaringan yang tidak memiliki infrastruktur.

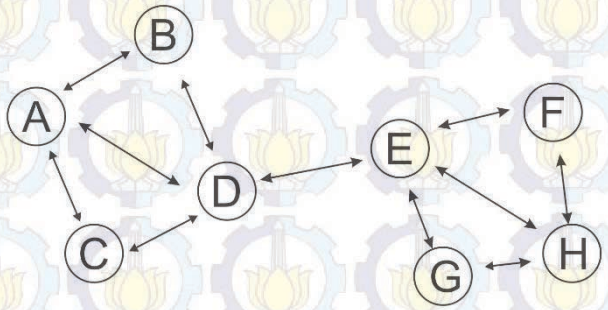
Ad hoc Network merupakan desentralisasi dari jaringan nirkabel, disebut *ad hoc network* karena tidak bergantungnya jaringan ini terhadap adanya infrastruktur, seperti *router* dalam jaringan kabel ataupun *Access Point* pada jaringan nirkabel. Dalam *Ad hoc network*, setiap node memiliki tugas me-*routing* data kepada *node* lain. Jadi, penentuan *node* mana yang mengirimkan data dibuat secara dinamis berdasarkan konektivitas dari jaringan itu sendiri. Sifat desentralisasi, protokol *routing* dinamis, dan mudah untuk diterapkan menjadikan *ad hoc network* cocok untuk diimplementasikan disaat jaringan terpusat tidak dapat digunakan (seperti ditengah laut atau situasi darurat seperti bencana alam atau konflik militer). Dalam beberapa tahun terakhir, banyak pakar jaringan mengalihkan perhatian mereka dari jaringan terpusat seperti jaringan internet dan telepon seluler dan berpindah ke *ad-hoc network* [3].

Mobile ad hoc Network (MANET) adalah jenis sistem komunikasi *ad hoc* yang mana setiap *node*-nya bebas bergerak secara independen dan acak serta topologi jaringannya terus berubah-ubah sesuai dengan kondisi mobilitas. Setiap perangkat *(node)* harus meneruskan paket yang dikirimkan sesuai tujuan alamat kendatipun tidak ada kaitannya dengan *node* itu sendiri, sebab itulah setiap *node* pada jaringan *ad hoc* atau *mobile ad hoc* disebut sebagai *router* dikarenakan memiliki tugas untuk me-*routing* paket yang dikirim oleh *sender* ke *destination*. Tantangan utama dalam membangun MANET adalah melengkapi setiap perangkat untuk terus menjaga informasi yang diperlukan untuk merutekan lalu lintas data secara benar. Jaringan pada MANET dapat beroperasi sendiri atau dapat dihubungkan ke jaringan internet yang lebih besar. Salah satu konsep dasar yang

ada pada MANET merupakan konsep dasar jaringan ad hoc yang memliki tabel *routing* dan berada di atas layer *Data Link* tepatnya pada layer *Network*.

Dalam jaringan *mobile ad hoc*, tidak terdapat *base-station*, dan tidak ada pengawas yang memantau kinerja jaringan secara keseluruhan. Node yang digunakan di jaringan *mobile ad hoc* akan aktif dan mencoba untuk menentukan berapa banyak *node* aktif lainnya yang berada dalam jangkauan transmisi. Secara bersama-sama, *node* kemudian mengumpulkan informasi apapun yang dibutuhkan untuk melakukan tugas secara kolektif.

Dalam pengertian lain, jaringan *ad hoc* adalah jaringan bersifat sementara tanpa bergantung pada infrastruktur yang ada dan bersifat independen. Jaringan *Ad Hoc* adalah jaringan *wireless* yang terdiri dari kumpulan *mobile node (mobile station)* yang bersifat dinamik dan spontan, dapat diaplikasikan di mana pun tanpa menggunakan jaringan infrastruktur (seluler ataupun PSTN) yang telah ada. Contoh *mobile* node adalah *notebook*, PDA dan ponsel. Jaringan ad hoc disebut juga dengan s*pontaneous network* atau disebut *MANET (Mobile Ad hoc NETwork)* [12].



Gambar 2.2 Struktur Dasar Jaringan *Ad Hoc* [12]

Saat ini *mobile ad hoc network* sudah banyak diterapkan dalam segala jenis aspek kebutuhan telekomunikasi. Jaringannya yang fleksibel menjadikan jaringan *ad hoc* banyak diminati untuk dikembangkan. Berikut ini adalah contoh-contoh aspek yang sudah menggunakan dan menerapkan sistem *mobile ad hoc network* di lapangan:

- Operasi militer, seperti yang telah diujicobakan kawasan pertempuran di Sudan. Dengan jaringan *ad hoc*, mempermudah untuk akses informasi antar personil militer.

- Komersial, jaringan *ad hoc* dapat digunakan pada situasi *emergency* atau upaya penyelamatan *(rescue operation)*, seperti banjir atau gempa bumi dan *entertainment* seperti acara *live music*.
- Jaringan yang cepat tersedia dengan menggunakan *notebook* untuk menyebarkan dan berbagi informasi di antara user seperti dalam konferensi atau ruang kuliah.
- *Personal Area Network*, untuk jarak pendek *(short distance)* lebih kurang 10 m, *ad hoc network* secara mudah berkomunikasi antar bermacam peralatan (seperti PDA, laptop dan telepon seluler) dengan laju data yang rendah.
- VMeS, digunakan untuk komunikasi antar node yang tersebar di wilayah laut yang membutuhkan sistem *ad hoc* untuk dapat berkomunikasi dengan pusat kontrol di darat, atau sebaliknya dikarenakan hanya dengan sistem *ad hoc*-lah VMeS dapat bekerja pada wilayah yang sama sekali tidak ada infrastruktur telekomunikasi.

Setiap sistem telekomunikasi memiliki kelebihan dan kekurangan tersendiri dalam penerapannya, masing-masing sistem berfokus pada kinerjanya masing-masing. Demikian juga dengan sistem jaringan *mobile ad hoc* yang juga memiliki kelebihan dan kekurangan dalam setiap penerapannya. Banyak faktor yang menyebabkan adanya kelebihan dan kekurangan dari pada sistem telekomunikasi seperti MANET, misalnya seperti kondisi lingkungan yang tidak dapat diprediksi menjadikan sistem telekomunikasi dipaksa menyesuaikan dengan fungsinya masing-masing. Adapun kelebihan dalam penerapan jaringan *mobile ad hoc* adalah sebagai berikut:

1. Tidak memerlukan dukungan *backbone* infrastruktur sehingga mudah diimplementasikan dan sangat berguna ketika infrastruktur tidak ada ataupun tidak berfungsi lagi.
2. *Mobile node* yang selalu bergerak, dapat mengakses informasi secara *real time* ketika berhubungan dengan *mobile node* lain, sehingga pertukaran data dan pengambilan keputusan dapat segera dilaksanakan.
3. Fleksibel terhadap suatu keperluan tertentu karena jaringan ini memang bersifat sementara.

4. Dapat direkonfigurasi dalam beragam topologi baik untuk jumlah user kecil hingga banyak sesuai dengan aplikasi dan instalasi *(scalability)*.

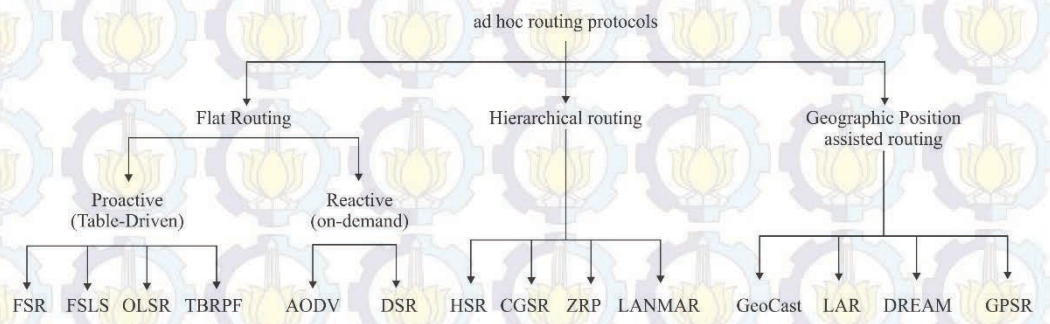
Jika kelebihan pada jaringan *mobile ad hoc* dapat dikemukakan, sudah tentu jaringan *mobile ad hoc* memiliki kekurangan atau kerugian jika menggunakan jaringan *mobile ad hoc*.

1. *Packet loss* (rugi-rugi paket) akan terjadi bila transmisi mengalami kesalahan *(error)*.
2. Seringkali terjadi *disconnection*, karena tidak selalu berada dalam area cakupan.
3. *Bandwidth* komunikasi yang terbatas.
4. *Lifetime* daya yang singkat.
5. Kapasitas kemampuan jangkauan *mobile node* yang terbatas dan bervariasi [3].

2.3 Protokol Routing dalam MANET

Protokol *routing* dalam teknologi *Mobile ad Hoc Network (MANET)* memiliki peran penting dalam melakukan komunikasi antar node. Protokol *routing* yang memiliki peran pada lapisan 3 *(network layer)* dalam lapisan OSI ini sudah menjadi tulang punggung dalam suksesnya melakukan komunikasi pada lingkungan *ad hoc*. Protokol *routing* difungsikan sebagai standar komunikasi atau aturan-aturan yang harus berlaku dengan teknik tertentu. Setiap protokol *routing* memiliki algoritma *routing* tersendiri sesuai standar yang dibuat oleh penemu dari teknik *routing* tertentu.

Telah banyak protokol *routing* yang dikembangkan pada komunkasi *ad hoc* ini. Mereka mengelompokkan protokol *routing ad hoc* menjadi tiga bagian utama, yaitu; *flat rooting protocols, Hierarchical routing protocols,* dan *Location-based routing protocols*.



Gambar 2.3 Pengelompokan Protokol Routing dalam *Ad Hoc* [13]

Flat routing protocols didalamnya terdapat dua kategori, yaitu proaktif dan reaktif. Perbedaan dasar dari ketiga jenis kategori ini adalah; proaktif berbasis *table-driven* sedangkan reaktif berbasis *on-demand*. Kedua kategori tersebut memiliki bagian-bagian jenis protokol *routing ad hoc* yang tentunya memiliki algoritma masing-masing [13]. Berikut ini adalah pembagian protokol *routing* dari *flat routing protocols*:

- Proaktif
  - ✓ *FSR (Fisheye State Routing)*
  - ✓ *FSLS (Fuzzy Sighted Link State)*
  - ✓ *OLSR (Optimized Link State Routing)*
  - ✓ *TBRPF (Topology Broadcast Based on Reverse Path Forwarding)*
- Reaktif
  - ✓ *AODV (Ad Hoc On-demand Distance Vector routing)*
  - ✓ *DSR (Dynamic Source Routing)*

Hierarchical routing protocols terbentuk saat ukuran jaringan semakin meningkat. Flat *routing* tidak mampu dikarekan terjadi *overhead*. *Hierarchical routing protocols* akan membagi jaringan menjadi beberapa grup dan memberikan fungsi yang berbeda antara grup yang di dalam jaringan dan grup di luar jaringan. *Hierarchical routing protocols* juga memiliki beberapa bagian protokol di dalamnya yaitu; *CGSR (Clusterhead-Gateway Switch Routing), HSR (Hierarchical State Routing), ZRP (Zone Routing Protocol), dan LANMAR (Landmark Ad Hoc Routing Protocol)*.

Location-based routing protocols merupakan bagian terakhir dari kategori utama pembagian protokol routing dalam jaringan *ah hoc*. Protokol yang dikelompokkan pada *Location-based routing protocols* ini mengandalkan *basic* informasi lokasi pada setiap *node*. Hal tersebut bisa diperoleh dengan menggunakan sensor dan GPS. Protokol ini memanfaatkan *geographical forwarding* untuk mengirim paket data. Adapun protokol routing yang terkelompok dalam kategori ini adalah; *LAR (Location-Aided Routing), DREAM (Distance Routing Effect Algorithm for Mobility), GPSR (Greedy Perimeter Stateless Routing)* dan *GLS (Grid Location Service* [3].

2.4 *Adhoc on-Demand Distance Vector (AODV)*

2.4.1 Definisi AODV

Ad hoc On-demand Distance Vector (AODV) merupakan *routing* protokol reaktif berbasis *on-demand* yang dirancang untuk jaringan *mobile ad hoc*. Algoritma yang dikembangkan dalam protokol routing AODV oleh C. Perkins dan M. B. Royer ini merupakan implementasi dari *Distributed Bellman-Ford (distance vector)* yang digunakan oleh smua protokol *routing Distance Vector (DV)* pada *mobile ad hoc network* termasuk AODV ini [6].

AODV menciptakan suatu jalur dengan menggunakan *Route Request (RREQ)* dan *Route Reply (RREP)*. Ketika *source node* menginginkan suatu jalur menuju *destination node* tetapi belum mempunyai jalur yang benar, maka *source node* akan menginisialisasi *route discovery process* untuk menemukan jalur ke *destination node*. *Source node* akan mem-*broadcast* paket RREQ menuju node tetangganya *(neighbor)*. RREQ paket berisi *source address*, *destination address*, *hop counter*, *source* dan *destination sequence number*, dan *broadcast* ID. Nilai *Broadcast* ID akan bertambah satu setiap suatu *source node* mengirimkan RREQ yang baru dan digunakan sebagai identifikasi sebuah paket RREQ [14].

Jika *node* yang menerima RREQ memiliki informasi jalur menuju *destination node*, maka *node* tersebut akan mengirim paket RREP kembali menuju *source node*. Tetapi jika tidak mengetahui maka *node* tersebut akan mem-*broadcast*

ulang RREQ ke *node* tetangganya setelah menambahkan nilai *hop counter*. *Node* yang menerima RREQ dengan nilai *source address* dan *broadcast* ID yang sama dengan RREQ yang diterima sebelumnya akan membuang RREQ tersebut. *Source sequence number* digunakan oleh suatu *node* untuk memelihara informasi yang valid mengenai *reverse path* (jalur balik) menuju ke *source node*. Pada saat RREQ mengalir menuju *node* tujuan yang diinginkan, dia akan menciptakan *reverse path* menuju ke *node*, setiap *node* akan membaca RREQ dan mengidentifikasi alamat dari *node* tetangga yang mengirim RREQ tersebut.

Ketika *destination node* atau *node* yang memiliki informasi jalur menuju *destination* menerima RREQ maka *node* tersebut akan membandingkan nilai *destination sequence number* yang ia miliki dengan nilai destination *sequence number* yang ada di RREQ. Jika nilai *destination sequence number* yang ada di RREQ lebih besar dari nilai yang dimiliki oleh *node* maka paket RREQ tersebut akan di-*broadcast* kembali ke *node* tetangganya, sebaliknya jika nilai *destination sequence number* yang ada di *node* lebih besar atau sama dengan nilai yang ada di RREQ maka *node* tersebut akan mengirim *route reply (RREP)* menuju *source node* dengan menggunakan *reverse path* yang telah dibentuk oleh RREQ. *Intermediate* node yang menerima RREP akan meng-*update* informasi *timeout* (masa aktif rute) jalur yang telah diciptakan. Informasi rute *source* ke *destination* akan dihapus apabila waktu pada tabel *lifetime*-nya habis [7] [14].

Selama jalur yang ditentukan tetap aktif, jalur tetap akan terus dipertahankan. Sebuah rute dianggap aktif apabila masih ada paket data secara periodik terkirim dari *node* sumber ke tujuan disepanjang jalur yang aktif ini. Setelah sumber berhenti mengirimkan paket data, akan terjadi *link timeout* dan akhirnya akan dihapus paket tersebut dari tabel *routing node* perantara. Jika suatu link terputus sementara rute aktif, node yang mengalami kerusakan akan mengirim pesan *Route Error (RERR)* ke node sumber untuk menginformasikan bahwa jalur ini tidak terjangkau seperti yang diilustrasikan pada Gambar 2.6. Setelah menerima pesan RERR, apabila ternyata node sumber masih tetap membutuhkan rute ini, maka *node* sumber akan mengulang kembali pembuatan jalur selama paket yang

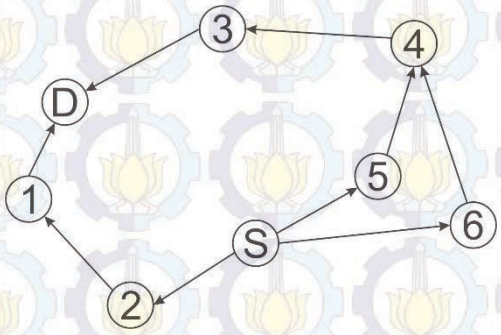
hendak dikirim masih belum kadaluarsa. Untuk prosedur dalam jalur *multicast* dan *broadcast* juga ditetapkan dengan cara yang sama [6].

Dalam algoritma *routing* AODV, setiap *node* diharuskan menjaga tabel *routing* pada masing-masing *node*. Adapun judul dari *field* pada tabel *routing* AODV adalah sebagai berikut:

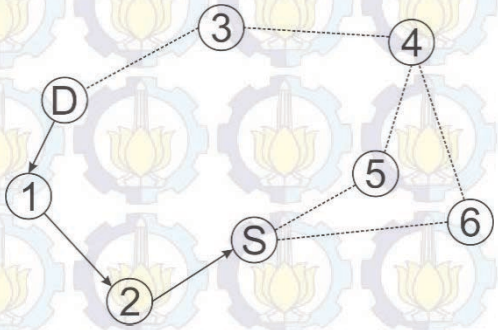
1. *Destination IP Address* berisi alamat IP dari *node* tujuan yang digunakan untuk menentukan jalur.
2. *Destination Sequence Number* bekerjasama untuk menentukan rute
3. *Next Hop* atau "loncatan" berikutnya, bisa berupa tujuan atau node tengah, *field* ini dirancang untuk meneruskan paket ke node tujuan.
4. *Hop Count* Jumlah hop dari alamat IP sumber sampai ke alamat IP tujuan.
5. *Lifetime* waktu dalam milidetik yang digunakan untuk node menerima RREP.
6. *Routing Flags* merupakan status dari sebuah jalur, *up* (valid), *down*(tidak valid) atau sedang diperbaiki.

AODV menggunakan tabel routing dengan satu entri untuk setiap tujuan, dan tanpa menggunakan *routing* sumber. AODV mempercayakan semua pemetaan jalur pada tabel routing pada setiap node-nya untuk menyebarkan *Route Reply (RREP)* kembali ke sumber dan secara sekuensial akan mengarahkan paket data menuju ketujuan. AODV juga menggunakan *sequence number* untuk menjaga setiap tujuan agar didapat informasi *routing* yang terbaru dan untuk menghindari *routing loops*. Semua paket yang diarahkan membawa *sequence number* ini ke *node* tujuan.

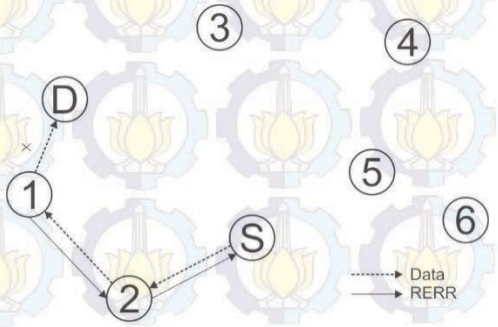
Penemuan jalur *(path discovery)* atau *route discovery* diinisialisasi dengan menyebarkan *Route Reply (RREP)*, seperti terlihat pada Gambar 2.5. Ketika RREP menjelajahi node, ia akan secara otomatis membangun jalur. Jika sebuah *node* menerima RREP, maka *node* tersebut akan mengirimkan RREP lagi ke node sumber berdasarkan *destination sequence number*. Pada proses ini, yang akan dicek pertama kali oleh setiap *node* saat menerima pesan RREP adalah *destination sequence number* ini.



Gambar 2.4 Pengiriman Pesan RREQ Saat Proses *Route Discovery* [7]



Gambar 2.5 Jalur Terdekat yang Dilintasi Oleh Pesan *route reply* (RREP) [7]



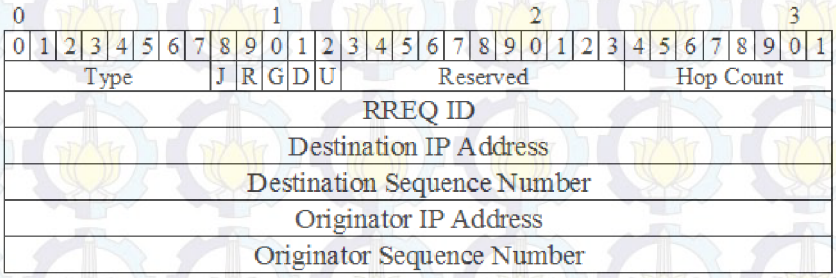
Gambar 2.6 Proses Pengiriman RERR ke *Source* Saat Terjadi *Route Break* [7]

2.4.2 Format Pesan dalam AODV

Protokol routing *Ad hoc On-demand Distance Vector (AODV)* memeliki paket pesan untuk memulai proses setiap segmennya dalam membuat jalur. Paket pesan tersebut antara lain adalah *Route Request (RREQ)*, *Route Reply (RREP)*, dan *Route Error (RERR)*. Setiap pesan memiliki format tersendiri, sesuai dengan standar yang sudah dibuat oleh C. Perkins dan EM. Royer yang tentunya juga sudah disesuaikan dengan kebutuhan *routing* pada AODV itu sendiri [6]. Adapun fotmat-format pesannya berdasarkan standar RFC 3561 adalah sebagai berikut:

2.4.2.1 Format Pesan *RREQ (Route Request)*

Berikut ini adalah format pesan RREQ yang digunakan saat proses *route discovery*.



Gambar 2.7 Format Pesan *RREQ (Route Request)* [14]

Format pesan RREQ yang diilustrasikan seperti Gambar 2.7, berikut adalah penjelasan dari format pesan tersebut:

Type : 1

J : *Join flag*, disediakan untuk *multicast*

R : *Repair flag*, disediakan untuk *multicast*

G : *Gratuitous RREQ flag*; menunjukkan apakah RREQ percuma untuk di-*unicast* ke node yang ditentukan dalam IP address tujuan.

D : *Destination only flag*; mengindikasikan bahawa hanya tujuan yang memungkinkan dapat menanggapi RREQ ini

U : *Unknown sequence number*; yang menunjukkan nomor urut tujuan tidak diketahui.

Reserved : Jika yang dikirim bernilai 0; maka pesan RREQ diabaikan pada penerima.

Hop Count : Jumlah *hop* dari IP *originator* ke node yang menangani *request*.

RREQ ID : Adalah sebuah nomor unik yang mengidentifikasi RREQ tertentu dari organisasi IP pada *node-node*.

Destination IP Address adalah alamat IP tujuan yang diinginkan

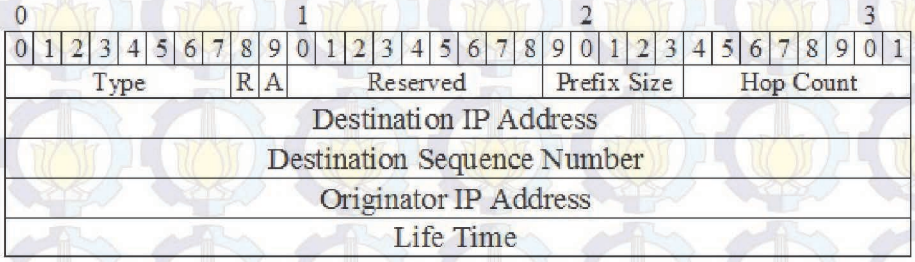
Destination Sequence Number adalah nomor urut terakhir yang diterima oleh *originator* untuk jalur apapun menuju destinasi.

Originator IP Address adalah alamat IP yang berasal dari *node* yang mengirim RREQ

Originator Sequence Number adalah nomor urut *node* yang akan digunakan dari *node* tujuan ke *node* sumber.

2.4.2.2 Format Pesan *RREP (Route Reply)*

Berikut ini adalah format pesan RREP yang digunakan saat proses membentuk jalur dari tujuan ke penerima.



Gambar 2.8 Format Pesan *RREP (Route Reply)* [14]

Format pesan RREP diilustrasikan seperti diatas, berikut adalah penjelasan dari format pesan diatas:

Type : 2

R : *Repair Flag*, disediakan untuk multicast

A : *Acknowledgment required* (pesan ACK)

Reserved : Jika yang dikirim berniali 0; maka diabaikan pada penerima.

Prefix Size : Jika nol, ukuran *Prefix* 5 bit menetapkan bahwa *hop* berikutnya yang ditunjukkan dapat digunakan untuk setiap node dengan awalan *routing* yang sama (seperti yang didefinisikan oleh *PrefixSize*) sebagai tujuan yang diminta

Hop Count : Jumlah *hop* dari alamat IP *originator* ke Alamat IP destinasi. Untuk rute permintaan *multicast* ini menunjukkan jumlah hop keanggota *multicast* yang mengirim RREP

Destination IP Address berisi alamat IP tujuan dan rute yang mana saja yang disediakan

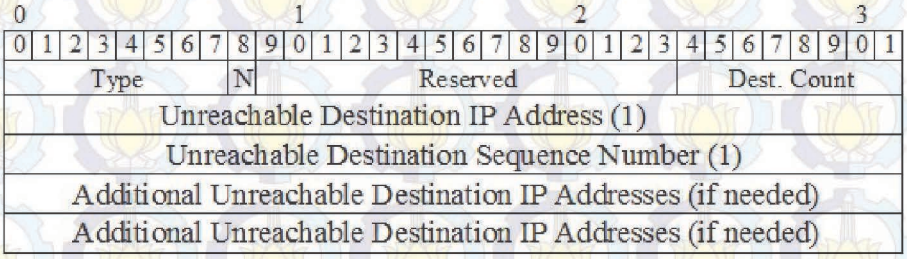
Destination Sequence Number merupakan nomor urut tujuan terkait.

Originator IP address adalah Alamat IP yang berasal dari node yang mengirim RREQ

Life Time adalah masa berlakunya rute yang terhitung dalam milidetik

2.4.2.3 Format Pesan *RERR (Route Error)*

Berikut ini adalah format pesan RRER yang digunakan apabila terjadi *route break* dari *source* ke *destination*.



Gambar 2.9 Format Pesan *RRER (Route Error)* [14]

Format pesan RRER diilustrasikan seperti pada Gambar 2.9, yang berisi bidang-bidang berikut ini:

Type : 3

N : *No delete flag* adalah yang mengatur kapan sebuah *node* akan melakukan perbaikan pada lokal *link*, dan node awal seharusnya tidak menghapus rute.

Reserved : Jika yang dikirim bernilai 0; maka diabaikan pada penerima.

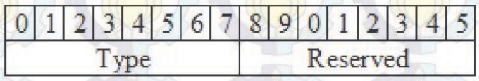
Destination Count adalah jumlah tujuan *unreachable* yang termasuk dalam pesan tersebut; minimal harus 1

Unreachable destination IP address adalah alamat IP tujuan yang tidak terjangkau karena terjadinya *link break (route break)*.

Unreachable Destination Sequence Number adalah nomor urut dalam entri tabel routing untuk tujuan yang tercantum dalam *unreachable IP address* tujuan sebelumnya.

2.4.2.4 Format Pesan *Route Reply Acknowledgment (RREP-ACK)*

RREP-ACK merupakan pesan yang digunakan oleh pengirim dalam routing AODV untuk mengetahui apakah paket yang dikirim sudah diterima oleh tujuan atau belum, ini diperlukan agar node pengirim tidak melakukan pembentukan ulang jalur. Adapun standar format pesan *Route Reply Acknowledgement (RREP-ACK)* adalah sebagai berikut:



Gambar 2.10 Format Pesan *RREP-ACK* [14]

Type : 3

Reserved : Jika yang dikirim bernilai 0; maka diabaikan pada penerima.

2.5 Arsitektur Protokol AODV *Backup Routing*

Penelitian demi penelitian terus dilakukan oleh para peneliti untuk menutupi dan untuk memperbaiki kekurangan yang ada pada AODV tradisional, salah satunya adalah AODV *Backup Routing* atau dikenal dengan AODV-BR yang dikembangkan oleh Lee dan Mario Gerla [8], AODV-BR diciptakan untuk menutupi berbagai macam kekurangan yang ada pada AODV tradisional, kekurangan tersebut meliputi:

- Tidak adanya *multiple path* atau jalur alternatif yang dapat menghandel paket data jika terjadi kerusakan pada jalur utama, sehingga harus dilakukan pembentukan jalur kembali.
- Jika terjadi *route break*, data akan di-*drop* akibat tidak adanya jalur alternatif yang dapat dilalui oleh data sampai jalur baru terbentuk kembali.
- Hal ini akan menjadi masalah besar jika pengiriman data memerlukan *real time delivery*, seperti *voice* dan *streaming video* disebabkan delay yang sangat tinggi.

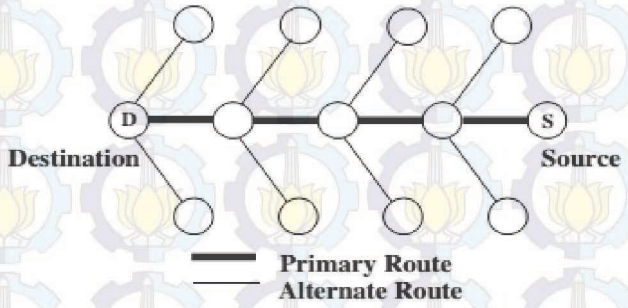
AODV-BR menawarkan sebuah algoritma baru yang dapat mengatasi persoalan-persoalan diatas, yaitu dengan menggunakan struktur *mesh* untuk menyediakan jalur alternatif yang ada pada *on-demand routing protokol* tanpa membentuk pesan tambahan atau dengan kata lain hanya menggunakan tiga pesan utama yang ada pada AODV tradisional yaitu RREP. Jalur alternatif sangat dibutuhkan dalam *ad hoc* karena jaringan *wireless* rentan terjadinya *route break* karena mobilitas *node*, *signal interference*, dan *packet collision* [9].

AODV-BR memiliki dua konsep utama dalam mengimplementasikan algoritma atau metode yang ditawarkan, yaitu *route construction* dan *Route*

Maintenance [9]. Pada bagian selanjutnya akan dijelaskan mengenai kedua konsep tersebut.

2.5.1 *Route Contruction*

Proses pembentukan AODV-BR tidak melakukan modifikasi terhadap ketiga pesan utama yang ada pada AODV tradisional, AODV-BR hanya memanfaatkan *route request* dan *route reply* yang ada yaitu dengan cara mem-*flooding* RREQ seperti pada AODV tradisional, kemudian RREP yang menuju *source node* akan disebarkan dan dicatat pada tabel *routing* pada masing-masing *node* dengan membuat setiap *node* dapat *overhear* terhadap RREP yang dikirim oleh *destination*, dan pada saat itu juga terbentuklah jalur utama dan jalur alternatif yang disebut struktur *mesh*.



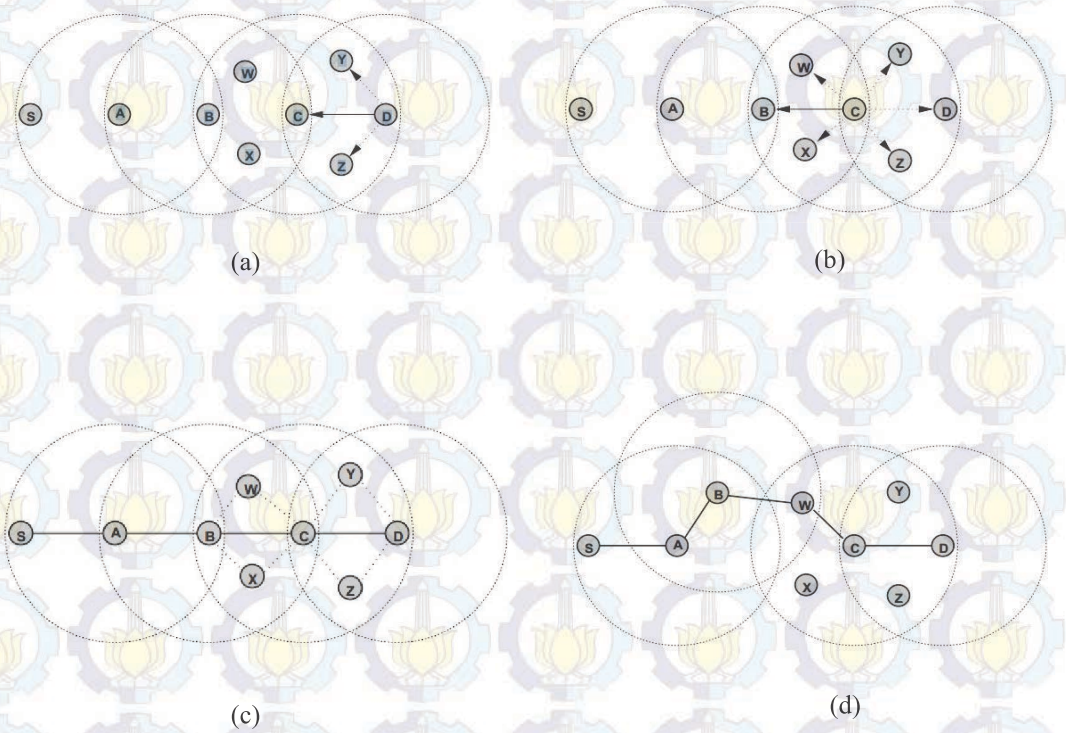
Gambar 2.11 Struktur Mesh yang menyerupai tulang Ikan, terbentuk jalur utama dan jalur alternatif [8]

Skema AODV-BR yaitu dengan memanfaatkan penyebaran *route reply (RREP)* secara *broadcast* yang ada pada AODV tradisional yang nantinya dapat membentuk stuktur *mesh* yang menyerupai tulang ikan, lihat pada Gambar 2.11. Setiap node bisa saja menerima banyak *reply* untuk rute yang sama, namun hanya akan dipilih jalur terbaik dan dimasukkan ke tabel *routing*.

2.5.2 *Route Maintenance*

Paket data dikirim melalui jalur utama sampai terdeteksi terjadinya kerusakan pada jalur utama tersebut misalnya menerima sinyal *feedback link layer* dari MAC *protokol*, tidak menerima *passive acknowledgments*, tidak menerima *hello packets* dalam beberapa waktu atau yang lain, maka satu *hop* akan mem-*broadcast* data dengan *header* yang memberitahukan bahwa *link* rusak dan membutuhkan jalur alternatif. Sedangkan untuk menghindari *looping* saat

menggunakan *alternative route*, *node* mem-*forward* data jika paket data tidak diterima dari *nexthop* tujuan dan tidak terjadi duplikasi. Ketika sebuah node dari rute utama menerima paket data dari rute alternatif, maka node beroperasi normal dan meneruskan paket ke *hop* berikutnya ketika paket tidak diduplikat. *Node* yang terdeteksi *link break* juga mengirim pesan *Route Error (RERR)* ke node sumber untuk memulai kembali penemuan rute.



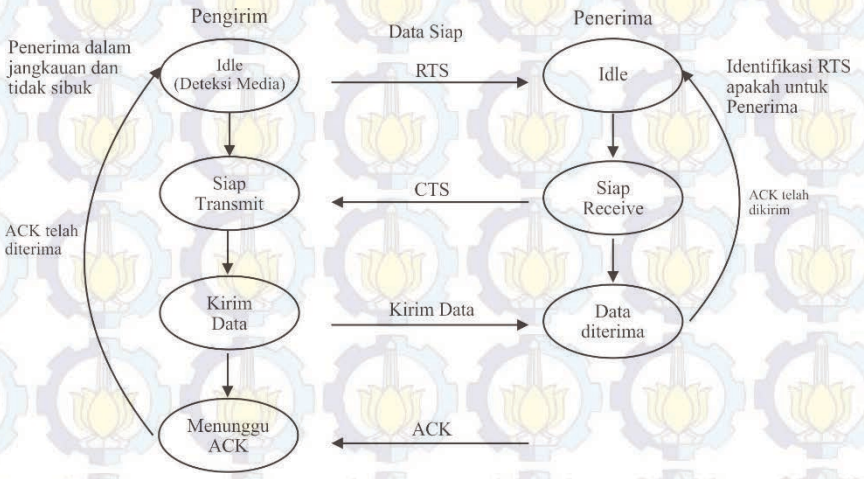
Gambar 2.12 Beberapa Konstruksi Jalur dan Penggunaannya: (a) *Node* Mengirim Sebuah RREP, (b) *Node* Meneruskan RREP *multicats*, (c) Jalur Utama dan Jalur Alternatif Ditetapkan, (d) Paket Data Dikirimkan Melalui Jalur Alternatif Bila Jalur Utama Terputus [9]

Di dalam protokol AODV tradisional, jalur yang *time out* akan diperbaharui untuk durasi waktu tertentu jika proses pengiriman data masih berlangsung. Disini menggunakan teknik yang sama untuk waktu keluar dari jalur alternatif. *Node* yang menyediakan jalur alternatif dapat menditeksi paket data. *Node* akan memperbaharui jalur, jika paket tersebut dikirimkan oleh *hop* berikutnya ke tujuan seperti yang ditunjukkan dalam tabel jalur alternatif dan *node* akan menghapus jalur dari tabel jika jalur alternatif tidak diperbaharui pada saat waktu habis.

2.6 Protokol MAC 802.11

Protokol MAC *(Medium Access Control)* merupakan lapisan *Datalink* yang mengatur dan menjaga komunikasi diantara stasiun-stasiun pada jaringan WLAN. Lapisan ini juga berfungsi sebagai pengatur akses protokol ke media fisik jaringan dan mengkoordinasi akses dalam menggunakan kanal radio untuk mempermudah komunikasi melalui media *wireless*.

Pada umumnya, MAC *wireless* LAN 802.11 menggunakan protokol *CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance)*. CSMA/CA dapat menghindari tabrakan paket dengan menggunakan mikanisme *frame* RTS/CTS *(Request to send/ Clear to send)* dan *acknowledgement (ACK)*. RTS/CTS merupakan *frame* yang mengizinkan akses poin untuk mengatur penggunaan media pada sebuah stasiun, maksudnya *frame* RTS digunakan *sender* untuk memberitahukan akan melakukan pengiriman data pada *receiver* setelah mendeteksi medium dalam kondisi *idle*, *frame* CTS digunakan untuk sebagai pesan balasan dari *receiver* untuk memberitahukan *sender* bahwa *receiver* telah menerima RTS dan siap menerima data, sedangkan *frame* ACK dikirim oleh stasiun penerima untuk mengkonfirmasi bahwa paket telah terkirim [15].



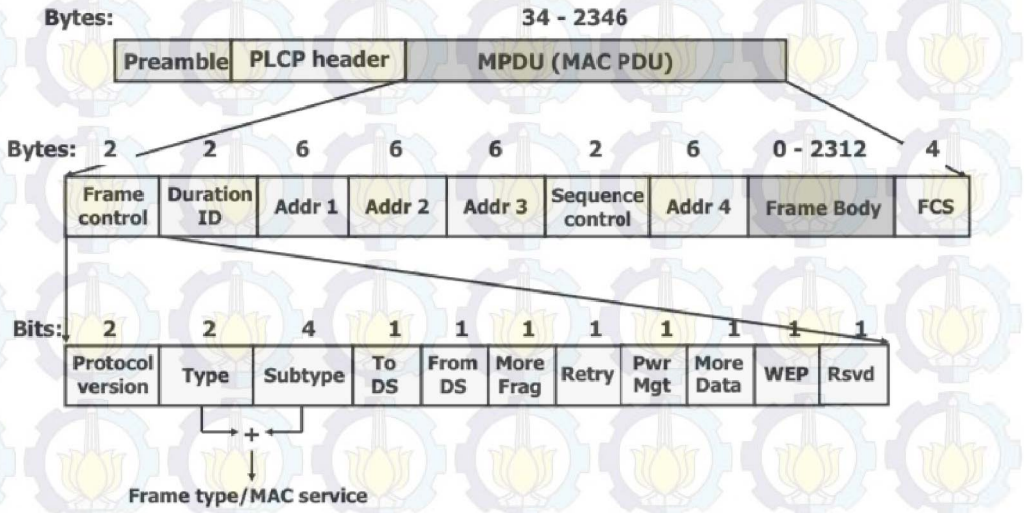
Gambar 2.13 Proses Pengiriman RTS / CTS [11]

Sebuah pesan yang dikirim pada layer *datalink* berupa *frame*, *frame* adalah paket yang dienkoding untuk keperluan transmisi data. Standar WLAN 802.11 mendefinisikan beberapa tipe *frame* yang digunkan pada setiap komunikasi antar station maupun yang digunakan untuk sebagai pengatur dan pengedali *link wireless*.

Setiap *frame* mempunya *header* dengan beberapa *field* yang digunakan diantara *MAC Sublayer* [15].

Terdapat 4 susunan utama dalam wireless LAN MAC melakukan manajemen, dan pengontrolan *frame*, yaitu: *Frame Control, MAC Header, Frame Body,* dan *Frame Check Sequence (FCS)*.

Format frame keseluruhan memiliki panjang 2346 Byte pada jaringan wireless LAN 802.11 dapat ditunjukkan pada Gambar 2.14.



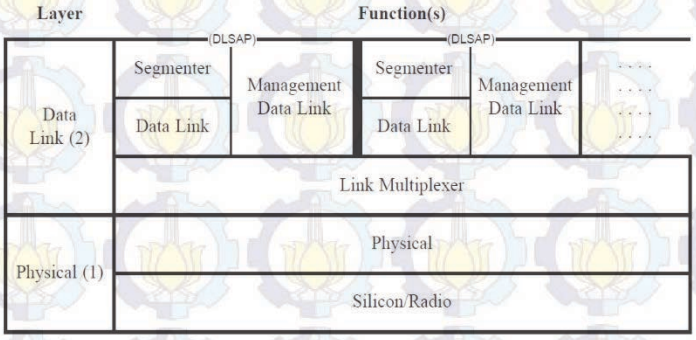
Gambar 2.14 Format *Frame* pada MAC 802.11 [15]

2.7 Protokol Radio AX.25

Protokol adalah sebuah aturan yang mendefinisikan beberapa fungsi yang ada dalam sebuah jaringan komputer, misalnya mengirim pesan, data, informasi dan fungsi lain yang harus dipenuhi oleh sisi pengirim dan sisi penerima agar komunikasi dapat berlangsung dengan benar, walaupun sistem yang ada dalam jaringan tersebut berbeda sama sekali. Protokol ini mengurusi perbedaan format data pada kedua sistem hingga pada masalah koneksi listrik. Standar protokol yang terkenal yaitu OSI *(Open System Interconnecting)* yang ditentukan oleh ISO *(International Standart Organization)*.

AX.25 adalah protokol layer 2 (merujuk pada *OSI Layer Reference*) yaitu *Data Link Layer*. Sebagai protokol pada lapisan 2 AX.25 bertanggungjawab untuk menbangun *link connection*, menyediakan prosedur *logic* untuk information transfer, dan *link disconnection*. Sehingga AX.25 cukup lengkap untuk dijadikan

contoh implementasi sebuah protokol [16]. Protokol Amatir X.25 (AX.25) adalah protokol radio turunan dari X.25 yang digunakan dalam jaringan paket radio. Untuk membangun hubungan antara dua buah terminal melalui *physical layer* dan lapisan *data link*. Protokol ini akan bekerja pada dua kondisi transmisi yaitu *half duplex* dan *full duplex*. Selanjutnya dua lapisan yang ada pada protokol ini yaitu *physical layer* dan lapisan *data link* dapat dibagi lagi ke dalam beberapa status keadaan seperti yang ditunjukkan pada Gambar 2.15. keadaan yang dimaksudkan adalah mendefinisikan keadaan suatu *link* komunikasi radio untuk *multi link* .



Gambar 2.15 Keadaan Protokol AX.25 untuk *Multi Link* [12]

Dengan *mengacu* Gambar 2.15 pada protokol AX.25 lapisan paling atas dari *layer* 2 adalah *Data Link Access Point* (DLAP). DLAP merupakan lapisan yang akan menyediakan untuk meneruskan paket data ke *layer* 3. Pada saat terjadi transmisi data maka hubungan antara *data link* diberikan oleh lapisan *data link* dengan menggabungkan antara dua atau lebih DLAP. Kemudian *data link* akan memberikan suatu urutan bit yang dipecah menjadi beberapa blok data yang disebut *frame*.

Format protokol AX.25 pada teknologi *packet radio* ditunjukkan pada Gambar 2.16 dengan memiliki maksimum 256 *byte* dalam satu *frame*. Pada pengiriman data kecepatan tinggi dan aplikasi TCP/IP dilakukan beberapa perubahan sehingga dimungkinkan untuk mengirim lebih dari 256 *byte* data dalam satu *frame*.

Flag	AX.25 Transfer Frame Header (128 bits)				Information Field	Frame-Check Sequence	Flag
	Destination Address	Source Address	Control Bits	Protocol Identifier			
8	56	56	8	8	0-2048	16	8

Gambar 2.16 Format *frame* Protokol Link AX.25 yang Digunakan dalam Komunikasi Paket Radio [16]

Frame AX.25 dimulai dan ditutup oleh *flag byte* yang berisi 01111110. *Adress field* berisi alamat tujuan, alamat pengirim paket dan stasiun-stasiun yang berfungsi sebagai *relay*. Dengan menggunakan stasiun lain sebagai *relay*, maka stasiun yang digunakan sebagai *relay* tersebut dapat mengirimkan data ke tempat tujuan. Hal tersebut dikenal sebagai konsep *digipeater* (*digital repeater*). Pada *control field* berisi identifikasi bentuk *frame* AX.25 yang dikirim. Apakah *frame* ini untuk melakukan koneksi (membuka hubungan komunikasi), koreksi (jika ada *frame* AX.25 yang rusak dalam pengiriman), untuk *broadcast* dan sebagainya. *Packet* ID (PID) digunakan untuk memberitahukan jenis data yang dikrim, apakah data berbentuk teks, binary atau protokol lapisan *network*. *Frame Check Sequense* (FCS) digunakan oleh bagian penerima pada proses pendeteksian kesalahan [17].

Protokol AX.25 dalam komunikasi data radio mempermudah pengguna untuk berkomunikasi data secara langsung dengan menggunakan program *hyperterminal* dan pengguna tidak perlu repot dengan masalah *acknowledgement* karena sudah ditangani oleh *terminal node controller* (TNC). Untuk melakukan komunikasi data yang dapat dikontrol secara langsung oleh *software* lebih fleksibel apabila menggunakan protokol lapisan yang lebih bawah. Sebagian besar TNC mendukung penggunaan *keep it simple and stupid* (KISS) sebagai protokol pada lapisan bawah untuk mengirimkan datagram secara langsung dari komputer/mikrokontroler. KISS *frame* ini sudah dilengkapi dengan proses deteksi kesalahan. KISS *frame* ini juga digunakan untuk komunikasi data secara langsung dengan menggunakan protokol TCP/IP [16].

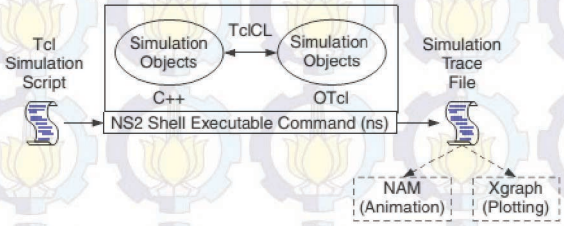
2.8 *Network Simulator 2*

Network Simulator (versi 2), biasa dikenal dengan NS2. Ns merupakan *software* untuk membuat sample simulasi kejadian pergerakan *(event-driven)* yang digunakan untuk mempelajari sifat dinamis dari sebuah jaringan dalam sistem komunikasi, misalnya mensimulasikan fungsi jaringan kabel, nirkabel, dan protokol dalam jaringan seperti algoritma routing, TCP dan UDP dapat disimulasikan menggunakan NS2. Secara umum, NS juga memfasilitasi pengguna misalkan dalam mensimulasikan sebuah protokol dalam jaringan yang sangat sesuai dengan kondisi ideal dan sesuai dengan karakteristiknya [18].

Sejak tahun 1989, NS sudah sangat popular di kalangan komunitas riset dikarenakan kefleksibelannya. Sudah beberapa kali dilakukan revolusi dan revisi terhadap NS disesuaikan dengan keberadaan alat jaringan yang terus berkembang berkat hasil-hasil riset yang terus dilakukan di lapangan, diantaranya adalah yang telah dilakukan oleh *University of California* dan *Cornell University*. Sejak tahun 1995, *Advanced Research Projects Agency (DARPA)* mendukung pengembangan NS melalui proyek *testbed Virtual Internetwork (Vint)*. Saat ini *National Science Foundation (NSF)* juga tergabung dalam pengembangan ini, dan masih banyak lagi kelompok peneliti yang mengembangkan NS agar dapat digunakan dalam banyak aspek simulasi.

2.8.1 Arsitektur dasar NS2

Network Simulator (NS) terbangun dari beberapa *script* yang saling memiliki kesenambungan. Bahasa pemprograman yang digunakan untuk membangun NS meliputi TCL, C++, OTCL, NAM, dan Xgraph. Berdasar pada bahasa pemprograman tersebut terbangunlah sebuah *software* simulasi jaringan ini. Kelima bahasa pemprograman tersebut memiliki kesinambungan satu sama lainnya, karena ia akan bekerja secara kolektif untuk satu tujuan. Adapun korelasi dari pemprograman tersebut seperti yang ditunjukkan pada Gambar 2.17.



Gambar 2.17 Arsitektur Dasar pada NS [18]

NS menfasilitasi pengguna untuk merancang sebuah argument atau skenario simulasi, skenario tersebut ditulis dalam *script file* TCL. TCL memungkinkan bagi pengguna untuk membuat animasi dalam simulasi, pengukuran serta plot grafik untuk hasil pengukuran dalam sebuah simulasi [18].

NS terdiri dari dua bahasa pemprograman utama; C++ dan *Object-oriented Tool Command Language* (Otcl). Kedua bahasa pemprogrman tersebut memiliki peran masing-masing. C++ memiliki peran untuk mendefinisikan mekanisme internal dari object simulasi, sedangkan Otcl menyiapkan simulasi dengan merakit dan mengkonfigurasi objek serta penjadwalan peristiwa secara diskrit. C++ dan Otcl digabungkan kinerjanya secara bersamaan menggunakan TclCL [18].

Dalam file TCL, setelah selesai simulasi akan memunculkan file TR atau *trace file* yang berisi *log* kejadian saat simulasi. Semua aktifitas objek saat simulasi terkem secara otomatis dalam *file .tr* tersebut. Dengan *file trace* tersebut kita akan mendapatkan hasil pengujian dari skenario simulasi yang kita buat dalam *file* TCL. Hasil dari *file trace* tersebut bisa kita olah untuk mendapatkan gambaran hasil simulasi dengan menggunakan *tool* yang sudah terpasang pada *ns all in one*, yaitu *Xgraph* dan *Nam*. *Xgraph* digunakan untuk *plot* hasil simulasi kedalam bentuk grafik, sedangkan *Nam* digunakan untuk memberikan gambaran simulasi berupa animasi sesuai dengan skenario yang dibuat pada waktu menuliskan *script* TCL [18].

2.8.2 Pengolahan Data Simulasi

Akhir simulasi pada NS2, akan mengahsilkan keluaran *file* berupa *log* yang mencatat seluruh kejadian saat simulasi, seperti jumlah paket yang dikirim dan diterima, penoromoran node, serta seluruh informasi-informasi yang terjadi saat

simulasi berlangsung. Adapun teknik pengambilan data dari hasil sumulasi adalah sebagai berikut:

2.8.2.1 *File Trace*

File trace merupakan pencatatan seluruh *event* (kejadian) yang dialami oleh node pada saat simulasi berlangsung. Pembuatan *file trace* dilakukan dengan memanggil objek *trace* yang terdapat pada *library* NS. Berikut ini merupakan contoh dari *file trace*.

```
r 1.3556 3 2 ack 40 --------------- 1 3.0 0.0 15 201
+ 2.3556 2 0 ack 40 --------------- 1 3.0 0.0 15 201
- 1.3556 2 0 ack 40 --------------- 1 3.0 0.0 15 201
d 1.35576 2 3 tcp 1000 --------------- 1 0.0 3.0 29 199
```

Gambar 2.18 Contoh *Trace File Output* Simulasi pada Jaringan *Wired*

Contoh file *trace* di atas memilikii format tabel seperti gambar berikut:

Event	Time	From node	To node	Pkt type	Pkt Size	Flags	fid	Scr addr	Dst addr	Seq num	Pkt id

Gambar 2.19 Format Tabel pada File *Trace*

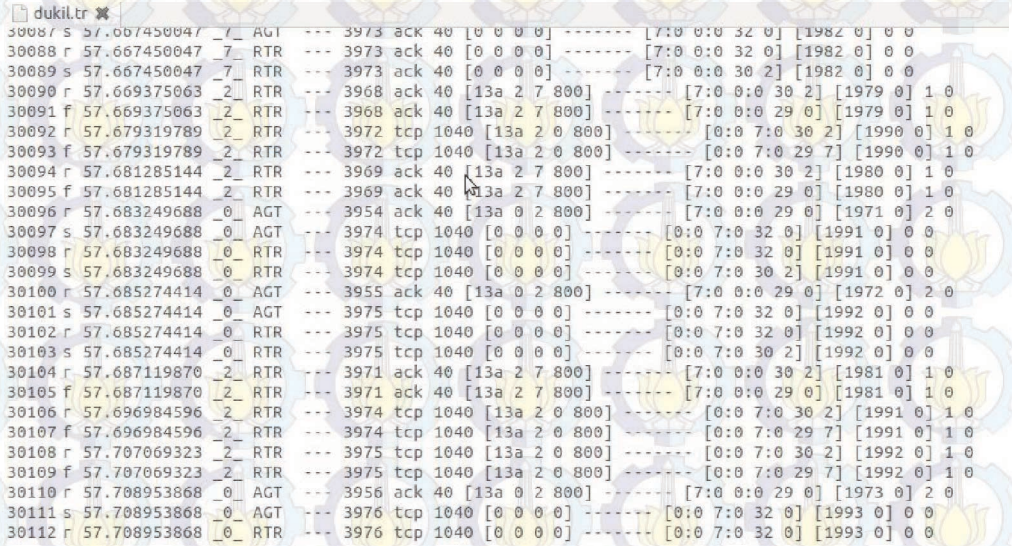
Adapun maksud dari setiap *field* kolom pada Gambar 2.19 diatas adalah sebagai berikut:

- *Event* : Adalah kejadian yang dicatat pada saat simulasi di NS, meliputi:
  r : *receive* (paket yang telah diterima oleh *node dest.*)
  d : *Drop* (sejumlah paket yang mengalami *drop* saat antrian)
  + : *Enter queue* yaitu paket masuk antrian atau keluar dari *node*.
  - : *Leave queue* paket yang keluar dari antrian
- *Time* : Adalah waktu event dalam detik bahkan mili detik
- *From node* : Menyatakan keberadaan paket. Pada saat *event*, paket berada di antrian *from node* atau *to node*..
- *To node* : Menyatakan keberadaan paket
- *Pkt size* : ukuran *(bytes)* dari paket
- *Flag* : Digunakan sebagai penanda. Pada data di atas *flags* tidak digunakan. Adapun macam-macam dari penanda yang dapat digunakan yaitu:

- ✓ E : Digunakan apabila terjadi kongesti *(Congstion Experienced / CE)*
- ✓ N : Digunakan untuk mengindikasi *ECT (ECN-Capable-Transport)* pada *header* IP
- ✓ C : Digunakan untuk *ECN-Echo*
- ✓ A : Digunakan untuk pengurangan *window* kogesti pada *header* TCP
- ✓ P : Untuk prioritas
- ✓ F : Untuk TCP *fast start*

- *Fid* : Merupakan penomoran unik dari tiap aliran data
- *Scr addr* : *Node* asal, memiliki format *node.port*
- *Dst. Addr* : *Node* tujuan, memiliki format *node.port*
- *Seq Num* : Adalah *sequence number* dari paket tersebut
- *Paket id* : Merupakan paket ID dari paket.

Sedangkan *file trace* yang dihasilkan untuk jaringan *wireless*, contohnya seperti Gambar 2.20 berikut:



Gambar 2.20 Contoh *File Trace* Jaringan *Wireless Ad Hoc*

Keterangan:

- *Next Hop Info* berisi *–Hs* dan *– Hd. –Hs* merupakan untuk *node* ini, sedangkan *–Hs* adalah Id untuk *next hop* yang akan melanjutkan paket ke tujuan.

- *Node Property tags* berisi:
  - ✓ Ni : Id pada *node*
  - ✓ Nx : Koordinat X yang dimiliki sebuah *node*
  - ✓ Ny : Koordinat Y yang dimiliki sebuah *node*
  - ✓ Nz : Koorninat Z yang dimiliki sebuah *node*
  - ✓ Ne : Level yang dimiliki *node*
  - ✓ N1 : *Level trace*, seperti AGT, RTP, MAC
- Paket yang dienkapsulasi ke level MAC
  - ✓ Ma : Durasi
  - ✓ Md : Alamat *ethernet* tujuan
  - ✓ Ms : Alamat *ethernet* pengirim
  - ✓ Mt : Tipe dari *ethernet*
- Paket Informasi pada level IP
  - ✓ Is : Alamat pengirim, *port* yang digunakan pengirim
  - ✓ Id : Alamat tujuan dan port yang digunakan
  - ✓ It : Tipe paket
  - ✓ Il : Ukuran paket
  - ✓ If : *Flow Id* (Id jalur)
  - ✓ Ii : ID unik
  - ✓ Lu : Nilai *time to life*

Untuk melakukan analisa terhadap *file trace* yang dihasilkan dari data simulasi, untuk memudahkannya dapat menggunakan *shell programming* atau AWK untuk melakukan penyaringan terhadap informasi parameter yang dibutuhkan saja.

AWK merupakan sebuah program yang biasanya digunakan untuk menfilter teks yang banyak untuk mengambil beberapa yang dibutuhkan saja. AWK dapat pula digunakan untuk mencari bentuk atau model dalam sebuah *file* teks. AWK ini memiliki karakterisrik pemprograman berbasis *shell* atau C. Dengan menggunakan script AWK, dapat memudahkan pemfilteran terhadap data-data yang dibutuhkan saja, misalnya pada tesis ini akan digunakan untuk menggambil data parameter QoS jaringan, seperti *delay* dan *throughput* saja [5].

2.8.2.2 *Parsing*

Parsing merupakan teknik untuk mendapatkan sebuah informasi yang dibutuhkan dari *file trace* yang mencatat seluruh kejadian saat simulasi berlangsung. Untuk men-*parsing file trace*, dibutuhkan *file* dengan *script* tertentu yang disimpan dalam bentuk *awk*. *File awk* digunakan untuk menfilter beberapa data yang dibutuhkan dalam *file trace*, sehingga hanya data-data yang dibutuhkan saja yang dapat dipisahkan. Contoh perintah parsing menggunakan *script awk* ditulis dengan format seperti pada Gambar 2.21 berikut:

```
exec awk -f delay.awk dukil.tr > delay.tr
```

Gambar 2.21 Contoh Perintah *Parsing* pada *File Trace*

Perintah pada Gambar 2.21 dijalankan pada CLI sesuai dengan OS yang digunakan. Jika menggunakan Linux, maka perintah *parsing* tersebut dapat dijalankan di terminal. *Delay.awk* adalah *file* yang berisi *script* parsing terhadap informasi *delay* yang terjadi pada saat simulasi berlangsung yang tersimpan pada *file trace* yang diberi nama *dukil.tr*. Untuk melakukan *parsing*, file awk harus diletakkan pada folder yang sama. Sedangkan *delay.tr* adalah file yang berisi hasil paring berupa *delay* pada saat simulasi berlangsung.

2.9 Pengukuruan *QoS* Jaringan

Untuk mengetahui kualitas dari teknik yang disusun dalam merencanakan suatu sistem, baik sistem simulasi maupun terapan, terdapat beberapa parameter yang dibutuhkan untuk diukur agar dapat menyimpulkan bahwa sebuah sistem dapat dianggap bekerja dengan baik. Dalam sistem jaringan *Mobile Ad Hoc Network (MANET)* terdapat beberapa parameter pengukuran yang dibutuhkan sebagai bahan untuk dapat mengambil kesimpulan, parameter yang diperlukan untuk mengetahui kualitas layanan dalam jaringan salah satunya dengan menggunakan parameter *Delay* dan *Throughput* [5].

2.9.1 Delay

Delay time adalah waktu yang dibutuhkan untuk mengirim sebuah paket data ke node yang lain selama proses simulasi berlangsung. Dari *file trace* yang dihasilkan saat simulasi selesai dapat dihitung *delay time* antara node pengirim dengan node penerima dalam proses transmisi data. *Delay* ini dapat diperoleh dari hasil perhitungan selisih waktu antara waktu paket dikirim dan paket diterima. Penghitungan ini dapat dirumuskan sebagai berikut:

$$waktu\ tunda\ (t) = \frac{Tr - Ts}{Pr} \quad 0 \leq t \leq T \qquad (2.1)$$

Dimana:

Tr : Waktu penerimaan paket
T : Waktu yang digunakan
Ts : Waktu pengiriman paket
t : Waktu pengambilan sampel
Pr : Paket yang diterima (paket)

2.9.2 Throughput

Throughput adalah waktu yang dibutuhkan untuk pengiriman paket dari satu node pengirim ke node tujuan dalam satuan waktu. *Throughput* ada juga yang mendefinisikan sebagai laju data yang terukur pada satuan waktu tertentu. *Throughput* sama halnya dengan konsumsi *bandwith* dimana kecepatan kanal yang diperoleh oleh satu user.

Kendatipun *throughput* memiliki satuan dan rumus yang sama dengan kecepatan laju data, tetapi *throughput* lebih menggambarkan laju data yang sebenarnya pada suatu waktu tertentu dan pada kondisi jaringan internet tertentu yang digunakan untuk mengirim dan menerima suatu data dalam ukuran waktu. Dalam menghitung *throughput* ini dapat menggunakan persamaan berikut:

$$throughput = \frac{Pr}{1\ detik}\ paket\ /\ detik \qquad 0 \leq t \leq T \qquad (2.2)$$

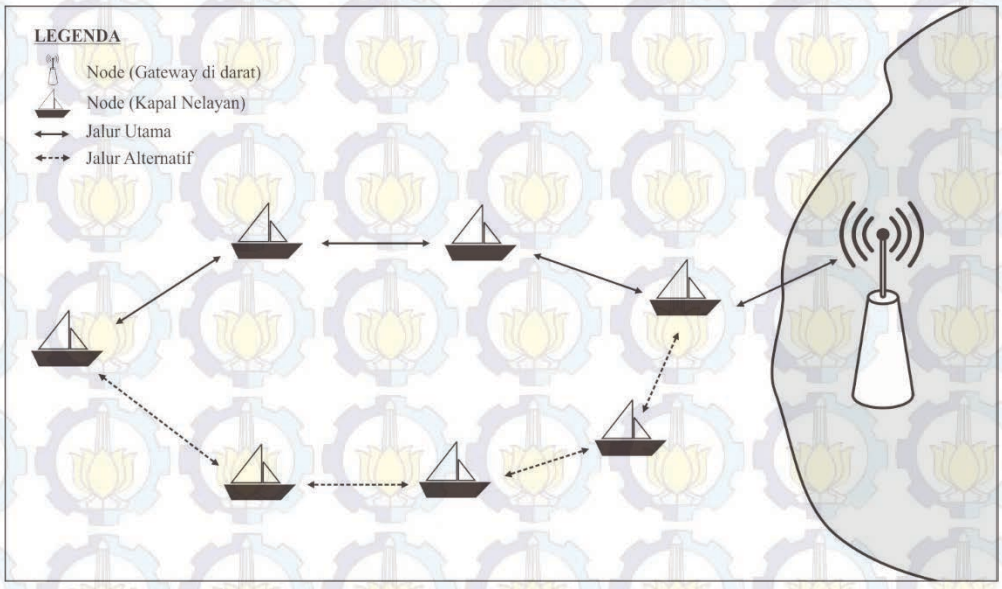
Dimana:

Pr : Paket yang diterima (paket)
T : Waktu yang digunakan (detik)
t : Waktu pengambilan sampel (detik)

BAB 3
METODE PENELITIAN

3.1 Gambaran Umum Sistem

Untuk menganalisa kenerja dari penerapan protokol MANET pada komunikasi VMeS, digunakan protokol manet reaktif yaitu AODV *backup routing* (AODV-BR). AODV-BR merupakan pengembangan dari protokol AODV tradisional yang menawarkan *multiple route* dikarenakan AODV tradisional sering terjadi masalah *route break* saat melakukan komunikasi dan harus melakukan *route discovery* dari awal sehingga membutuhkan waktu yang lebih lama. AODV-BR menawarkan *alternative route* agar pada saat melakukan komunikasi tidak perlu melakukan *route discovery* melainkan memanfaatkan jalur alternatif yang disediakan jika terjadi *route break*. Pemaparan yang jelas untuk skema *routing* dari AODV-BR ini sudah dibahas pada subbab 2.5. Untuk melakukan analisa terhadap kinerja AODV-BR pada VMeS digunakan dua cara, yaitu mensimulasikan kinerja protokol *routing* menggunakan *software Network Simulator 2.35* dan mengimplementasikannya pada testbed menggunakan Laptop untuk menguji kinerja dari protokol AODV-BR secara riil.



Gambar 3.1 Pemanfaatan Jalur Alternatif pada Komunikasi VMeS

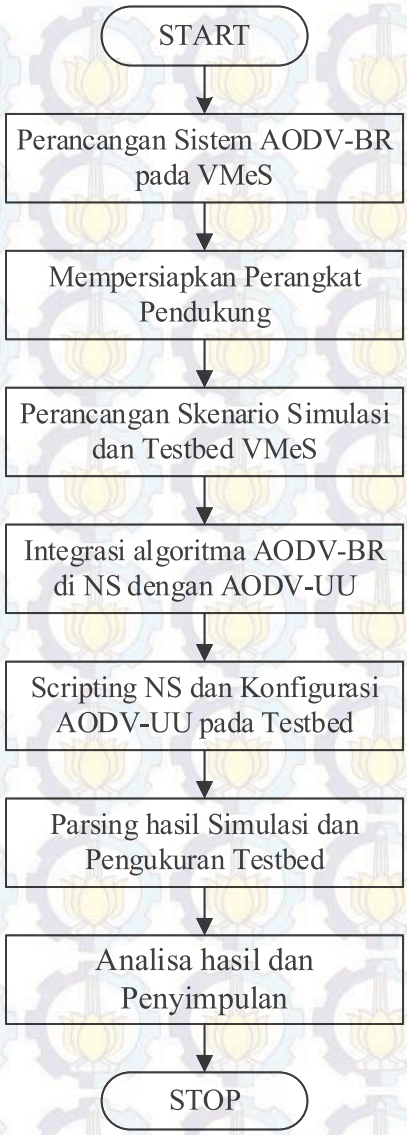
Analisa kinerja protokol routing AODV-BR pada penelitian ini, dilakukan pada protokol *medium access control (MAC)* 802.11 dikarenakan peralatan yang akan digunakan oleh VMeS saat ini masih dalam tahap pengembangan.

3.2 Rancangan Penelitian

Pada penelitian ini akan dibuat rancangan simulasi dan *testbed* menggunakan laptop untuk mengetahui kinerja dari protokol routing AODV *Backup Routing* (AODV-BR) pada sistem komunikasi VMeS secara riil. Pada tahap perancangan ini akan ditentukan skenario simulasi dan implementasi serta teknik pengambilan data serta analisa data. Untuk melakukan simulasi protokol *routing* AODV-BR pada sistem komunikasi VMeS ini digunakanlah *software Network Simulator 2* (NS 2.35) sebagai simulatornya. Untuk implementasi pada laptop digunakan AODV-UU sebagai *software* protokol *routing* yang berjalan pada Linux dengan kernel 2.6.x.

Penelitian ini diawali dengan mempersiapkan perangkat-perangkat berupa laptop yang sudah terinstall OS Linux dengan distro Ubuntu yang didalamnya sudah terpasang perangkat lunak *Network Simulator*. Untuk tahap-tahapan dalam perencanaan simulasi akan dibahas lebih rinci pada sub bab 3.4.1. Untuk uji coba kinerja AODV-BR pada perangkat, dibutuhkan beberapa laptop yang difungsikan sebagai *node*. Untuk langkah-langkah perancangannya akan dibahas pada sub bab 3.4.2. Setelah perangkat yang dibutuhkan untuk analisa kinerja protokol roting AODV-BR sudah siap, maka dilakukan pembuatan skenario simulasi dan implementasi sesuai dengan skenario VMeS dengan mensimulasikan kinerja protokol *routing* AODV-BR menggunakan NS dan meguji coba pada *testbed* laptop menggunakan AODV-UU yang bertujuan untuk memperoleh hasil-hasil pengukuran berdasarkan parameter kinerja yang ditentukan. Selanjutnya akan dilakukan pengolahan hasil simulasi dan hasil uji coba *routing* pada perangkat untuk kemudian dijadikan bahan untuk menganalisa hasil akhir dari penelitian terhadap kinerja protokol *routing* AODV-BR pada komunikasi VMeS.

Berikut ini adalah *flowchart* dari langkah-langkah yang akan dilakukan dalam penelitian buku tesis ini:



Gambar 3.2 *Flowchart* Tahapan Penelitian

Gambar 3.2 merupakan *flowchart* tahapan penelitian yang akan dilakukan untuk menganalisa kinerja protokol routing AODV-BR pada komunikasi VMeS, *flowchart* tersebut dibagi menjadi dua tahapan utama, yaitu tahapan yang dilakukan saat simulasi dan tahapan yang akan dilakukan saat implementasi dan uji coba pada perangkat *testbed*. Adapun tahapan-tahapan yang akan dilakukan pada penelitian ini adalah sebagai berikut:

1. Tahap awal adalah merencanakan sistem yang akan digunakan untuk menganalisa kinerja dari protokol routing AODV-BR, sistem yang akan digunakan meliputi simulasi menggunakan *tool network simulator* dan uji coba protokol untuk implementasi pada *testbed* laptop.
2. Mempersiapkan perangkat pendukung berupa perangkat lunak *(software)* dan perangkat keras *(hardware)* yang akan digunakan untuk menganalisa kinerja dari protokol routing AODV-BR. Untuk hal ini akan dirinci pada sub bab berikutnya.
3. Tahap Simulasi
   - Merancang skenario dan algoritma routing yang digunakan pada NS. Untuk skenario dan algoritma routing dari AODV-BR ini ditulis dalam bahasa pemprograman C++. Untuk penelitian ini, menggunakan modul *routing* yang sudah ada pada NS, setelah dilakukan sedikit modifikasi.
   - Proses *scripting* topologi VMeS di NS, topologi yang digunakan disamakan dengan kondisi riil pada lingkungan jaringan VMeS. Untuk membuat topologi ini dilakukan penulisan *script* agar dapat dapat dieksekusi oleh NS, yaitu menggunakan bahasa pemprograman TCL/OTCL.
   - Parsing Hasil simulasi merupakan tahap pemfilteran data yang tercatat pada *log* yang dihasilkan dari *file trace* akhir setiap simulasi di jalankan. Untuk melakukan ini, peneliti menggunakan *script AWK* untuk melakukan parsing terhadap parameter yang digunakan yaitu meliputi nilai *delay* dan *throughput* pada saat simulasi dijalankan.
4. Tahap Impelementasi
   - Perancangan skenario implementasi diperlukan untuk menentukan perangkat yang dibutuhkan, topologi, dan lokasi yang akan digunakan untuk melakukan uji coba kinerja pada protokol routing AODV-BR ini.
   - Integrasi *script* C++ AODV-BR di NS ke dalam AODV-UU dibutuhkan untuk menyamakan skenario yang sudah dibuat pada NS. AODV-UU merupakan *software* yang terbangun dari bahasa C yang dapat terintegrasi dengan NS.
   - Instalasi dan konfigurasi AODV-UU diinstal pada OS Ubuntu 10.04.1 yang membutuhkan banyak *software* tambahan, seperti *build-essential*, *linux-*

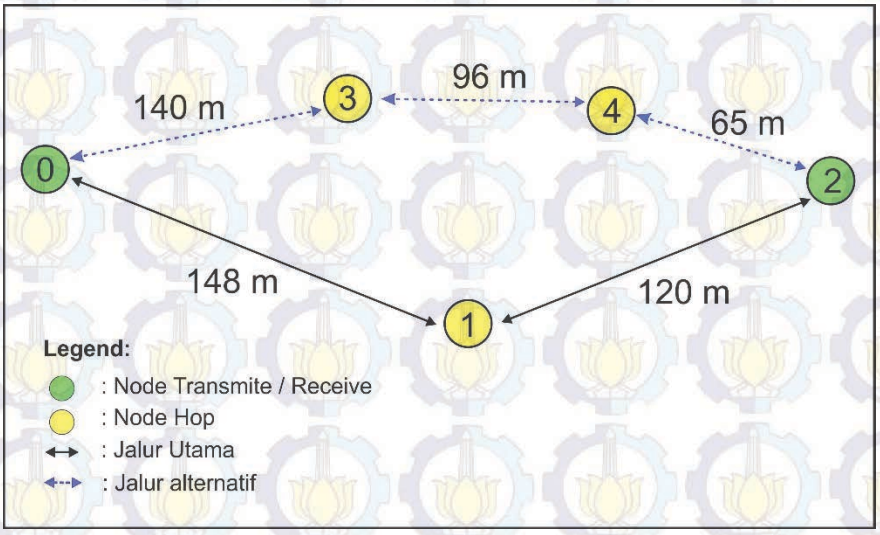
header yang ter-*update*, dan melakukan *compailing* pada AODV-UU agar dapat berjalan pada kernel OS 2.6.28.

- Uji coba dan pengambilan data implementasi dilakukan saat semua konfigurasi perangkat dan *software* yang dibutuhkan sudah siap digunakan. Untuk mengambil data dari uji coba pada *testbed*, digunakan *software Iperf* sebagai alat untuk melakukan testing pengiriman paket yang sudah ditentukan untuk uji coba.

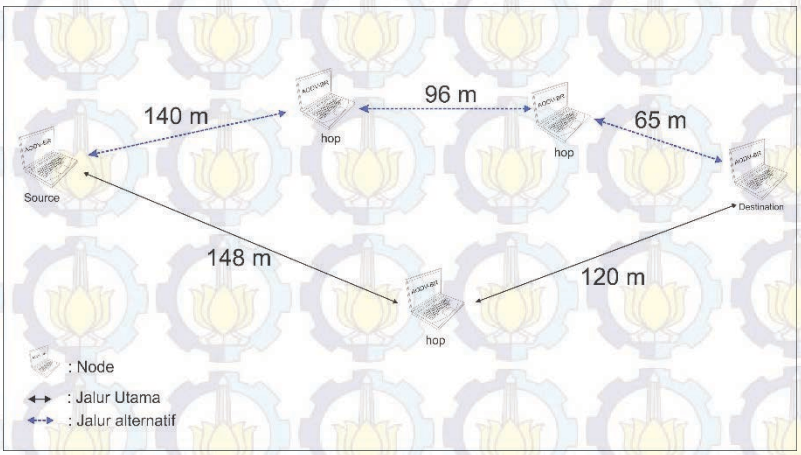
5. Analisa hasil dan penyimpulan dilakukan setelah proses dan tahapan-tahapan dari simulasi dan uji coba pada perangkat sudah dilakukan.

3.3 Desain Topologi Jaringan

Desain topologi yang digunakan pada penelitian ini meliputi dua aspek, yaitu desain topologi pada simulasi dan desain topologi pada *testbed*. Desain dari kedua aspek tersebut disamakan agar dapat dibandingkan kinerja dari masing-masing uji coba dari kinerja protokol routing AODV-*Backup Routing* terhadap komunikasi VMeS. Pada desin topologi *node* terbagi menjadi 3 bagian, yaitu *node source*, *hop*, dan *receiver*. Untuk menguji skema *backup routing* pada AODV dilakukan pada pengaturan *node* yang dijadikan *neighbor* atau *hop*, yaitu dari konfigurasi posisi dan konfigurasi arah pergerakan.

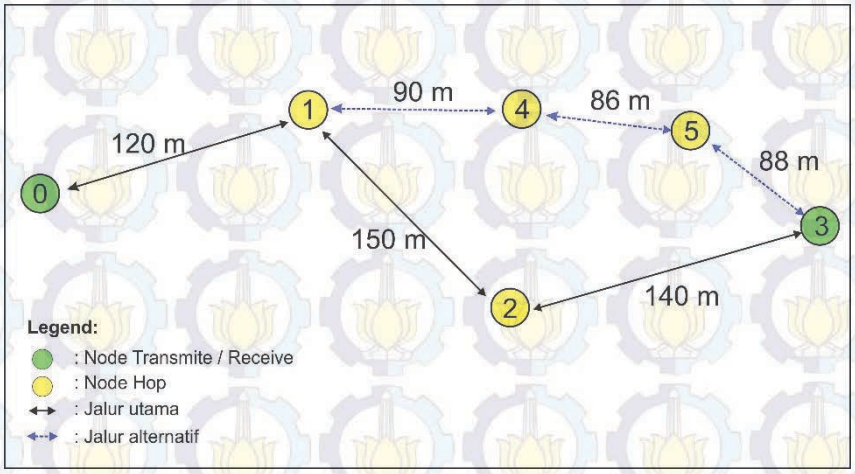


Gambar 3.3 Desain Topologi Simulasi untuk 3 *Hop*

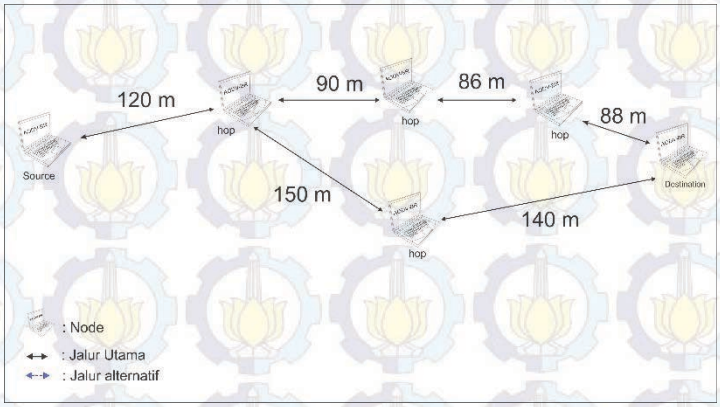


Gambar 3.4 Desain Topologi *Testbed* untuk 3 *Hop*

Pada Gambar 3.3 merupakan desain topologi simulasi untuk 3 *hop*, sedangkan pada Gambar 3.4 desain topologi *testbed* untuk 3 *hop* dimana pada setiap desain memiliki beberapa *node* yang memiliki peran masing-masing. Terdapat *node* yang memiliki peran sebagai pengirim dan *node* sebagai penerima serta beberapa *node* yang dijadikan sebagai loncatan (*hop*). Pada kedua Gambar 3.3 dan 3.4 didefinisikan konfigurasi jarak antar *node* satu dengan lainnya. Desain topologi untuk 3 *hop* dirancang untuk dijadikan pembanding terhadap pengaruh kinerja protokol *routing* dalam sebuah jaringan terhadap perubahan jumlah *node*, yang mana dalam penelitian ini dijadikan sebagai salah satu jenis parameter ukur terhadap kinerja protokol *routing* AODV *backup routing*. Topologi yang direncanakan disesuaikan dengan teknik pengujian yang akan dilakukan terhadap kinerja protokol *routing* AODV-BR untuk komunikasi VMeS.



Gambar 3.5 Desain Topologi Simulasi untuk 4 *Hop*



Gambar 3.6 Desain Topologi *Testbed* untuk 4 *Hop*

Pada Gambar 3.5 merupakan desain topologi simulasi untuk 4 *hop*, sedangkan pada Gambar 3.6 desain topologi *testbed* untuk 4 *hop* dimana pada setiap desain memiliki beberapa *node* yang memiliki peran masing-masing seperti pada desain untuk 3 *hop*. Terdapat *node* yang memiliki peran sebagai pengirim dan *node* sebagai penerima serta beberapa *node* yang dijadikan sebagai loncatan (*hop*). Kedua Gambar 3.5 dan 3.6 tersebut mendefinisikan konfigurasi jarak antar *node* satu dengan lainnya serta model penyusunan *node* jaringan. Desain topologi untuk 4 *hop* dirancang untuk dijadikan pembanding atas 3 *hop* terhadap pengaruh kinerja protokol *routing* dalam sebuah jaringan terhadap perubahan jumlah *node*, yang mana dalam penelitian ini dijadikan sebagai salah satu jenis parameter ukur terhadap kinerja protokol *routing* AODV *backup routing*.

Penelitian ini melakukan simulasi protokol routing dengan jenis topologi jaringan yang ditentukan menggunakan *Network Simulator 2.35* dan merancang testbed menggunakan laptop sebagai pembanding uji kinerja pada simulasi lingkungan ideal dan uji coba kinerja pada kondisi riil VMeS. Alasan utama dibuat dua skenario pengujian pada protokol routing AODV-BR untuk komunikasi VMeS agar sistem dan algoritma *routing* yang direncanakan dapat diadopsi penelitian selanjutnya dalam melakukan impelementasi langsung pada perangkat VMeS yang sebenarnya dengan cara mengacu pada teknik perancangan yang dilakukan pada *testbed*. Pengujian pada simulasi digunakan sebagai langkah awal untuk menguji kinerja protokol *routing* sebelum melakukan impelementasi. Melakukan perbandingan terhadap uji coba simulasi dan *testbed* dilakukan agar dapat

mengetahui kinerja protokol AODV-BR secara konseptual dan secara terapan pada lingkungan nyata.

3.4 Pembuatan Sistem

Pada sub bab ini akan membahas langkah-langlah untuk membuat sistem untuk menganalisa kinerja protokol *routing* AODV-*Backup Routing* terhadap komunikasi VMeS. Terdapat dua langkah yang akan dilakukan, yaitu tahap persiapan terhadap rancangan simulasi dan implementasi *testbed*.

3.4.1 Simulasi

3.4.1.1 Kebutuhan Dasar Simulasi

Dalam proses melakukan simulasi dan analisa kinerja protokol *routing* AODV-BR pada tesis ini dibutuhkan perangkat pendukung yang meluputi:

1. Perangkat keras *(hardware)* yang digunakan.
2. Perangkat lunak *(software)* yang dibutuhkan.

Adapun perangkat keras yang digunakan untuk mensimulasikan AODV-BR ialah menggunakan laptop dengan spesifikasi sebagai berikut:

- Processor : Intel Core i5-2430M 2.4 GHz
- Memory : DDR 3 SDRAM, 4 GB
- VGA : HD Graphics 3000
- HDD : 500 GB

Sedangkan perangkat lunak yang dibutuhkan untuk melakukan simulasi ialah sebagai berikut:

- *Operating System (OS)* yang digunakan adalah Ubuntu 12.04 LTS dengan *code-name Precise Pangolin*.
- *Network Simulator 2.35* sebagai perangkat lunak untuk melakukan simulasi jaringan.
- G-Edit yang akan digunakan untuk *script* editor pada bahasa C dan TCL.
- NAM digunakan untuk menampilkan animasi simulasi.

Selain persiapan secara perangkat, yang perlu dipersiapkan juga adalah skenario dalam perancangan tersebut, adapun hal-hal yang perlu dirancang sebelum melakukan simulasi adalah sebagai berikut:

- Merencanakan simulator objek dan *event-scheduler*
- Merencakanan topologi jaringan yang akan digunakan meliputi node pengirim dan penerima, node yang akan dijadikan loncatan *(hop)* serta rencana *link* antar node.
- Mendefinisikan pola trafik dengan membuat *agent*, *application* dan *flow*.
- Mengatur jalannya skenario simulasi seperti trafik *flow*, *trace* dan *event* lainnya.

3.4.1.2 Konfigurasi dan Instalasi Network Simulator 2 (NS 2.35)

Untuk dapat melakukan simulasi protokol routing AODV *Backup Routing*, yang dilakukan pertama kali adalah melakukan instalasi alat berupa *software* simulator dalam penelitian ini menggunakan *Network Simulator 2.35*. Sebelum melakukan instalasi dan konfigurasi terhadap NS, terdapat beberapa paket yang perlu diinstal agar *network simulator* dapat berjalan dengan sempurna. Adapun paket-paket yang perlu diinstal sebelum melakukan konfigurasi terhadap NS adalah sebagai berikut:

- *tcl8.5-dev, tk8.5-dev*
- *gcc-4.4, g++-4.4*
- *build-essential*
- *autoconf, automake*
- *perl*
- *xgraph*
- *libxt-dev, lib11-dev, dan*
- *libxmu-dev.*

Adapun persiapan serta langkah-langkah instalasi dan konfigurasi terhadap software Network Simulator 2.35 secara detail dipaparkan pada lampiran poin 1 dari buku tesis ini.

3.4.1.3 Konfigurasi Perangkat *Node* pada Simulasi

Node pada simulasi merupakan posisi titik lokasi perangkat komunikasi yang digunakan, baik yang diatur sebagai node *fix* maupun *node mobile* yang dalam penelitian ini merupakan *node* yang dibawa oleh kapal nelayan. Konfigurasi *node* yang akan digunakan pada saat simulasi sudah diatur sesuai dengan skenario pengujian dan kebutuhan sistem. Pengaturan yang disusun dalam simulasi disesuaikan dengan parameter perangkat *testbed* agar dapat dibandingkan hasil kinerja terhadap beberapa parameter ukur. Adapun konfigurasi terhadap parameter *node* pada simulasi adalah pada Tabel 3.1 dibawah ini.

Tabel 3.1 Konfigurasi *Node* pada Simulasi

NO	PARAMETER	VALUE
1	*Channel*	*Wireless Channel*
2	Jenis Propagasi	*TwoRayGround*
3	Antena	*Omni Antenna Directional*
4	Ketinggian Antena	1,5 m
5	Lapisan Fisik	*NetIf*
6	Protokol MAC layer *Datalink*	802.11
7	*Routing* Protokol	AODV-BR / AODV

Pada Tabel 3.1 merupakan konfigurasi perancangan yang dilakukan pada setiap node simulasi. *Channel* merupakan jenis kanal yang digunakan, dalam hal ini diseting menggunakan *wireless* karena jaringannya lebih dinamis. Jenis propagasi yang digunakan adalah *TwoRayGround* dimana setiap *node* dapat saling melihat atau *line of sight (los)* terhadap *node* lainnya atau memantulkan propagasi jika diatas permukaan bumi. Antena yang digunakan adalah *antenna omni directional* dimana pola radiasi yang dipancarkan lebih lebar secara vertikal apabila posisi antena ditegakkan. Ketinggian antena disamakan dengan *testbed* yaitu 1,5 m di atas permukaan bumi. Protokol yang digunakan pada lapisan *datalink* adalah *Medium Access Control* 802.11 sebagai standar yang digunakan oleh Wifi. Protokol *routing* yang digunakan adalah AODV-BR (modifikasi dari AODV) dan AODV tradisional sebagai protokol pembanding kinerja.

3.4.1.4 Konfigurasi Topologi Jaringan pada Simulasi

Pembuatan dan perancangan topologi simulasi mengacu pada Gambar 3.3 dan Gambar 3.5 sebagai patokan perancangan topologi yang direncanakan. Topologi jaringan di *wireless ad hoc network* untuk simulasi yang digunkana pada komunikasi VMeS ini ada dua macam, yaitu *static* dan *dynamic ad hoc network*. Untuk istilah *static ad hoc network* digunakan pada *node* yang tidak bergerak, dalam sistem VMeS digunakan sebagai *gateway* di wilayah darat. Misalnya node_1 (G1) pada awal sampai dengan akhir simulasi berada pada posisi X=10 dan Y=10. Untuk konfigurasi topologi secara keseluruhan terdapat pada Tabel 3.2 berikut.

Tabel 3.2 Konfigurasi *Node* pada Simulasi

NO	PARAMETER	VALUE
1	Jangkauan transmisi max.	150 m
2	Luas topologi	600 x 600 m
3	Jumlah node yang terlibat	5 *node* (3 *hop*) dan 6 *node* (4 *hop*)
4	Kecepatan pergerakan *node*	0.5 m / detik (rata-rata 20 Km/jam)
5	Arah pergerakan	Keluar dari *coverage area*
6	Jumlah *node* yang bergerak	1 *node* bergerak, sisanya statis

Pada Tabel 3.2 dapat dijelaskan parameter yang akan dibuat acuan pengaturan terhadap topologi simulasi. Jangkauan transmisi setiap *node* diatur 150 m, Luas topologi yang digunakan adalah 600 x 600 m, Jumlah node yang terlibat bervariasi tergantung skenario pengujian yang akan dilakukan. Kecepatan pergerakan *node mobile* merupakan kecepatan maksimum mobile node dalam hitungan detik, diseting 0.5 m/detik karena kecepatan kapal laut nelayan dengan ukuran 5 GT kurang lebih 20 km/jam [2]. Arah pergerakan yaitu keluar dari *caverage area* jaringan pada konfigurasi awal posisi *node* yang sudah dibahas pada sub bab 3.3 (Desain Topologi Jaringan). Pada uji coba ini, *node* yang bergerak dengan kecepatan 0.5 m/detik hanyalah satu *node* yang dianggap sebagai kapal nelayan, untuk node lainnya distatiskan. Untuk konfigurasi posisi *node* pada simulasi 3 *hop* dapat dilihat pada Tabel 3.3 dibawah ini.

Tabel 3.3 Konfigurasi *Node* Awal pada Simulasi 3 *hop*

NO	ID Node	Posisi X	Posisi Y	Status
1	Node_(0)	10	10	Statis
2	Node_(1)	158	150	Pindah ke X,Y (158,10)
3	Node_(2)	278	350	Statis
4	Node_(3)	150	290	Statis
5	Node_(4)	246	310	Statis

Pada Tabel 3.3 merupakan konfigurasi posisi *node* yang ditulis dalam *script* TCL yang disusun untuk topologi VMeS. Posisi pada masing-masing *node* ditentukan secara statis dan hanya ada satu *node* yang bergerak. Skenario ini

disusun untuk dapat disesuaikan dengan teknik pengujian AODV *Backup Routing* yaitu dalam teknik memanfaatkan jalur alternatif yang dibuatnya. *Node* yang bergerak yaitu node_(1) yang merupakan *node neighbor*, bergerak dari posisi semula yaitu X (158) dan Y (150) dengan kecepatan 0.5 m/detik menuju posisi X (158) dan Y (10) pada topograpi NS.

Tabel 3.4 Konfigurasi *Node* Awal pada Simulasi 4 *Hop*

NO	ID Node	Posisi X	Posisi Y	Status
1	Node_(0)	10	20	Statis
2	Node_(1)	130	150	Statis
3	Node_(2)	280	320	Pindah ke X,Y (280,30)
4	Node_(3)	420	420	Statis
5	Node_(4)	230	310	Statis
6	Node_(5)	306	330	Statis

Tabel 3.4 merupakan konfigurasi pengaturan terhadap topologi simulasi dengan jarak 4 *hop*. Pada pengaturan tersebut, semua node statis dan hanya node_(2) sebagai *neighbor* / hop dari node_(1) ke node_(3) yang melakukan perpindahan ke arah luar caverage jaringan pada topologi 4 hop. Node_(2) berpindah dari posisi awal X,Y (280, 230) menuju X,Y (280,30) pada topograpi simulasi di Network Simulator 2.35.

3.4.1.5 Konfigurasi Jaringan pada Simulasi

Konfigurasi pada jaringan simulasi digunakan untuk memetakan sistem jaringan yang akan digunakan. Konfigurasi ini meliputi beberapa hal yang memiliki peran dalam seting konfigurasi jaringan, seperti jumlah paket yang akan ditransmisikan, *transport agent* yang digunakan dan lain sebagainya.

Transport agent yang digunakan pada simulasi ini adalah TCP pada sisi pengirim dan TCPSink pada sisi penerima. Aplikasi untuk pengiriman paket adalah *Constant Bit Rate (CBR).* Kapasitas data yang ditransmisikan yaitu sebesar 32, 64, 128 KB sebagai sampel pengiriman paket. Waktu transmisi awal adalah waktu dimana simulai mulai mengirim paket data, konfigurasi yang ditetapkan *node* sender mulai mengirim paket mulai dari milidetik ke 10 sejak dimulainya simulasi dan diakhiri pada milidetik ke 19990 karena total waktu simulasi yang digunakan

20 detik. Adapun konfigurasi yang digunakan pada simulasi sistem protokol *routing* AODV-BR pada komunikasi VMeS ini pada Tabel 3.4 dibawah.

Tabel 3.5 Pengaturan Dasar Jaringan pada Simulasi

NO	PARAMETER	VALUE
1	Transport Agent	TCP/TCPSink
2	Aplikasi Trafik	CBR
3	Kapasitas data	32, 64, 128 KB
4	Waktu awal transmit	10 ms
5	Waktu Akhir Transmit	19990 ms
6	Waktu Simulasi	20 detik

3.4.2 *Testbed*

3.4.2.1 Spesifikasi Peralatan

Dalam proses penelitian untuk menganalisa kinerja dari protokol *routing* AODV-BR secara simulasi pada sistem komunikasi VMeS ini, diperlukan beberapa perangkat pendukung, yaitu berupa perangkat keras dan perangkat lunak. Perangkat keras yang digunakan untuk melakukan analisa dan implementasi protokol *routing* AODV-BR ini membutuhkan beberapa *node*. Setiap *node testbed* dianggap memiliki performasi perangkat nirkabel yang sama, yaitu menggunakan USB Dongle TP-Link TL-WN722N IEEE 802.11 n seperti pada Gambar 3.7.

Gambar 3.7 USB Dongle TP-Link TL-WN722N

Adapun spesifikasi dari USB Dongle TP-Link TL-WN722N tersebut adalah sebagai berikut:

Tabel 3.6 Spesifikasi *Wireless Adapter* yang Digunakan

NO	FITUR	VALUE
1	Port	USB 2.0
2	Dimensi (W x D x H)	3.7 x 1.0 x 0.4 in. (93.5 x 26 x 11mm)
3	Tipe Antena	Detachable Omni Directional (RP-SMA)
4	Gain Antena	4dBi
5	Frekuensi	2.400-2.4835GHz
6	Standar Wireless	IEEE 802.11n, IEEE 802.11g, IEEE 802.11b
7	Jangkauan Area Los	Up to 200 m

Sedangkan *node-node testbed* yang digunakan berupa laptop yang memiliki spesifikasi sebagai berikut:

Tabel 3.7 Spesifikasi *Hardware* yang Digunakan

NODE	PROCESSOR	RAM	OS
1	Intel Core i5-2430M CPU @ 2.40GHz	4 GB	Linux Kernel 2.6.32 (Ubuntu 10.04.1)
2	Intel Atom CPU Z520 @ 1.33GHz	2 GB	Linux Kernel 2.6.32 (Ubuntu 10.04.1)
3	Intel Core 2 Duo CPU E7500 @ 2.93GHz	2 GB	Linux Kernel 2.6.32 (Ubuntu 10.04.1)
4	Intel ® core 2 duo T6600	2 GB	Linux Kernel 2.6.32 (Ubuntu 10.04.1)
5	Intel Core Dual Core, 2.1GHz	2 GB	Linux Kernel 2.6.32 (Ubuntu 10.04.1)
6	Intel atom CPU 2.1 GHz	1 GB	Linux Kernel 2.6.32 (Ubuntu 10.04.1)

Setelah kebutuhan *hardware* sudah terpenuhi, dibutuhkan perangkat lunak untuk menjalankan sistem yang diinginkan, yaitu untuk menganalisa kinerja protokol *routing* AODV-BR pada VMeS. Terdapat beberapa *software* yang akan digunakan, yaitu *source* AODV yang digunakan adalah *aodv-uu-0.9.6.orig.tar.gz*. Selain beberapa kebutuhan *software* utama, terdapat beberapa *software* tambahan yang dibutuhkan yaitu seperti *cpp-4.4, dpkg-dev, fakeroot, g++, libgcc, iptable*

1.36. untuk *tool* pengukuran performasinya jaringan menggunakan *iperf* versi 2.0.2 yang bekerja dibawah terminal pada linux.

3.4.2.2 Instalasi dan Konfigurasi AODV-UU

Untuk melakukan instalasi dan konfigurasi AODV-UU pada operating system Linux dengan kernel 2.6.32 dibutuhkan *source AODV-UU* dengan versi 0.9.6 dan melakukan *updating* terhadap *repository* yang digunakan sebelum melakukan instalasi. Melakukan instalasi beberapa paket *software* yang dibutuhkan sebelum melakukan *compailing* terhadap *source* AODV seperti *gcc, lib gcc*, dll. Melakukan *update* terhadap *header kernel* yang dibutuhkan. Untuk melakukan beberapa hal yang disebutkan tadi, berikut langkah-langkah yang digunakan untuk melakukan konfigurasi dan instalasi AODV-UU.

- Gunakan perintah *apt-get install build-essential* pada terminal untuk melakukan instalasi terhadap beberapa paket *software* yang dibutuhkan sebelum instalasi dan konfigurasi AODV-UU seperti *gcc, lib gcc*, dll.
- Lakukan instalasi linux header seperti yang digunakan pada penelitian ini yaitu *linux-headers-2.6.32-24-generic* dengan mengeksekusi perintah ini melalui terminal *apt-get install linux-headers-$(uname –r)*.
- Lakukan ektraksi pada *aodv-uu-0.9.6.orig.tar.gz.* kemudian arahkan posisi direktori pada terminal kedalam *folder* aodv-uu yang sudah diekstrak.
- Gunakan perintah *make* pada terminal untuk melakukan *compailing* pada paket AODV-UU yang sudah diekstrak tadi.
- Jika tidak terjadi *error*, maka proses *compailing* berhasil dilakukan. Kemudian gunakan perintah *make install* untuk melakukan instalasi paket AODV-UU yang sudah berhasil di-*compail*.

3.4.2.3 Instalasi dan Konfigurasi Alat Ukur

Iperf merupakan *tool* performasi pada sebuah jaringan, difungsikan untuk menguji coba pengiriman paket dengan ketentuan kapasitas tertentu. *Iperf* pada penelitian ini ditugaskan untuk menangani kebutuhan pada lapisan aplikasi yaitu digunakan sebagai sarana untuk melakukan perintah mengirim dan menerima paket antar *node*, dan menghitung performa dari jaringan yang digunakan. Untuk

melakukan instalasi dan konfigurasi terhadap *iperf* dilakukan dengan beberapa cara berikut:

- Lakukan instalasi via terminal dengan mengetikkan perintah *apt-get install iperf* dan biarkan OS melakukan instalasi secara *online*.
- Untuk melakukan konfigurasi *iperf* dalam penelitian ini, dilakukan seting yang berbeda antara sisi node pengirim paket dan sisi *node* penerima. Pada sisi penerima, *iperf* distatuskan sebagai penerima dengan mengetikkan perintah:

```
iperf -s
```

- Sedangkan untuk sisi pengirim, perintah yang dieksekusi memiliki format *iperf -c [ip tujuan] –n [kapasitas data yang hendak dikirim misalnya 128K] –I 0.1*, contoh perintah yang digunakan pada terminal:

```
Iperf –c 10.10.10.4 –n 128K –i 0.1
```

3.4.2.4 Konfigurasi Jaringan pada *Testbed*

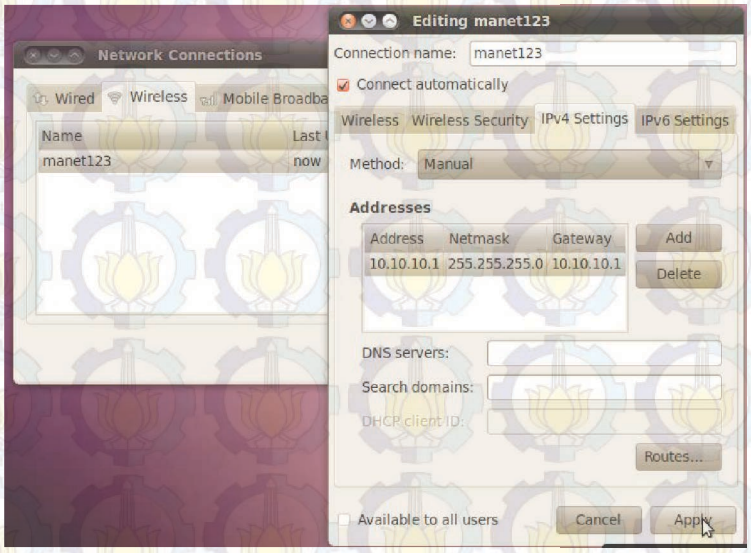
Setting network dalam sistem ini dibutuhkan untuk mengidentitaskan masing-masing *node* agar dapat dikenali oleh *node* yang memiliki pengaturan yang sama. *Setting network* yang dimaksud meliputi; seting IP, seting SSID jaringan yang digunakan, dan melakukan pancaran jaringan *ad hoc* pada masing-masing *node*.

Untuk melakukan pengaturan-pengaturan tersebut, dibutuhkan beberapa langkah sebagai berikut:

- Langkah awal adalah memastikan bahwa OS sudah mendeteksi *hardware* jaringan digunakan, untuk penelitian ini menggunakan *hardware wireless*.
- Membuat SSID dalam penelitian ini SSID yang digunakan adalah “manet123” dan memancarkan sinyal *ad hoc* dari masing-masing *node* dengan cara klik kanan pada *toolbar* jaringan di taskbar OS Ubuntu, kemudian klik *create new wireless*.

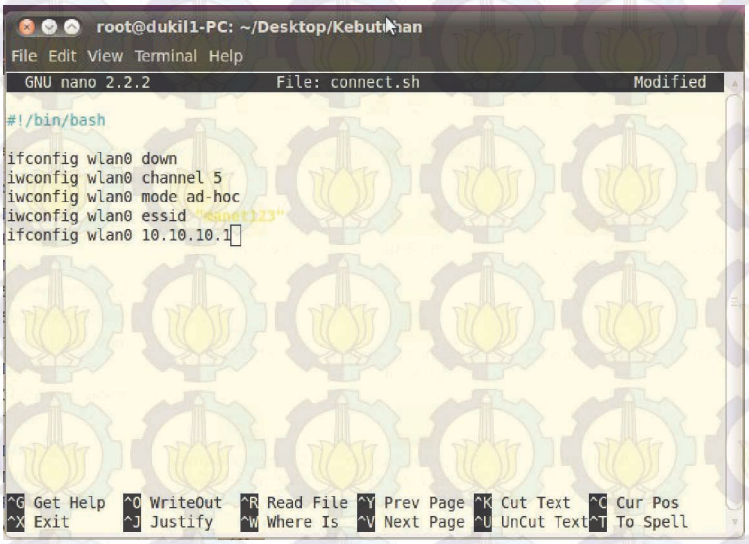
Gambar 3.8 Pembuatan SSID *Ad Hoc*

- Lakukan seting IP dan *subnetmask* yang sekelas, agar saat sebuah node tergabung dalam jaringan yang sama, langsung terkonfigurasi dan terkoneksi secara otomatis.



Gambar 3.9 *Seting IP* pada Jaringan *Ad Hoc* yang Digunakan

Untuk beberapa versi OS lain dari linux seperti Debian (etch), untuk melakukan konfigurasi ini dapat dilakukan dengan sekali cara, yaitu dengan membuat *file.sh* yang nantinya akan dieksekusi satu kali dan otomatis semua konfigurasi seperti pembuatan SSID jaringan *ad hoc* dan seting IP akan berubah dengan satu kali langkah, berikut tampilan *file* yang berisi *script* dan perintah yang digunakan untuk langkah ini:



Gambar 3.10 *Script Config Wireless Ad Hoc* Menggunakan *File.sh*

Adapun seting dan konfigurasi yang dilakukan pada setiap *node* yang digunakan untuk analisa kinerja protokol routing AODV *backup routing* pada *testbed* adalah pada Tabel 3.8 berikut:

Tabel 3.8 Konfigurasi *Node* pada *Testbed*

NODE	IP	SSID	MODE
1	10.10.10.1	manet123	AdHoc
2	10.10.10.2	manet123	AdHoc
3	10.10.10.3	manet123	AdHoc
4	10.10.10.4	manet123	AdHoc
5	10.10.10.5	manet123	AdHoc
6	10.10.10.6	manet123	AdHoc

3.5 Skenario Pengujian Sistem

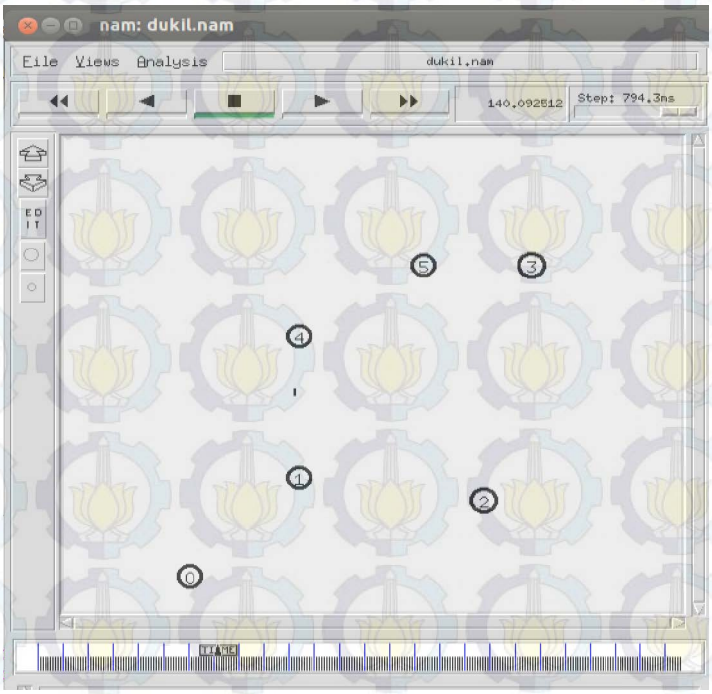
Skenario pengujian merupakan langkah-langkah yang akan dilakukan untuk menguji kinerja dari protokol routing yang ditentukan. Skenario pengujian ini meliputi pengujian pada saat simulasi dan pengujian pada *testbed*. Adapun skenario dibagi menjadi sebagai berikut:

3.5.1 Simulasi

3.5.1.1 Pengujian Simulasi pada NS

Pada proses pengujian simulasi pada NS, digunakan bahasa pemprograman TCL untuk pengaturan topologi jaringan, data yang dikirim, dan beberapa ketentuan lain yang tentunya disesuaikan dengan sistem yang diinginkan untuk

bahan pengujian kinerja dari algoritma protokol AODV-BR pada komuinkasi VMeS dapat berjalan dengan baik, sehingga dari hasil simulasi diharapkan dapat menarik kesimpulan tentang kinerja dari protokol *routing* AODV-BR pada komunikasi VMeS.



Gambar 3.11 Pengujian simulasi yang ditampilkan dalam file NAM

3.5.1.2 Pengukuran Parameter QoS

Pada pengujian *Quality of Service (QoS)* pada jaringan *ad hoc* yang dibuat untuk komunikasi VMeS, perlu ditentukan parameter yang akan diukur untuk menganalisa sebuh kinerja dari protokol routing AODV-BR. Parameter-parameter yang dibutuhkan untuk analisa yaitu *delay* dan *throughput*. Untuk mendapatkan nilai dari hasil kinerja pada saat simulasi, digunakan *script awk* untuk menparsing beberapa data yang terdapat pada *file trace* di akhir simulasi.

3.5.2 *Testbed*

3.5.2.1 Lokasi Pengujian

Lokasi pengujian terhadap testbed yang sudah berhasil dijadikan *router* dalam *ad hoc* VMeS ini diuji coba di salah satu pantai Madura berdekatan dengan lokasi jembatan Suramadu. Lokasi diasumsikan *line of sight* tanpa adanya penghalang (*obstacle* yang tentunya memiliki pengaruh besar) karena antar *node*

diseting saling terhubung langsung dengan jarak bervariasi seperti yang sudah diskenariokan. Pengujian diharuskan *los* karena sistem komunikasi VMeS diterapkan pada wilayah laut yang sudah dipastikan antar node tidak terdapat penghalang apapun. Adapun denah lokasi pengukuran dan posisi node ditunjukkan pada Gambar 3.12 dan Gambar 3.13 berikut:

Gambar 3.12 Lokasi Pengukuran dan Konfigurasi Posisi 3 *Hop*

Gambar 3.13 Lokasi Pengukuran dan Konfigurasi Posisi 4 *Hop*

3.5.2.2 Pengujian Koneksi antar *Node*

Sebelum melakukan uji kemampuan *routing* pada *node* yang sudah ditanamkan didalamnya routing AODV-BR, terlebih dahulu perlu diuji konektivitas tanpa menjalankan *routing*. Untuk melakukan uji konektivitas pada setiap *node*, *node* dipastikan secara bersama memancarkan sinyal *ad hoc* dengan SSID yang sama dan IP *address* yang berbeda. Setelah saling terkoneksi dengan jaringan yang dipancarkan satu *node* dengan lainnya, maka lakukan PING di terminal terhadap IP node yang sedang terkoneksi. Perintah yang dieksekusi melalui terminal misalnya hendak melakukan PING pada IP 10.10.10.6 adalah seperti

```
Ping 10.10.10.6
```

Dalam penelitian ini, PING juga difungsikan sebagai cara untuk konfigurasi posisi masing-masing *node* dan pengambilan data. Dengan cara menjalankan perintah ping terhadap *node* tetangga kemudian menggeser *node* hingga ujung *caverage area*. Untuk mengetahui ujung *caverage area* masing-masing dari jangkauan transmisi node bisa diketahui dari proses ping yang awalnya berjalan menghitung *seq*, *ttl*, dan *time* berubah status menjadi *destination host unreachable*.

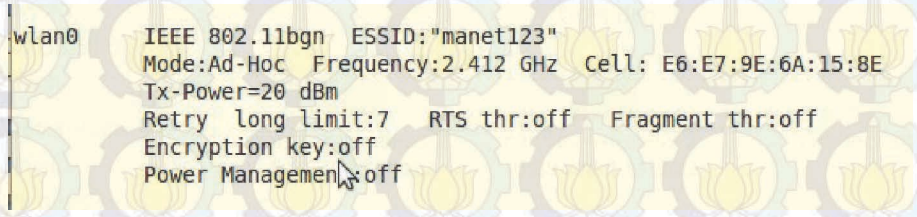
3.5.2.3 Pengujian jaringan yang digunakan

Untuk melakukan seting dan konfigurasi terhadap jaringan yang digunakan, perlu dikoreksi terlebih dahulu pengaturan jaringan yang hendak digunakan. Uji coba ini, hanya dibutuhkan untuk memastikan agar tidak terjadi kesalahan seperti perbedaan SSID jaringan, dan memastikan untuk tidak terjadinya IP ganda pada *node* yang terhubung.

Untuk melakukan *cheking* terhadap status seting dan konfigurasi jaringan yang sudah dilakukan, ketikkan perintah berikut pada terminal:

```
iwconfig
```

Maka anda dapat mengkoreksi status dari seting dan konfigurasi jaringan yang hendak digunakan, seperti pada Gambar 3.14 berikut:



Gambar 3.14 Seting Konfigurasi *Wireless* yang Digunakan

3.5.2.4 Pengujian protokol *routing* AODV-BR

Pengujian kinerja protokol *routing* AODV-BR dilakukan untuk mengetahui kinerja dari protokol ini. Pengujian dilakukan dengan cara mengecek jalur dari *node sender* ke *node destination* setelah protokol *routing* dijalankan. Untuk menjalankan routing AODV-BR ini dapat dilakukan melalui terminal dengan perintah.

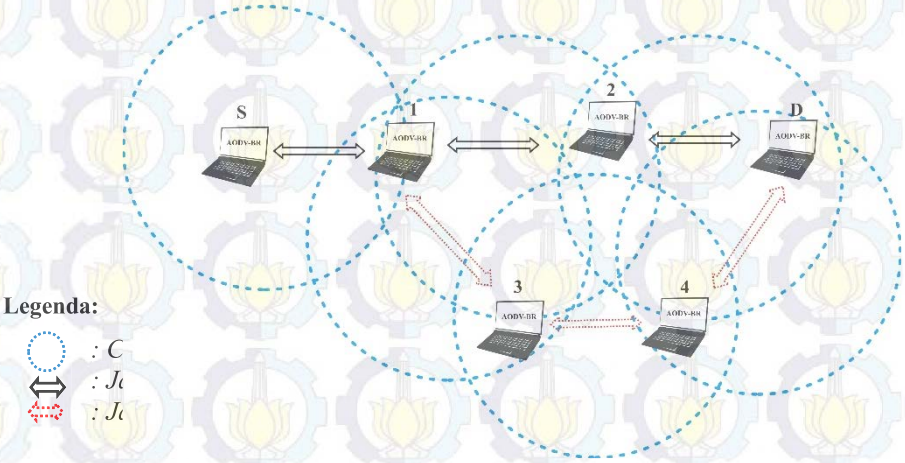
```
aodvd -D
```

Setelah memastikan setiap *node* yang tergabung dalam jaringan *ad hoc* aplikasi protokol routingnya jalan secara keseluruhan. Langkah selanjutnya untuk melakukan uji coba apakah jalur sudah terbentuk atau tidak, maka lakukan dengan cara *thresould* terhadap IP tujuan atau lakukan dengan cara PING dengan ekstensi –R terhadap IP tujuan, maka akan tampak beberapa jalur berupa identitas IP perangkat yang akan dilalui untuk menuju ke *node* destinasi.

3.5.2.5 Skenario Pengujian pada *Testbed*

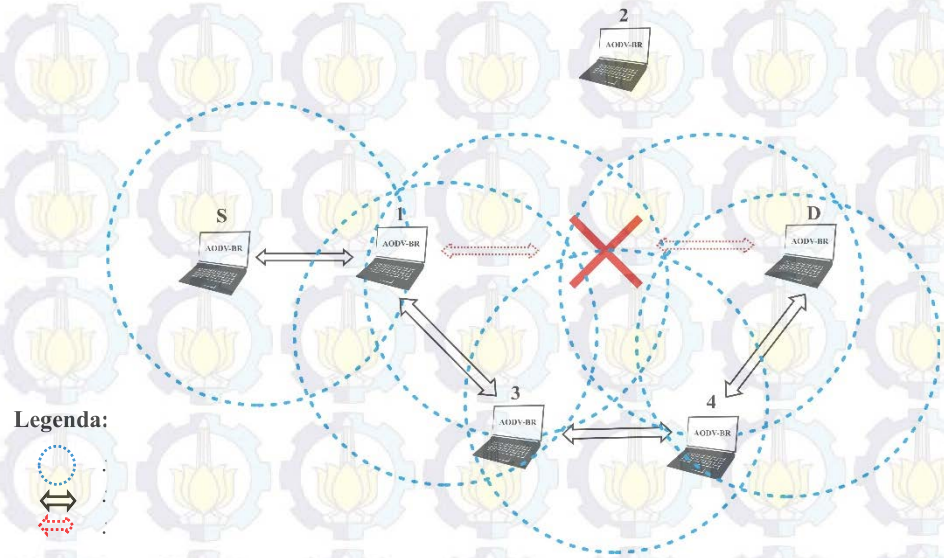
Skenario pengujian meliputi topologi yang digunakan saat uji coba protokol *routing* AODV-BR pada *node* yang berupa laptop. Posisi antar *node* dalam penelitian ini diasumsikan berada pada ujung *caverage* masing-masing *node* yang lain, artinya tidak diperbolehkan dalam satu *caverage* memiliki dua jangkaun terhadap *node* yang dijadikan *neighbor* kecuali *node* yang menjadi *hop* sebelumnya dan *hop* setelahnya. Misalkan dalam sebuah jaringan terdapat tiga *node* yaitu node_1, node_2, dan node_3. Node_1 hanya dapat menjangkau node_2 dan tidak boleh menjangkau node_3. Sedangkan node_2 yang menjadi *hop* dari *node* node_1 diharuskan menjangkau node_1 dan node_3. Node_3 hanya diperbolehkan menjangkau node 2. Alasan diskenariokan seperti ini, dikarenakan konsep protokol *routing* yang digunakan adalah protokol *routing* AODV dimana *routing algoritm* yang diadopsi oleh AODV adalah *routing* yang berdasarkan jarak *hop count* dan *sequence number*. Pada prakteknya, AODV tidak dapat menjadikan node *hop*

kendatipun posisi *node* sangat dekat jika node *destination* yang diinginkan masih dalam *coverage area node sender*. *Node sender* tidak akan memilih *node hop* melainkan langsung men-*direct* ke *destination*. Jadi, sebuah *node* diskenariokan tidak diperbolehkan untuk mencakup lebih dari satu node dalam *coverage area* yang dimilikinya, kecuali *node hop* yang mencakup dua *node* pada posisi kanan dan kirinya. Adapun topologi yang digunakan pada skenario pengujian kinerja protokol *routing* AODV-BR pada implementasi keperangkat ini adalah diitunjukkan pada Gambar 3.15 berikut:



Gambar 3.15 Topologi Pengujian Sistem *Routing* dengan 4 *Hop*

Pada Gambar 3.15 merupakan skenario pengujian yang dilakukan untuk mengetahui kinerja dari protokol routing AODV-BR, dimana masing-masing *node* hanya dapat menjangkau satu *node* yang akan dijadikan *next hop* atau *neighbor*. *Node* diletakkan sesuai skenario untuk menguji *backup routing* dari AODV-BR. Skenario pengujian yang akan dilakukan adalah pada saat *node* melakukan pengiriman data dari *node* ujung ke *node* penerima yang mana pada Gambar 3.15 *node* sumber ditandai dengan (S) dan node penerima ditandai dengan (D) salah satu *node* yang dijadikan *hop* dari pengirim ke penerima akan dipindahkan untuk keluar dari *coverage area* atau dengan cara mematikan *node* untuk menguji jalur alternatif yang dimiliki oleh AODV-BR.



Gambar 3.16 Kondisi Jalur pada AODV-BR Saat *Node* Dikeluarkan dari *Caverage Area*

Pada Gambar 3.16 merupakan skenario pengujian mengeluarkan satu *hop* yang dijadikan jalur utama oleh AODV-BR *caverage area* node yang lain agar AODV-BR dapat melakukan *switch* untuk merubah jalur yang akan dilalui paket ke jalur alternatif tanpa melakukan *route discovery* kembali.

3.6 Proses Penyimpulan Hasil Penelitian

Dari seluruh proses yang dilakukan pada penelitian ini yang meliputi tahap perancangan, implementasi dan pengujian, akan didapatkan beberapa kesimpulan yang dapat diambil. Kesimpulan diambil berdasarkan analisa data dengan beberapa parameter kinerja dari protokol *routing* AODV-BR secara keseluruhan.

Analisa kinerja yang akan dilakukan adalah berdasarkan nilai yang dihasilkan dari pengukuran performasi jaringan meliputi parameter *delay* menggunakan persamaan 2.1 dan *throughput* pada jaringan menggunakan persamaan 2.2 dengan protokol *routing* yang sudah ditentukan, yaitu AODV *Backup Routing* dan AODV tradisional sebagai protokol pembanding.

BAB 4
HASIL DAN ANALISA DATA

Pada bab ini akan dibahas mengenai hasil dari simulasi AOV-BR pada NS dan *testbed* menggunakan laptop. Pada akhir simulasi, diperoleh *file trace* yang mencatat seluruh aktivitas kejadian selama simulasi berlangsung. *File trace* ini, akan dijadikan bahan dasar untuk menfilter data dengan melakukan parsing menggunakan *script awk*. Pada simulasi jumlah *node* yang diuji coba sama halnya dengan yang digunakan saat implementasi pada alat. Sedangkan untuk *testbed*, teknik pengambilan data menggunakan *tool iperf* yang dapat menguji performasi jaringan dengan protokol *routing* yang sudah ditentukan, adapun jumlah *node* yang digunakan untuk *testbed* adalah 6 *node* dikarenakan keterbatasan alat saat melakukan uji coba *testbed*.

Protokol *Medium Access Control (MAC)* pada lapisan *datalink* yang digunakan VMeS, semestinya adalah protokol AX.25 sebagai protokol radio disebabkan prangkat pada lapisan fisik yang diinginkan harus mendukung frekuensi yang digunakan oleh komunikasi VMeS, yaitu HF dan VHF. Namun pada penelitian ini, masih menggunakan lapisan *Datalink* dan Fisik 802.11 baik pada sisi simulasi maupun implementasi testbed, dikarekan *tool* simulasi yang digunakan yaitu *Network Simulator 2.35* belum mendukung protokol *Medium Access Control (MAC)* AX.25 yang digunakan sebagai protokol MAC pada VMeS. Untuk uji coba kinerja protokol *routing* terhadap perangkat juga memanfaatkan *Medium Access Control (MAC)* 802.11 disebabkan perangkat yang mendukung protokol AX.25 masih dalam tahap pengembangan.

Sebagai pembanding terhadap analisa kinerja dari AODV-BR peneliti juga melakukan simulasi dan uji coba menggunakan AODV tradisional yang masih belum memiliki *multipath routing*. Parameter yang digunakan untuk analisa terhadap kinerja AODV-BR adalah menguji performa terhadap *QoS* jaringannya, yaitu *delay* yang dibutuhkan dan *throughput* yang dihasilkan, adapun parameter pengujian ditunjukkan pada Tabel 4.1 dibawah.

Tabel 4.1 Parameter Uji Coba untuk Analisa

PARAMETER UJI	SIMULASI	TESTBED
Jarak 3 Hop	√	√
Jarak 4 Hop	√	√
Throughput	√	√
Delay	√	√
Variasi Data	√	√

4.1 Hasil Pengujian Simulasi

Analisa pada hasil simulasi dilakukan untuk mengetahui performa *routing* dari *AODV-Backup Routing* dengan AODV berdasarkan parameter *Quality of Service (QoS)* pada jaringan, dalam penelitian ini parameter yang digunakan untuk dianalisa berdasarkan pada hasil *delay* dan *throughput* yang dihasilkan terhadap jumlah hop yang terlibat pada jaringan berdasarkan variasi data yang ditentukan, yaitu 32 KB, 64 KB, 128 KB.

4.1.1 *Delay*

Delay merupakan interval atau selisih waktu yang perlukan untuk pengiriman sebuah paket dari *node* pengirim ke *node* penerima. Pengukuran terhadap *delay* yaitu membandingkan waktu yang dibutuhkan oleh AODV-BR dengan AODV dalam sekenario pengukuran yang sama. Hasil simulasi menegenai parameter *delay* ini berdasarkan jumlah *hop* yang digunakan saat simulasi adalah sebagai berikut:

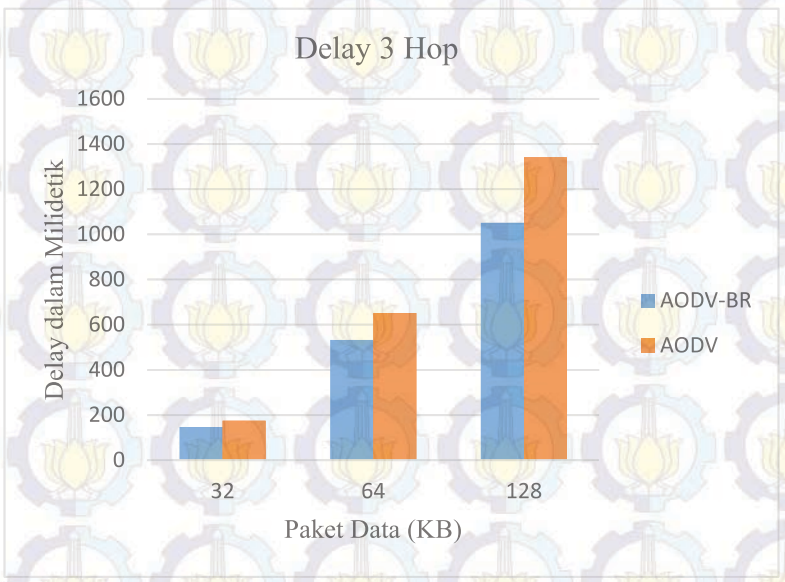
4.1.1.1 *Delay* untuk 3 hop

Untuk pengukuran *delay* yang dibutuhkan dalam pengiriman paket dengan jumlah 4 *hop*, dimana terdapat 4 *node* yang akan dilalui oleh paket data, yaitu dikirim dari *node* 1 ke *node* 4 atau sebaliknya yang melalui *node* 3 dan *node* 2 sebagai *intermediate node*. Data yang diukur dibandingkan antara AODV-*backup routing* dengan AODV tradisional dengan pengambilan sampel besaran data yang bervariasi yaitu 32KB, 64 KB, dan 128 KB.

Tabel 4.2 Delay yang Dibutuhkan untuk 3 *Hop* AODV-BR dan AODV

Perc. Ke	32 KB		64 KB		128 KB	
	AODV-BR	AODV	AODV-BR	AODV	AODV-BR	AODV
1	142	172	523	642	1077	1301
2	147	181	510	644	1067	1300
3	148	169	532	650	1000	1376
4	151	175	541	669	1010	1389
5	152	168	513	641	1045	1320
6	151	177	545	661	1002	1334
7	134	181	525	654	1082	1308
8	144	180	511	640	1100	1340
9	150	171	521	653	1092	1350
10	150	174	610	659	1120	1398
Rata-rata	**147**	**175**	**533**	**651**	**1051**	**1342**

Pada Tabel 4.2 merupakan hasil simulasi terhadap *delay* yang dibutuhkan untuk simulasi 3 *hop* pada protokol *routing* AODV-BR dan AODV tradisional. Perbedaan *delay* yang dibutuhkan antara AODV-BR dan AODV dengan besaran paket data yang bervariasi yaitu 32 KB, 64 KB, dan 128 KB dari 10 kali percobaan pada skenario yang sama, dapat disimpulkan kedalam hasil rata-rata dari sekian percobaan. Untuk variasi paket data 32 KB, rata-rata AODV-BR membutuhkan waktu pengiriman paket 147 milidetik sedangkan AODV 175 milidetik. Untuk 64 KB, AODV-BR membutuhkan 533 milidetik sedangkan AODV 651 milidetik. Untuk 128 KB, rata-rata AODV-BR membutuhkan 1051 milidetik sedangkan AODV membutuhkan 1342 milidetik. Untuk memudahkan menganalisa hasil, hasil data rata-rata yang diperoleh dapat diplot ke dalam grafik. Grafik rata-rata dari waktu yang dibutuhkan dalam pengiriman paket oleh AODV-BR dan AODV untuk 3 *hop* dapat dilihat pada Gambar 4.1 dibawah.



Gambar 4.1 Perbadingan Rata-rata *Delay* 3 *Hop*

Berdasarkan Gambar 4.1 diatas, terlihat perbedaan dalam kebutuhan *delay* terhadap variasi data yang dikirim terlihat begitu signifikan. AODV-BR membuktikan kebutuhan *delay* yang lebih rendah disetiap macam variasi data yang dikirim. Hal ini dikarenakan AODV-BR tidak membutuhkan *route discovery* kembali disaat terjadi *route break* melainkan secara otomatis memanfaatkan *routing* alternatif yang dibuat saat proses *setup route.* Sedangkan AODV membutuhkan *route discovery* dan *setup route* dari awal dikarenakan tidak adanya jalur alternatif dalam AODV. Dari jumlah 3 *hop* ini, estimasi waktu yang dibutuhkan AODV-BR lebih sedikit dari pada AODV tradisional.

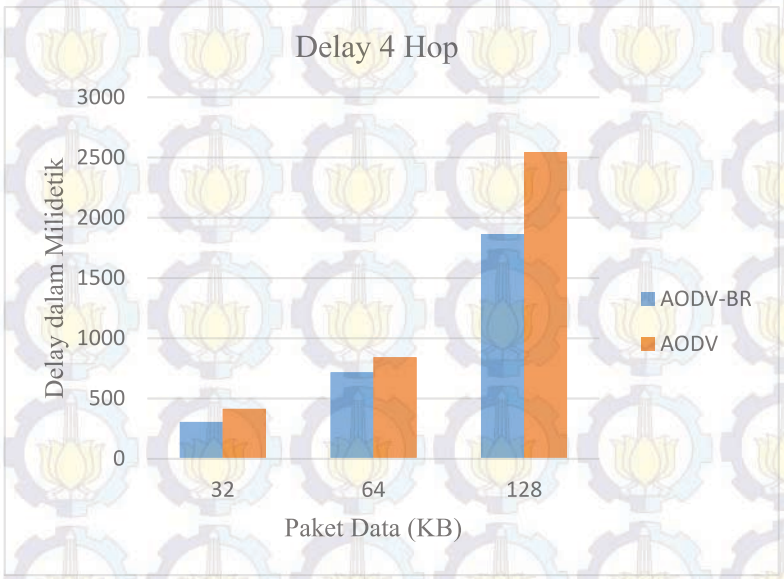
4.1.1.2 *Delay* untuk 4 *Hop*

Untuk pengukuran *delay* yang dibutuhkan dalam pengiriman paket dengan jumlah 4 *hop*, dimana terdapat 5 *node* yang akan dilalui oleh paket data, yaitu dikirim dari *node* 1 ke *node* 5 atau sebaliknya yang melalui *node* 4, *node* 3, node 2 sebagai *intermediate node*. Data yang diukur dibandingkan antara AODV-*Backup Routing* dengan AODV tradisional dengan pengambilan sampel besaran data yang bervariasi yaitu 32KB, 64 KB, dan 128 KB.

Tabel 4.3 *Delay* yang Dibutuhkan untuk 4 *Hop* AODV-BR dan AODV

Perc. Ke	32 KB		64 KB		128 KB	
	AODV-BR	AODV	AODV-BR	AODV	AODV-BR	AODV
1	200	390	661	812	1800	2510
2	232	395	665	822	1867	2598
3	220	399	687	831	1851	2577
4	312	407	691	832	1899	2501
5	323	413	703	833	1910	2489
6	331	417	714	843	1934	2563
7	335	422	737	851	1854	2570
8	351	427	765	852	1881	2562
9	376	431	773	876	1841	2542
10	382	436	782	881	1812	2550
Rata-rata	**306**	**414**	**718**	**843**	**1865**	**2546**

Pada Tabel 4.3 merupakan hasil simulasi terhadap *delay* yang dibutuhkan untuk simulasi 4 *hop* pada protokol *routing* AODV-BR dan AODV tradisional. Perbedaan rata-rata *delay* yang dibutuhkan antara AODV-BR dan AODV dengan besaran paket data yang ditentukan yaitu 32 KB, 64 KB, dan 128 KB dari 10 kali percobaan pada skenario pengujian yang sama, dapat diambil nilai rata-rata dari sekian percobaan untuk memudahkan dalam menyimpulkan hasil. Untuk variasi paket data 32 KB, rata-rata AODV-BR membutuhkan waktu pengiriman paket 306 milidetik sedangkan AODV 414 milidetik. Untuk 64 KB, AODV-BR membutuhkan 718 milidetik sedangkan AODV 843 milidetik. Untuk 128 KB, rata-rata AODV-BR membutuhkan 1865 milidetik sedangkan AODV membutuhkan 2546 milidetik. Untuk memudahkan proses analisa, hasil data rata-rata yang diperoleh dapat diplot ke dalam grafik. Grafik rata-rata dari waktu yang dibutuhkan dalam pengiriman paket oleh AODV-BR dan AODV untuk 5 *hop* dapat dilihat pada Gambar 4.2 dibawah.



Gambar 4.2 Perbadingan Rata-rata *Delay* 4 *Hop*

Berdasarkan Gambar 4.2 diatas, terlihat perbedaan dalam kebutuhan *delay* terhadap variasi data yang dikirim terlihat begitu signifikan. AODV-BR membuktikan kebutuhan *delay* yang lebih rendah disetiap macam variasi data yang dikirim. Hal ini dikarenakan AODV-BR tidak membutuhkan *setup route* dari awal disaat terjadi *route break* karana AODV-BR menyediakan alternatif *route*, sedangkan AODV membutuhkan *route discovery* dan *setup route* dari awal dikarenakan tidak adanya jalur arternatif dalam AODV. Dari jumlah 5 *hop* ini, estimasi waktu yang dibutuhkan AODV-BR lebih sedikit dari pada AODV tradisional.

4.1.2 *Throughput*

Throughput merupakan jumlah data yang sukses terkirim dalam ukuran detik melalui sebuah sistem atau media komunikasi. Untuk hasil simulasi, *throughput* didapatkan dari hasil *parsing* data pada akhir simulasi. Pengukuran *throughput* dilakukan untuk mengetahui kinerja dari protokol *routing* yang digunakan. *Throughput* berbanding lurus dengan *delay* yang dibutuhkan untuk pengiriman paket. Pengukuran terhadap *throughput* ini untuk mengetahui performa *routing* dari AODV-BR dan AODV tradisional yang tentunya dalam skenario

pengujian yang sama. Perbandingan untuk menganalisa *throughput* yang didapatkan berdasarkan jumlah *hop* yang digunakan waktu simulasi.

4.1.2.1 *Throughput* untuk 3 hop

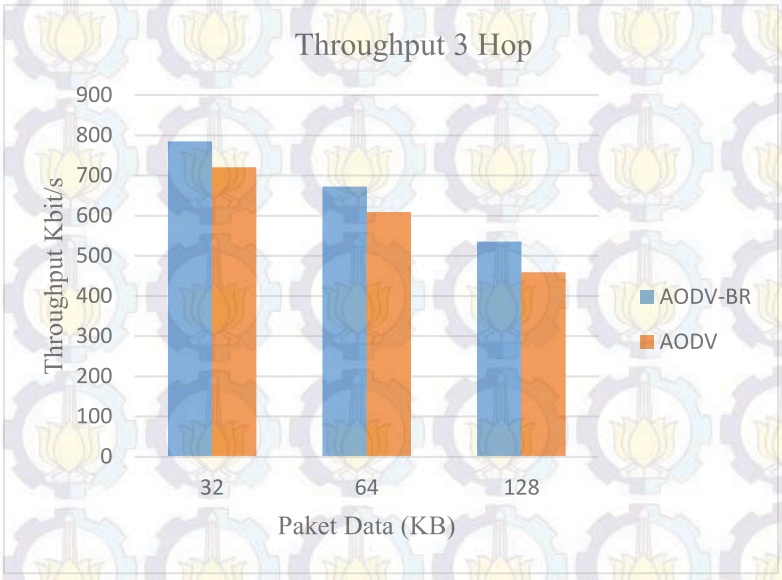
Pengukuran terhadap *throughput* paket per-detiknya dibutuhkan untuk menganalisa performa dari sebuah jaringan. Pada pengukuran 3 hop, terdapat 4 node yang akan dilalui data. Pada simulasi, diasumsikan semua *node* memiliki performa yang sama karena dalam kondisi ideal. Semakin tinggi *delay* yang dibutuhkan, otomatis nilai *throughput* akan semakin rendah. Untuk besaran variasi data yang digunakan dalam mengukur *throughput* sama besarnya saat pengukuran *delay* yaitu 32 KB, 64 KB, dan 128 KB. *Throughput* dan *delay* didapat secara bersamaan dalam setiap kali uji coba.

Tabel 4.4 *Throughput* yang Didapatkan untuk Simulasi 3 *Hop* AODV-BR dan AODV

Perc. Ke	32 KB		64 KB		128 KB	
	AODV-BR	AODV	AODV-BR	AODV	AODV-BR	AODV
1	800	729	663	640	554	467
2	792	701	654	613	556	462
3	791	731	693	616	571	452
4	768	726	651	598	518	443
5	767	732	678	596	531	480
6	785	724	667	613	532	448
7	807	701	676	615	551	450
8	798	704	699	623	508	465
9	770	730	650	582	524	463
10	770	726	690	590	501	461
Rata-rata	**785**	**720**	**672**	**609**	**535**	**459**

Pada Tabel 4.4 merupakan hasil pengukuran dalam simulasi terhadap *throughput* yang dihasilkan sesuai dengan *delay* pengiriman paket yang dibutuhkan. Rata-rata *throughput* yang dihasilkan pada pengujian simulasi 3 *hop* terhadap protokol routing AODV-BR memiliki nilai *throughput* yang lebih tinggi dari pada AODV tradisional. Hal ini dipicu oleh faktor *delay* yang dibutuhkan AODV saat terjadi *route break* pada waktu pengiriman paket data berlangsung.

Dari sekian data hasil pengujian simulasi yang bervariasi, untuk meudahkan dalam penyimpulan dibutuhkan nilai rata-rata terhadap *throughput* yang dihasilkan. Untuk variasi paket 32 KB, AODV-BR menghasilkan *throughput* rata-rata 785 Kbit/s sedangkan AODV 720 Kbit/s. Untuk 64 KB, AODV-BR menghasilkan *throughput* rata-rata sebesar 672 Kbit/s sedangkan AODV 609 Kbit/s. Untuk 128, AODV-BR menghasilkan *throughput* rata-rata sebesar 535 Kbit/s, sedangkan AODV 459 Kbit/s. Dari rata-rata yang didapatkan dapat diplot ke dalam grafik. Grafik rata-rata *throughput* yang dihasilkan oleh proses simulasi terhadap kinerja protokol *routing* AODV-BR dan AODV untuk 3 *hop* dapat dilihat pada Gambar 4.3 dibawah.



Gambar 4.3 Perbadingan Rata-rata *Throughput* 3 Hop

Berdasarkan Gambar 4.3 perbandingan *throughput* yang dihasilkan dari uji coba simulasi 3 *hop* pada AODV-BR menghasilkan nilai *throughput* yang lebih tinggi dari AODV. Hal ini dipicu oleh *delay* yang dibutuhkan AODV-BR lebih sedikit dari pada AODV tradisional.

4.1.2.2 *Throughput* untuk 4 hop

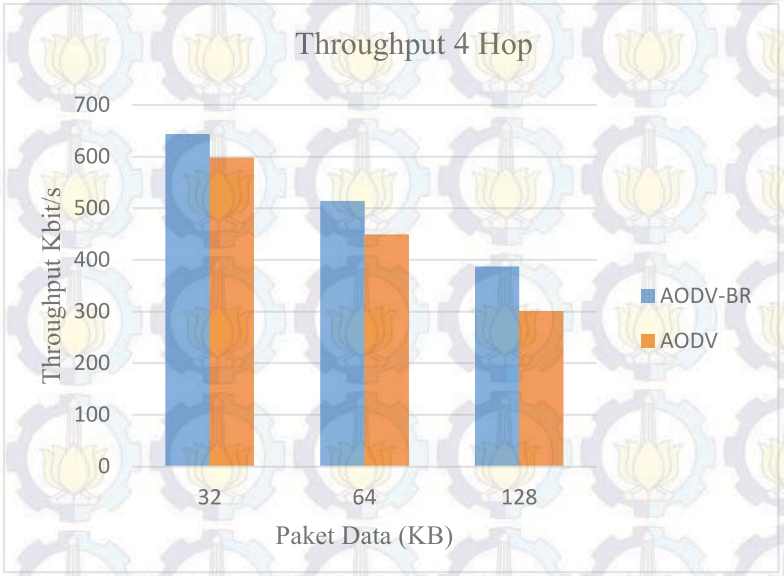
Pengukuran terhadap *throughput* yang dihasilkan oleh simulasi dengan 4 *hop* memiliki skenario yang sama dengan pengujian 3 *hop*. Perbedaan jumlah *node* yang awalnya 3 *hop* menjadi 4 *hop* memberikan kontribusi nilai hasil yang berbeda,

karena dipengaruhi oleh jumlah *node* yang terlibat. Adapun hasil pengukuran *throughput* dari hasil simulasi terhadap performa AODV-BR dan AODV tradisional terdapat pada Tabel 4.4 dibawah.

Tabel 4.5 *Throughput* yang Didapatkan untuk Simulasi 4 *hop* AODV-BR dan AODV

Perc. Ke	32 KB		64 KB		128 KB	
	AODV-BR	AODV	AODV-BR	AODV	AODV-BR	AODV
1	680	620	540	470	410	320
2	668	613	537	463	397	287
3	672	616	536	460	399	307
4	640	618	520	458	370	312
5	637	601	512	456	360	325
6	633	596	510	449	348	231
7	631	593	502	443	398	298
8	628	585	498	442	378	309
9	625	570	496	426	400	311
10	623	564	490	420	405	310
Rata-rata	**644**	**598**	**514**	**449**	**387**	**301**

Dari beberapa hasil uji coba simulasi terhadap performa routing AODV-BR dan AODV yang ditunjukkan pada Tabel 4.5 diatas dapat diambil nilai rata-rata dikarenakan data yang dihasilkan tidak sama dalam setiap percobaan disebabkan pergerakan *node* yang sifatnya acak dan untuk lebih mudahnya dalam menyimpulkan sebuah hasil. Hasil simulasi AODV-BR dan AODV tradisional yang melibatkan jumlah 4 *hop* atau 5 *node* dengan variasi sampel pengiriman data 32 KB, 64 KB, dan 128 KB. Untuk pengrimana data 32 KB, AODV-BR menghasilkan rata-rata *throughput* sebesar 644 Kbit/s, sedangkan AODV 598 Kbit/s. Untuk 64 KB, AODV-BR menghasilkan rata-rata *throughput* 514 Kbit/s, sedangkan AODV 449 Kbit/s. Untuk 128 KB, AODV-BR menghasilkan rata-rata *throughput* 387 Kbit/s, sedangkan AODV 301 Kbit/s. Untuk lebih mudahnya dalam proses analisa terhadap nilai rata-rata dari sepuluh percobaan pengambilan data, nilai rata-rata dapat diplot ke dalam grafik. Grafik rata-rata *throughput* yang dihasilkan dapat dilihat pada Gambar 4.4 dibawah.



Gambar 4.4 Perbadingan Rata-rata *Throughput* dalam 4 *Hop*

Berdasarkan grafik pada Gambar 4.4, AODV-BR memiliki hasil *throughput* yang lebih tinggi dari pada AODV tradisional dari tiga variasi data yang sudah ditentukan. Simulasi AODV-BR memiliki kinerja *routing* yang lebih baik dengan adanya penambahan *multiple route* atau *alternative route* pada AODV tradisional.

4.2 Hasil Pengujian *Testbed*

Hasil pengujian terhadap kinerja protokol *routing* AODV-BR pada VMeS menggunakan *testbed* laptop dengan memanfaatkan *wireless* IEEE 802.11. Proses pengujian sesuai dengan skenario yang sudah dibahas pada sub bab 3.4. Pengujian *testbed* dilakukan dengan tujuan untuk mengetahui kinerja protokol routing AODV-BR tidak hanya dengan hasil simulasi menggunakan *software* simulator saja, melainkan mengetahui secara riil performa dari protokol ini setelah diimplementasikan pada sebuah perangkat. Parameter *Quality of Service (QoS)* yang diukur terhadap performasi jaringan pada *testbed* disesuaikan dengan simulasi, yaitu mengukur nilai parameter *delay* yang dibutuhkan dan *throughput* yang dihasilkan.

4.2.1 *Delay*

4.2.1.1 *Delay* untuk 3 *hop*

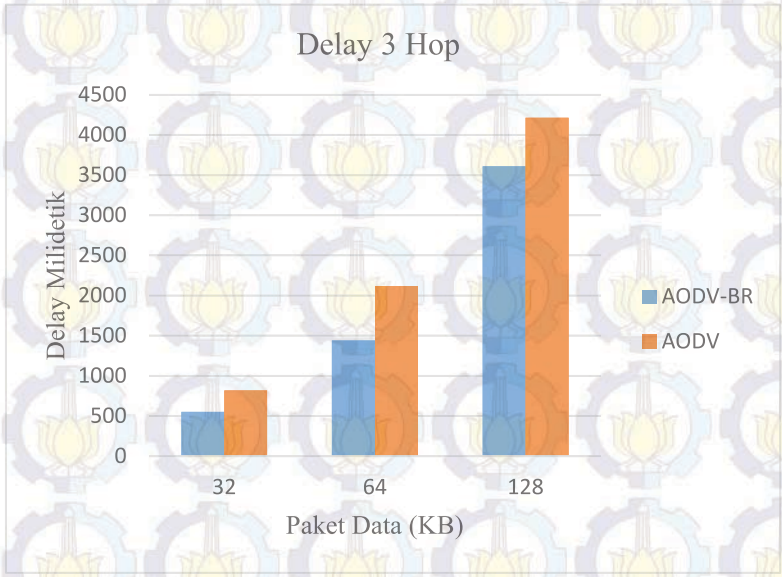
Untuk pengukuran *delay* yang dibutuhkan dalam pengiriman paket dengan jumlah 3 *hop*, dimana terdapat 4 *node* yang akan dilalui oleh paket data pada skema 4 *hop* yang akan dijadikan sebagai jalur alternatif. Paket dikirim dari *node* 1 (10.10.10.1) ke *node* 3 (10.10.10.3) atau sebaliknya yang melalui *node* 2 (10.10.10.2) sebagai *intermediate node* untuk *primary route*. Untuk pengiriman data melalui alternatif *route*, *node* 1 (10.10.10.1) ke *node* 3 (10.10.10.3) akan melalui *node* 4 (10.10.10.4) dan *node* 5 (10.10.10.5) sebagai *intermediate node*. Data *delay* yang diukur dibandingkan antara AODV-*Backup Routing* dengan AODV tradisional dengan pengambilan sampel besaran data yang bervariasi yaitu 32 KB, 64 KB, dan 128 KB.

Tabel 4.6 *Delay* untuk 3 *Hop* AODV-BR dan AODV pada *Testbed*

Perc. Ke	32 KB		64 KB		128 KB	
	AODV-BR	AODV	AODV-BR	AODV	AODV-BR	AODV
1	510	865	1401	2000	3156	4654
2	507	754	1400	2103	3231	4100
3	545	776	1476	2034	3547	4643
4	589	852	1489	2187	4721	4653
5	565	823	1420	2174	3800	3540
6	511	857	1434	2032	4752	4431
7	576	759	1408	2100	2439	3760
8	571	843	1440	2123	4100	5551
9	586	832	1450	2057	3587	3700
10	591	832	1498	2400	2770	3124
Rata-rata	**555**	**819**	**1442**	**2121**	**3610**	**4215**

Pada Tabel 4.6 merupakan hasil pengukuran yang dilakukan terhadap *delay* yang butuhkan pada saat uji coba pada perangkat dengan jumlah 3 *hop*. Perbedaan *delay* yang dibutuhkan dibedakan antara AODV-BR dan AODV tradisional. Besaran paket yang digunakan yaitu 32 KB, 64 KB, dan 128 KB. Uji coba dilakukan dalam skenario yang sama sampai 10 kali percobaan. Pada Tabel 4.6 terlihat perbedaan antara percobaan satu dengan lainnya, hal ini disebabkan pengaruh perubahan konfigurasi posisi dan lingkungan pada saat uji coba. Dari 10

kali percobaan yang dilakukan diambil nilai rata-rata untuk memudahkan dalam analisa. Hasil pengujian pada perangkat, untuk variasi paket data 32 KB, rata-rata AODV-BR membutuhkan waktu pengiriman paket 555 milidetik sedangkan AODV 819 milidetik. Untuk 64 KB, AODV-BR membutuhkan 1442 milidetik sedangkan AODV 2121 milidetik. Untuk 128 KB, rata-rata AODV-BR membutuhkan 3610 milidetik sedangkan AODV membutuhkan 4215 milidetik. Untuk memudahkan dalam analisa, hasil data rata-rata yang diperoleh dapat diplot ke dalam grafik. Grafik rata-rata dari waktu yang dibutuhkan dalam pengiriman paket oleh AODV-BR dan AODV untuk 3 *hop* dapat dilihat pada Gambar 4.5 dibawah.



Gambar 4.5 Perbadingan rata-rata *delay* dalam 3 *hop*

Pada Gambar 4.5 merupakan grafik perbandingan waktu yang diperlukan untuk pengiriman data pada 3 *hop* uji coba *routing* pada testbed. AODV-BR memberikan performasi waktu yang lebih rendah dari pada AODV tradisional. Pada grafik tersebut menunjukkan, semakin besar sampel data yang dikirim semakin besar pula *delay* yang dibutuhkan. Pada pengiriman data dengan kapasitas data 128 KB, membutuhkan *delay* 4-6 detik untuk 3 *hop* atau 4 *node* yang dilalui.

4.2.1.2 Delay untuk 4 *Hop*

Untuk pengukuran *delay* yang dibutuhkan dalam pengiriman paket dengan jumlah 4 *hop*, dimana terdapat 5 *node* yang terlibat dalam jaringan pada skema 4 *hop* yang akan dijadikan sebagai jalur alternatif. Sesuai dengan skenario pengujian, paket dikirm dari *node* 1 (10.10.10.1) ke *node* 4 (10.10.10.4) atau sebaliknya yang melalui *node* 2 (10.10.10.2) dan *node* 3 (10.10.10.3) sebagai *intermediate* node untuk *primary route*. Untuk pengiriman data melalui alternatif *route*, *node* 1 (10.10.10.1) ke *node* 4 (10.10.10.4) akan melalui node 2 (10.10.10.2), *node* 5 (10.10.10.5), dan *node* 6 (10.10.10.6) sebagai *intermediate node*. Data *delay* yang diukur dibandingkan antara AODV-*backup routing* dengan AODV tradisional dengan pengambilan sampel besaran data yang bervariasi yaitu 32 KB, 64 KB, dan 128 KB.

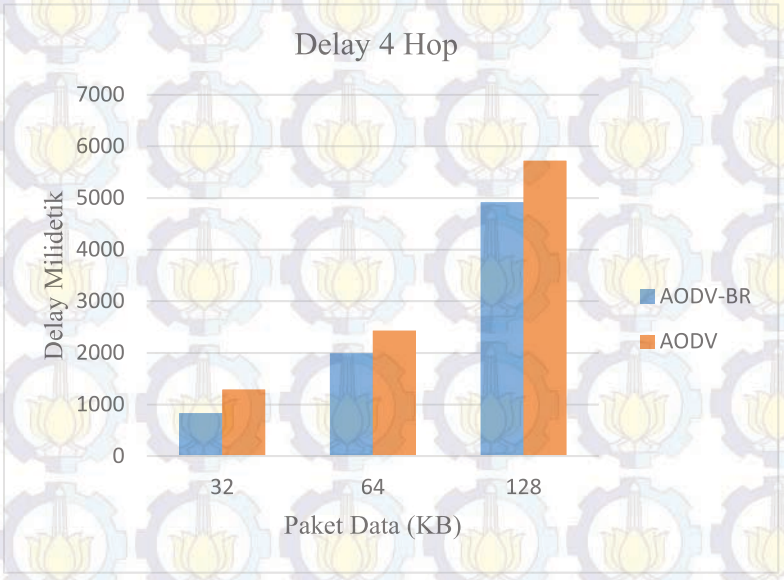
Tabel 4.7 *Delay* yang Dibutuhkan untuk 4 Hop AODV-BR dan AODV

Perc. Ke	32 KB		64 KB		128 KB	
	AODV-BR	AODV	AODV-BR	AODV	AODV-BR	AODV
1	800	1230	2000	2400	5654	5120
2	812	1283	1800	2430	5100	6044
3	801	1301	1832	2471	5643	6321
4	832	1324	2201	2429	4653	6542
5	854	1287	1983	2466	5540	5871
6	832	1293	2000	2412	4431	6232
7	866	1243	1832	2422	4760	5989
8	841	1382	2110	2462	4551	5821
9	841	1300	2265	2410	4700	4761
10	830	1280	1863	2420	4124	4541
Rata-rata	**831**	**1292**	**1989**	**2432**	**4916**	**5724**

Pada Tabel 4.7 merupakan hasil pengukuran yang dilakukan terhadap *delay* yang butuhkan pada saat uji coba pada perangkat dengan jumlah 4 *hop*. Perbedaan *delay* paket yang dibutuhkan dibedakan antara AODV-BR dan AODV tradisional. Besaran paket yang digunakan yaitu 32 KB, 64 KB, dan 128 KB. Uji coba dilakukan dalam skenario yang sama pada 10 kali percobaan. Pada Tabel 4.7 terlihat perbedaan antara percobaan satu dengan lainnya, hal ini disebabkan pengaruh perubahan konfigurasi posisi dan pengaruh lingkungan transmisi dari

setiap *node* yang berbeda-beda. Hasil pengujian pada perangkat, untuk variasi paket data 32 KB, rata-rata AODV-BR membutuhkan waktu pengiriman paket 831 milidetik sedangkan AODV 1292 milidetik. Untuk 64 KB, AODV-BR membutuhkan 1989 milidetik sedangkan AODV 2432 milidetik. Untuk 128 KB, rata-rata AODV-BR membutuhkan 4916 milidetik sedangkan AODV membutuhkan 5724 milidetik. Dari hasil pengukuran, tampak semakin bertambahnya node yang telibat untuk jadi hop dari *source* ke *destination* akan mempengaruhi terhadap *delay* yang dibutuhkan untuk setiap variasi data yang digunakan.

Untuk memudahkan dalam analisa, hasil data rata-rata yang diperoleh dapat diplot ke dalam grafik. Grafik rata-rata dari waktu yang dibutuhkan dalam pengiriman paket untuk menguji performa routingnya oleh AODV-BR dan AODV untuk 4 *hop* dapat dilihat pada Gambar 4.6 dibawah.



Gambar 4.6 Perbadingan Rata-rata *Delay* dalam 4 *Hop*

Pada Gambar 4.6 merupakan grafik perbandingan waktu yang diperlukan untuk pengiriman data pada 4 *hop* uji coba *routing* pada *testbed*. AODV-BR memberikan performasi waktu yang lebih rendah dari pada AODV tradisional. Pada grafik tersebut menunjukkan, semakin besar sampel data yang dikirim semakin besar pula *delay* yang dibutuhkan. Pada pengiriman data dengan kapasitas data 128 KB, membutuhkan *delay* 14-16 detik untuk 4 Hop atau 5 node yang dilalui.

4.2.2 *Throughput*

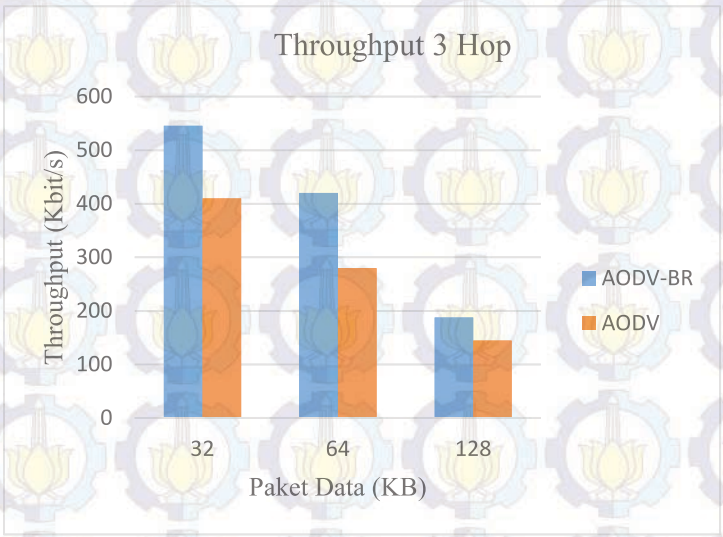
4.2.2.1 *Throughput* untuk 3 *hop*

Untuk hasil data pengukuran terhadap *throughput* yang dihasilkan dari uji coba implementasi pada *testbed*. *Throughput* yang dihasilkan sesuai dengan *delay* yang dibutuhkan, semakin tinggi *delay* pada proses pengiriman data, maka *throughput* yang dihasilkan akan semakin rendah. Untuk hasil pengukuran terhadap *throughput* untuk jarak 3 *hop* pada uji coba kinerja routing AODV-BR dan AODV terdapat pada Tabel 4.8 dibawah.

Tabel 4.8 *Throughput* yang Didapatkan untuk 3 *Hop* AODV-BR dan AODV

Perc. Ke	32 KB		64 KB		128 KB	
	AODV-BR	AODV	AODV-BR	AODV	AODV-BR	AODV
1	591	388	430	320	201	141
2	598	429	432	292	192	155
3	571	418	409	301	185	142
4	518	394	415	279	180	142
5	531	410	427	281	172	147
6	532	412	423	206	179	149
7	586	423	429	295	188	138
8	508	407	420	285	215	146
9	524	400	419	298	183	140
10	501	415	400	243	180	152
Rata-rata	**546**	**410**	**420**	**280**	**188**	**145**

Pada Tabel 4.8 merupakan data pengukuran terhadap *throughput* yang dihasilkan pada saat uji coba pada perangkat pada jarak 3 *hop* atau 4 *node* yang akan dilalui data dari sumber ke penerima. Dari 10 kali percobaan pengambilan data menunjukkan hasil yang berbeda, untuk memudahkan dalam mengambil kesimpulan maka perlu diambil nilai rata-rata dari setiap percobaan. Pada variasi data sebesar 32 KB, AODV-BR mengahasilkan rata-rata throughput 546 Kbit/s, sedangkan AODV 410 Kbit/s. Untuk 64 KB AODV-BR menghasilkan rata-rata *throughput* 420 Kbit/s, sedangkan AODV 280 Kbit/s. untuk 128 KB, AODV-BR menghasilkan rata-rata throughput 188 Kbit/s, sedangkan AODV 145 Kbit/s. Dari nilai rata-rata diatas dapat diplot ke dalam grafik pada Gambar 4.7.



Gambar 4.7 Perbadingan Rata-rata *throughput* dalam 3 *Hop*

4.2.2.2 *Throughput* untuk 4 hop

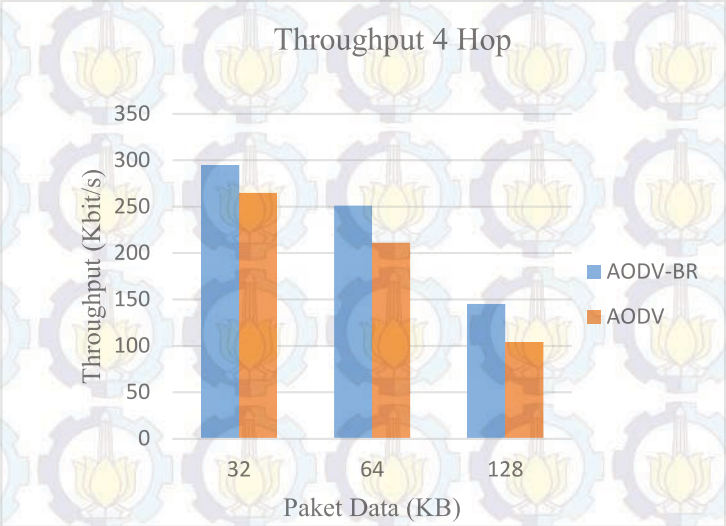
Untuk hasil pengukuran terhadap *throughput* untuk jarak 4 *hop* pada uji coba kinerja *routing* AODV-BR dan AODV pada *testbed* laptop terdapat pada Tabel 4.9 dibawah.

Tabel 4.9 *Throughput* yang Didapatkan untuk 4 *Hop* AODV-BR dan AODV

Perc. Ke	32 KB		64 KB		128 KB	
	AODV-BR	AODV	AODV-BR	AODV	AODV-BR	AODV
1	312	226	255	217	141	107
2	307	223	263	213	155	116
3	311	317	261	201	142	105
4	296	267	235	214	142	100
5	280	320	258	202	147	98
6	298	219	255	216	149	103
7	275	224	261	215	138	114
8	285	312	245	203	146	98
9	285	217	229	216	140	99
10	298	321	250	214	152	100
Rata-rata	**295**	**265**	**251**	**211**	**145**	**104**

Pada Tabel 4.9 merupakan data pengukuran terhadap *throughput* yang dihasilkan pada saat uji coba pada perangkat pada jarak 4 *hop* atau 5 *node* yang akan dilalui data dari sumber ke penerima. Dari 10 kali percobaan pengambilan data

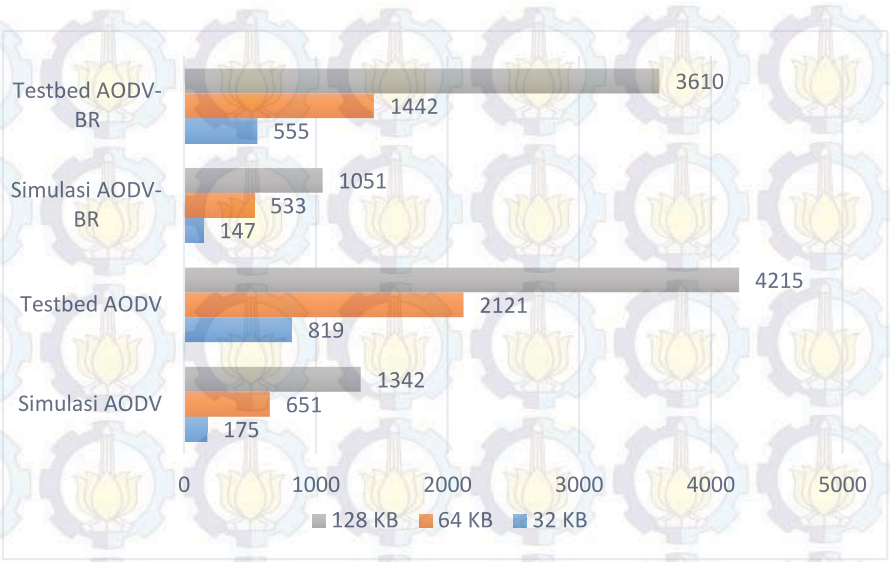
menunjukkan hasil yang berbeda, untuk memudahkan dalam mengambil kesimpulan maka perlu diambil nilai rata-rata dari setiap percobaan. Pada variasi data sebesar 32 KB, AODV-BR menghasilkan rata-rata *throughput* 295 Kbit/s, sedangkan AODV 265 Kbit/s. Untuk 64 KB AODV-BR menghasilkan rata-rata *throughput* 251 Kbit/s, sedangkan AODV 211 Kbit/s. Untuk 128 KB, AODV-BR menghasilkan rata-rata throughput 145 Kbit/s, sedangkan AODV 104 Kbit/s. Dari nilai rata-rata diatas dapat diplot ke dalam grafik pada Gambar 4.8 dibawah.



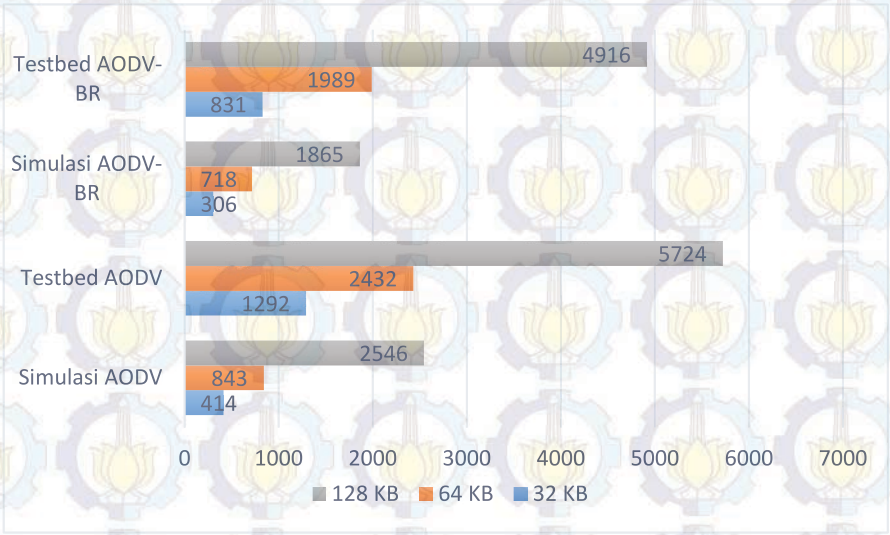
Gambar 4.8 Perbadingan Rata-rata *throughput* dalam 3 *Hop*

4.3 Analisa dan Pembahasan

Untuk mengetahui dan memaparkan hasil analisa terhadap data-data hasil pengukuran simulasi maupun uji coba pada *testbed* dapat dibandingkan, agar dapat menganalisa kinerja dari protokol AODV-BR terhadap pengaruh antara pertambahan jumlah *node* dengan jumlah data yang ditransmisikan. Untuk melakukan analisa terhadap hasil pengukuran pada parameter *QoS*, dipaparkan hasil perbandingann antara hasil pengukuran simulasi pada *network simulator* dengan hasil pengukuran pada *testbed* laptop.



Gambar 4.9 Perbandingan Rata-rata *Delay* Simulasi dan *Testbed* AODV-BR dengan AODV untuk 3 *Hop*



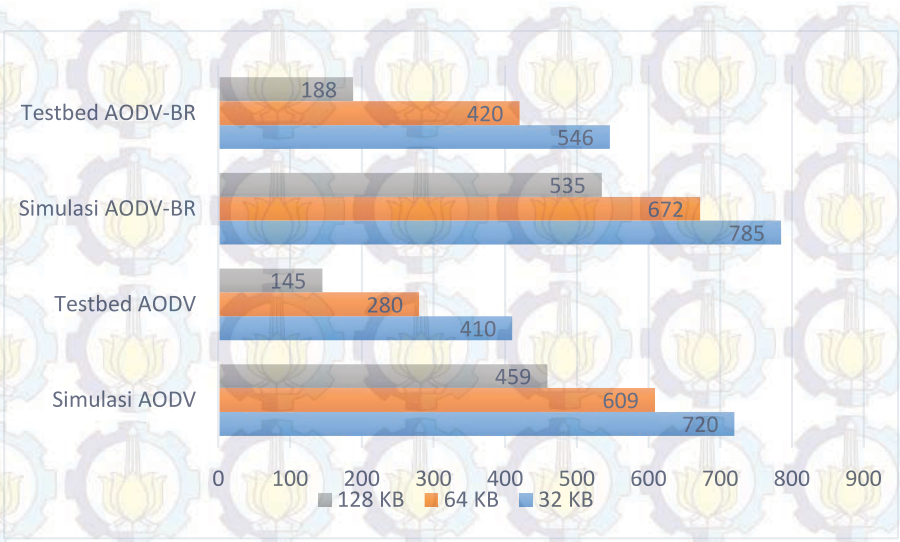
Gambar 4.10 Perbandingan Rata-rata *Delay* Simulasi dan *Testbed* AODV-BR dengan AODV untuk 4 *Hop*

Pada Gambar 4.9 Menunjukkan grafik perbandingan *delay* antara hasil simulasi dan hasil *testbed* untuk 3 hop dan pada Gambar 4.10 menunjukkan grafik perbandingan *delay* antara hasil simulasi dan hasil *testbed* untuk 4 *hop*. Perbedaan *delay* yang sangat signifikan terjadi pada pengiriman paket data sebesar 128 KB, dari masing-masing variasi jumlah *hop* yang diuji coba baik pada AODV-BR maupun pada AODV pada hasil *testbed*. *Delay* yang tinggi pada *testbed* dipengaruhi oleh faktor lingkungan pada saat melakukan uji coba *routing* dan faktor

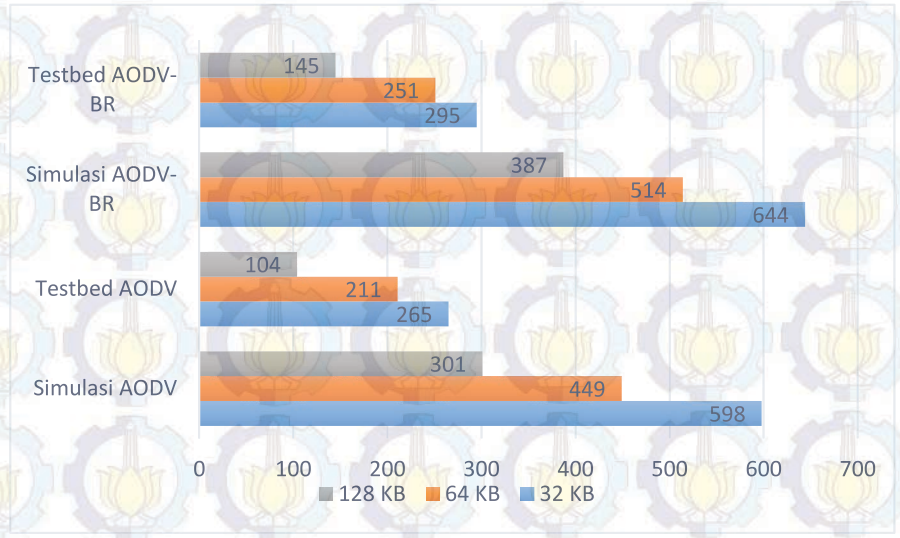
perangkat yang digunakan. Setiap *node* yang digunakan memiliki performa yang berbeda, sehingga mengakibatkan *delay* yang tinggi untuk *testbed* 3 *hop* hingga 3610 milidetik untuk AODV-BR dan 4215 milidetik untuk AODV, dan untuk *testbed* 4 *hop* juga mengalami peningkatan yang jauh lebih besar hingga 4916 milidetik untuk AODV-BR dan 5724 milidetik untuk AODV. Kebutuhan *delay* yang besar dipengaruhi oleh faktor besaran data yang ditransmisikan serta beberapa faktor lain seperti faktor *delay* perangkat yang digunakan serta proses *route discovery* pada masing-masing protokol *routing*.

Untuk rata-rata pada pengujian *testbed* 3 – 4 hop, AODV-BR memiliki estimasi waktu yang lebih rendah dari pada AODV, disebabkan pada waktu pengujian diskenariokan jalur *primary* putus agar *routing* AODV-BR dapat secara otomatis memanfaatkan jalur alternatif yang dijadikan kontribusi perbaikan terhadap AODV tradisional. Skenario pengujian yang sama juga diterapkan pada AODV tradisional agar dapat membandingkan hasil kinerja apabila terjadi *route break* pada saat melakukan komunikasi. AODV membutuhkan *delay* yang lebih lama walaupun dalam kondisi skenario pengukuran yang sama, disebabkan AODV tradisional perlu melalukan *setup route* kembali saat jalur yang digunakan mengalami *route break*. Berdasarkan grafik yang ditampilkan pada Gambar 4.9 dan Gambar 4.10 AODV-BR memiliki estimasi waktu $\pm$ 10% s/d 15% lebih kecil dari pada *delay* yang dibutuhkan oleh AODV tradisional.

Untuk delay yang dihasilkan simulasi yang ditunjukkan dalam grafik pada gambar rata-rata antara tiga jenis variasi yang digunakan masih dalam kondisi stabil dikarenakan pada simulasi, perangkat yang digunakan diasumsikan sama, dan delay pada setiap node diasumsikan 0. Delay yang berpengaruh pada simulasi hanya delay yang disebabkan oleh *route discovery* dan waktu pengiriman paket.



Gambar 4.11 Perbandingan Rata-rata *Throughput* Simulasi dan *Testbed* AODV-BR dengan AODV untuk 3 *Hop*



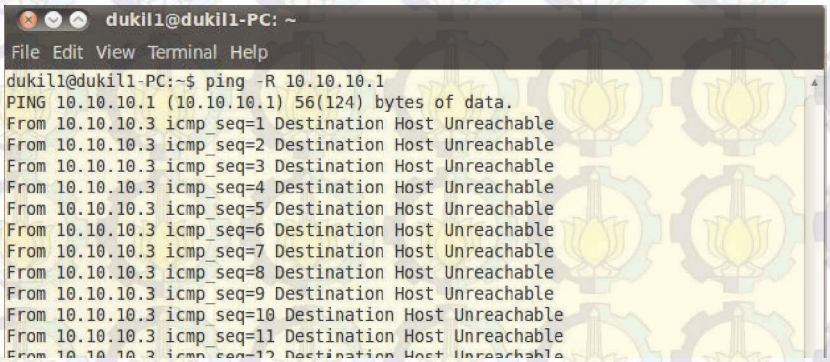
Gambar 4.12 Perbandingan Rata-rata *Throughput* Simulasi dan *testbed* AODV-BR dengan AODV untuk 4 *Hop*

Pada Gambar 4.11 dan Gambar 4.12 menunjukkan grafik perbandingan terhadap rata-rata *throughput* yang dihasilkan dari keseluruhan pengujian terhadap kinerja protokol routing AODV-BR dan AODV tradisional. *Throughput* yang dihasilkan berbading lurus dengan *delay* yang dibutuhkan. Semakin besar *delay* yang dibutuhkan, maka *throughput* yang dihasilkan semakin rendah. Pada hasil *testbed*, rata-rata antara uji coba pada 3 *hop* dan 4 *hop* mengalami penurunan dalam setiap variasi kapasitas data yang dikirimkan, baik pada AODV-BR maupun AODV

tradisional. *Throughput* tertinggi yang dihasilkan dari pengukuran terhadap kinerja *routing* AODV-BR maupun AODV tradisional yaitu pada hasil simulasi, dikarenakan pada simulasi *delay* yang dibutuhkan sangat rendah. *Throughput* tertinggi yang dihasilkan oleh AODV-BR mencapai 785 Kbit/s pada uji coba simulasi 3 *hop* dan 644 Kbit/s pada uji coba simulasi 4 *hop* untuk variasi data 32 KB. Sedangkan *throughput* tertinggi yang dihasilkan AODV mencapai 720 Kbit/s pada uji coba 3 hop dan 598 Kbit/s pada uji coba simulasi 4 hop untuk variasi data 32 KB.

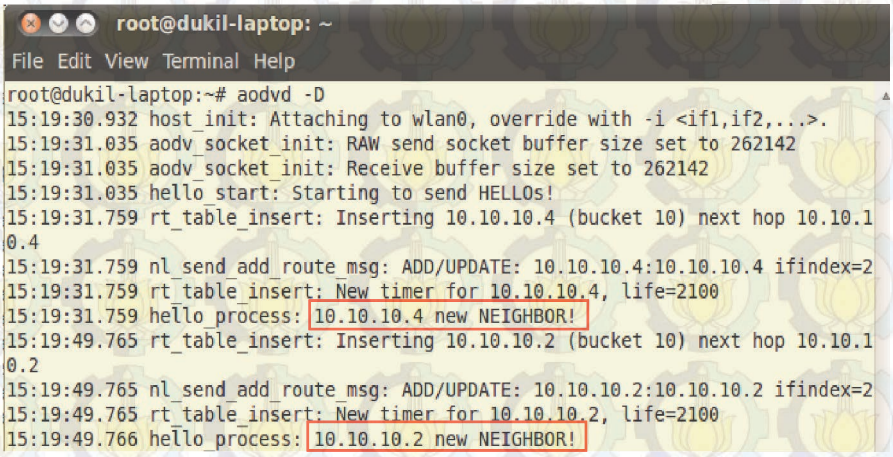
4.4 Uji Kemampuan *Routing Testbed*

Pengujian ini dilakukan setelah semua *node* yang akan dilibatkan dalam pengukuran yang terdiri dari 3 dan 4 *hop* ditata sesuai topologi jaringan untuk pengujian. Sebelum melakukan *cheking* terhadap jalur yang akan dilalui, setiap *node* harus menjalankan AODV-BR secara serempak untuk dapat melakukan uji coba *routing*. Pertama dilakukan uji coba koneksi dari *node* 3 ke *node* 1 tanpa menjalankan *routing* protokol.



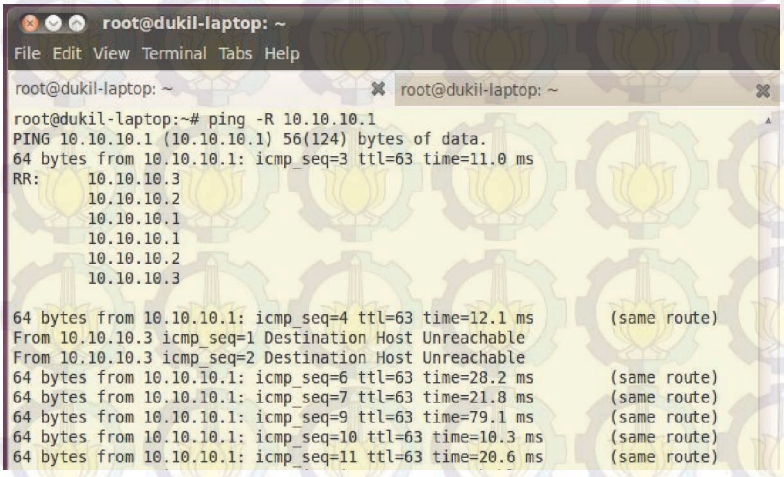
Gambar 4.13 Tidak Ada Koneksi dari *Node* 3 ke 1

Gambar 4.13 di atas menampilkan tidak adanya koneksi antara *node* 3 (10.10.10.3) ke *node* 1 (10.10.10.1) disebabkan *node* 2 (10.10.10.2) yang dijadikan *next hop* oleh *node* 3 menuju node 1 sebagai jalur *primary* tidak terditeksi adanya protokol *routing*. Atau melewati *node* 5 (10.10.10.5) dan *node* 4 (10.10.10.4) sebagai jalur alternatif.



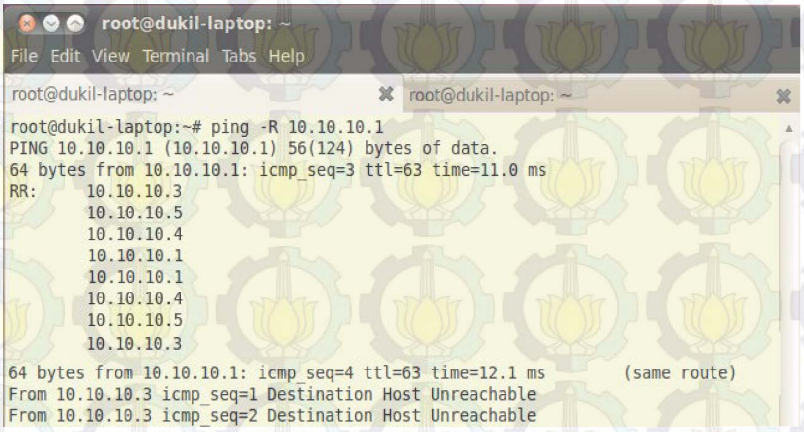
Gambar 4.14 Protokol *Routing* Mendeteksi *Node* Tetangga *(neighbor)*

Pada Gambar 4.14 protokol *routing* AODV-BR mendeteksi *node* tetangga dan melakukan *update table routing* secara berkala pada setiap *node*. *Broadcast* pesan *hello* digunakan untuk mendeteksi *node* tetangga untuk dijadikan *next hop*. Secara *default*, *broadcast hello* dilakukan secara periodik per 2100 milidetik. Setelah masing-masing *node* melakukan *update routing* pada setiap *node*. Selanjutnya, dilakukan pengujian *routing* menggunakan perintah *ping* dengan ekstensi –R untuk melihat rute yang akan dilalui.



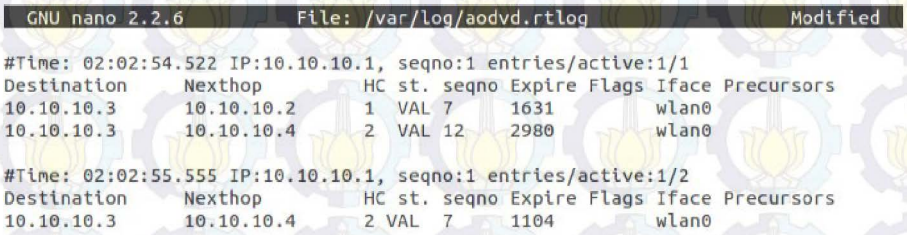
Gambar 4.15 Uji *Routing* untuk 3 *Hop Primary Route*

Pada Gambar 4.15 melakukan *Ping* dengan ekstensi –R dari *node* 3 ke *node* 1 dengan menampilkan *next hop* yang akan dilalui paket yaitu *node* 2. Mengacu pada skenario pengujian, *primary route* akan diputus atau dikeluarkan dari *coverage area* agar secara otomatis AODV-BR memanfaatkan jalur alternatif.



Gambar 4.16 Uji *Routing* untuk 3 *hop* Jalur Alternatif

Pada Gambar 4.16 menampilkan jalur yang akan dilalui dalam pengujian 3 *hop*, dimana jalur alternatif memiliki jumlah *next hop* yang lebih besar dari pada jalur *primary*. Untuk pengujian pada 4 *hop*, akan dilakukan pada skenario yang sama. Pengujian koneksi *routing* ini dilakukan pada saat proses pengiriman sampel paket data *dummy* menggunakan *tool iperf* dengan besaran paket yang sudah ditentukan.



Gambar 4.17 Tabel *Routing* AODV-BR pada *Node* 1

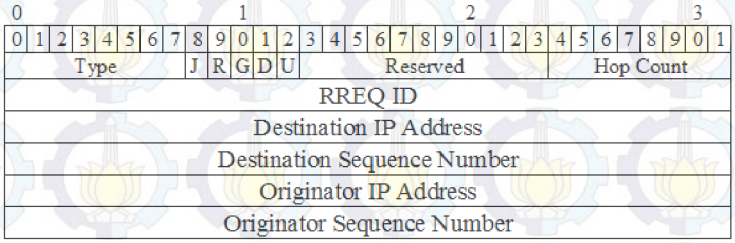
AODV-BR menyediakan alternatif *route*. alternatif *route* akan difungsikan sesuai dengan skenario pengujian. AODV-BR akan memanfaatkan *primary route* jika tidak terjadi *route break*. Proses yang terlihat pada Gambar 4.17 beberapa detik kemudian, AODV-BR menggeser *alternative route* ke urutan utama dan menghapus jalur *primary* dan tetap melanjutkan pengiriman paket data dikarenakan *node* jalur utama dikeluarkan dari *caverage area*. AODV-BR dalam menentukan jalur utama dan jalur alternatif disesuaikan dengan jumlah *hop count* yang akan dilalui. Jumlah *hop cont* yang lebih sedikit akan diprioritaskan.

4.5 Pembahasan Rekomendasi Routing Protokol

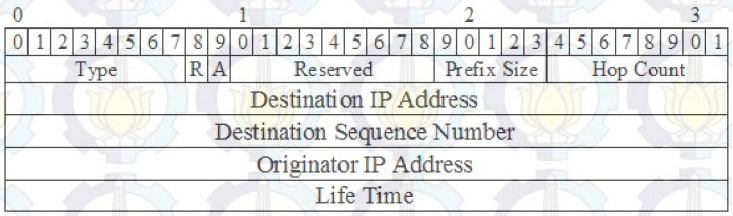
Hasil pengujian kinerja terhadap protokol AODV-BR dibandingkan dengan AODV tradisional menunjukkan AODV-BR lebih efektif diterapkan untuk VMeS berdasarkan hasil parameter *delay* yang dihasilkan dari pengukuran. AODV-BR mampu mereduksi *delay* yang digunakan oleh AODV, delay ditimbulkan berdasarkan skenario pengukuran disebabkan oleh pemodelan pengukuran, bukan dikarenakan kapasitas *overhead*. *Delay* yang kecil cocok diterapkan pada data rate yang rendah seperti HF/VHF yang hanya memiliki *data rate* 1200 bit [4].

Penelitian ini menggunakan MAC protokol 802.11 pada lapisan *datalink*-nya, dimana MAC protokol 802.11 memiliki kapasitas *frame* yang tinggi hingga 2346 *byte* (2,3 KB). Sedangkan perangkat VMeS yang akan dirancang pada protokol MAC-nya menggunakan Amatir AX.25 (AX.25) disebabkan lapisan fisik yang digunakan adalah perangkat yang mendukung frekuensi HF/VHF seperti *handy talky* dll. AX.25 memiliki kapasitas *frame* yang lebih pendek dari pada protokol MAC 802.11, yaitu 256 *byte* (0,25 KB). Kapasitas *frame* yang lebih sedikit diperlukan optimasi pada setiap kapasitas paket yang dikirimkan oleh lapisan *Network* melalui lapisan *Data Link* agar dapat mengoptimalkan kinerja dari suatu layanan sistem telekomunikasi.

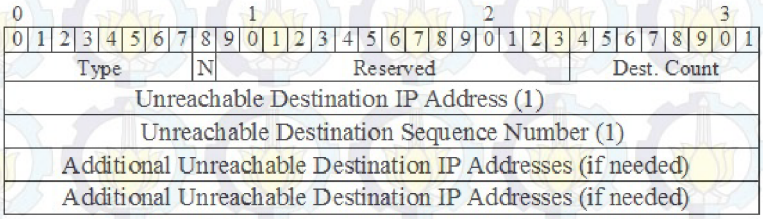
Pembahasan ini berdasarkan referensi standar protokol *routing* dan protokol MAC yang sudah digunakan. Protokol routing AODV yang digunakan memiliki tiga komponen paket utama dalam proses penentuan jalur, yaitu RREQ, RREP, dan RRER. Masing-masing paket memiliki total *bit* yang berbeda, hal ini dapat dianalisa dengan membandingkan dengan kapasitas *frame* yang dimiliki oleh protokol MAC yang digunakan.



Gambar 4.18 Format Pesan *RREQ (Route Request)* [14]



Gambar 4.19 Format Pesan *RREP (Route Reply)* [14]



Gambar 4.20 Format Pesan *RRER (Route Error)* [14]

Gambar 4.18 merupakan format RREQ yang difungsikan untuk permintaan pembuatan jalur memiliki panjang 32 bit, dimana total dari keseluruhan paket adalah 24 *Byte*. Gambar 4.19 merupakan format RREP yang digunakan untuk *setup route*, memiliki panjang 32 bit dengan total kapasitas paket 20 *Byte*. Gambar 4.20 merupakan format RRER yang digunakan informasi apabila terjadi *link break* atau *route error*, memiliki panjang 32 bit dengan total kapasitas paket 20 *Byte*.

Gambar 4.21 Fromat *Frame* 802.11 [15]

Gambar 4.21 merupakan format *frame* pada protokol MAC 802.11 layer data link. Dapat dilihat kapasitas dari *field frame body* memiliki pajang dari 0 –

2312 byte. *Frame body* merupakan alokasi kapasitas *byte* yang dikhususkan untuk kapasitas paket yang akan ditransmisikan dalam bentuk *frame*. RREQ memliki kapasitas 24 *Byte*, RREP memiliki kapasitas 20 Byte, dan RRER memiliki kapasitas 20 *Byte*. Kapasitas 24, 20, 20 *Byte* ditransmisikan satu-satu sesuai dengan kebutuhan sistem pada setiap *frame* pada layer *data link*. Kapasitas paket 24, 20 *Byte* jika dibandingkan dengan panjang dari *frame* yang disedikan protokol MAC 802.11 hanya digunakan 1 % saja dari total keseluruhan kapasitas dalam satu *frame*. Sehingga beban penggunaan data paket RREQ, RREP, dan RERR AODV pada *medium access control* 802.11 tidak memiliki pengaruh yang siginifikan terhadap beban kapasitas *frame*.

Flag	AX.25 Transfer Frame Header (128 bits)				Information Field	Frame-Check Sequence	Flag
	Destination Address	Source Address	Control Bits	Protocol Identifier			
8	56	56	8	8	0-2048	16	8

Gambar 4.22 Fromat *frame* AX.25 [17]

Gambar 4.21 merupakan format *frame* yang dimiliki oleh AX.25, pada *field info* memiliki kapasitas maksimal 256 *byte*. Kapasitas paket yang akan dikirimkan dari Layer Network akan diletakkan dalam *frame Information Field*, kapasitas paket apabila memiliki kapasitas melebihi 256 *byte* akan dipecah menjadi beberapa *frame* pada saat ditransmisikan sesuai dengan standar yang dimiliki oleh protokol lapisan *Data Link*. Kapasitas paket yang akan dikirimkan pada AODV adalah 20 – 24 *byte* untuk pembentukan suatu jalur sesuai standar protokol *routing* yang digunakan. Penggunaan *frame* untuk *overhead* digunakan 9,5 % dari total keseluruhan kapasita *frame* pada *field* informasi.

AODV dalam melakukan proses penentuan jalur pada layer *network*, melakukan pengriman paket pembentukan jalur secara periodik sesuai kebutuhan. Data yang dikirimkan dalam AODV tidak disertakan dengan paket penemuan jalur. Paket penemuan jalur dikatakan sebagai *overhead* apabila disertakan dengan data setelah melalui proses enkapsulasi.

Pada protokol MAC 802.11 penggunaan paket data pesan RREQ, RREP, dan RERR hanya menggunakan 1% dari keseluruhan *frame*, sedangkan pada

Protokol MAC AX.25 menggunakan 10% dari keseluruhan *frame*. Tidak perlu dilakukan modifikasi terhadap standar paket pesan dari protokol routing AODV disebabkan *overhead* tidak dikirimkan bersamaan dengan data seperti pada protokol TCP IP. Pada AODV data selain *overhead* dikirimkan setelah jalur terbentuk sempurna.

Jika pada penelitian selanjutnya hendak memodifikasi pada setiap jenis paket pesan pada AODV, dapat dilakukan kalkulasi terhadap kapasitas paket, mengurangi kapasitas *bit* setiap *field*, dan membuang *field* yang tidak terlalu digunakan yang ada pada paket pesan standar AODV sehingga bisa mencapai 5% dari penggunaan alokasi bit *frame* pada protokol AX.25. Rekomendasi ini, dapat dijadikan referensi untuk perancangan protokol *routing* pada penelitian selanjutnya yang akan diterapkan pada sistem dengan menggunakan protokol MAC AX.25.

DAFTAR PUSTAKA

[1] M. Ardita and A. Affandi, “Perancangan Terminal Komunikasi Data Terintegrasi untu Jaringan Ad-Hoc Vessel Messagging System (VMes)”, Surabaya: Thesis JTE-ITS, 2010.

[2] A. Affandi, "Sistem Komunikasi Data Terpadu Armada Perahu Nelayan Menggunakan Kanal Frekuensi Tinggi (VMeS-Vessel Messagging System)," in Hibah Pasca (HPTP), ITS Surabaya, 2007.

[3] A. Boukerche, “Algorithms and Protocols for Wireless and Mobile Adhoc Network”, Canada: A John Wiley & Sons, Inc. Publication, 2009.

[4] B. Y. Pratomo and A. Affandi, “Implementasi Protokol Jaringan Ad Hoc Pada Teminal Komunikasi Data untuk Sistem Komunikasi Kapal Laut”, Surabaya: JTE-ITS, 2011.

[5] K. Syarif, A. Affandi and D. S. Rahardjo, “Analisa Kinerja Protokol Routing Ad-Hoc on Demand Distance Vector (AODV) Pada Komunikasi VMeS”, Surabaya: JTE ITS, 2012.

[6] C. Perkins and E. M. Royer, "Ah-hoc On-demand Distance Vector Routing," in WMCSA'99 Second IEEE Workshop, New Orleans, LA, 1999.

[7] E. B. Royer, C. Perkins and S. Das, “Ad hoc On-Demand Distance Vector (AODV) Routing”, Internet Society, 2003.

[8] M. G. SJ Lee, "AODV -BR: Backup Routing in Ad hoc Networks," in IEEE, Chicago, IL, 2000.

[9] W. K. Lai, S. Y. Hsiao and Y. C. Lin, "Adaptive backup routing for ad-hoc networks," Elsevier, vol. 30, no. Computer Communications, pp. 453-464, 2007.

[10] F. P. T. A. A. Andhika, "Protokol Interchangeable Data pada VMeS (Vessel Messaging System) dan AIS (Automatic Identification System)," in Jurnal Teknik ITS, September 2012.

[11] A. A. Fuad Ahmad, “Mekanisme Komunikasi Data Gateway Multi-Terminal VMeS jaringan ad hoc Menggunakan Konsep Metode Akses Protokol MAC 802.11”, Surabaya: Tesis JTE-FTI-ITS, 2012.

[12] S. K. Sarkar, T. G. Basavaraju and C. Puttamadappa, “Ad Hoc Mobile Wireless Networks: Principles, Protocols and Applications”, France: Auerbach Publications, 2007.

[13] X. Hong, K. Xu and M. Gerla, "Scalable Routing Protocols for Mobile Ad Hoc Networks," IEEE, vol. 16, no. 4, pp. 11 - 21 , 2002.

[14] E. B. R. C. Perkins, “Ad hoc On-Demand Distance Vector (AODV) Routing”, RFC 3561, 2003.

[15] N. Pietro, "IEEE 802.11 frame format," www.studioreti.it, 2005.

[16] D. W. S. S. H. A. Prasetyo, "Perancangan dan Impelementasi Protokol AX.25 (Simplefield) pada MCS-51," Prosiding Seminar Nasional Sistem dan Informatika, 2007.

[17] N. D. E. T. J. B. A. William, “AX.25 Link Access Protokol for Amateur Packet Radio”, Tucson Amateur Packet Radio Corporation, 1997.

[18] T. Issariyakul and E. Hossain, “Introduction to NS 2”, Springer Science, 2009.

BAB 5
PENUTUP

5.1 Kesimpulan

Berdasarkan hasil analisa dan pengujian sistem yang dilakukan pada penelitian ini, yaitu mengenai analisa kinerja protokol *routing* AODV-BR pada komunikasi VMeS dapat diambil kesimpulan sebagai berikut:

1. Hasil pengukuran menunjukkan, perbandingan delay antara AODV-BR lebih kecil dari pada AODV tradisional dengan berbagai jenis variasi data yaitu 23, 64, 128 KB dengan jumlah hop yang sudah ditentukan.
2. *Throughput* yang dihasilkan oleh AODV-BR lebih besar dari pada AODV disebabkan AODV-BR tidak terjadi *routing loop* saat terjadi *route break* sehingga data yang ditransmisikan dengan teknik penambahan skema *backup routing* lebih banyak dari pada AODV tradisional.
3. Protokol routing AODV-BR memberikan teknik skema routing pada *layer network* untuk mengembangkan skema routing pada AODV tradisional dengan menambah *multipath routing* untuk mengatasi masalah *route break* yang berpengaruh terhadap meningkatnya *delay* yang dibutuhkan.
4. Hasil simulasi menunjukkan perbedaan yang signifikan dibandingkan dengan hasil testbed terhadap kinerja dari protokol routing yang dilakukan uji coba, dikarenakan pada testbed, faktor eksternal seperti lingkungan dan perangkat masih dipertimbangkan.
5. Penambahan jumlah node yang terlibat dalam komunikasi dan bertambahnya variasi data memiliki pengaruh terhadap *delay* yang dibutuhkan dan *throughput* yang dihasilkan terhadap AODV-BR dan AODV tradisional. Semakin bertambah jumlah node yang terlibat dan bertambahnya kapasitas data yang ditransmisikan, maka *delay* yang dibutuhkan semakin meningkat, dan *throughput* yang dihasilkan semakin menurun.

6. Hasil pengujian yang dilakukan didapatkan perbadingan kinerja dari AODV-BR pada VMeS berdasarkan *delay* yang dibutuhkan lebih sedikit dari pada AODV tradisional ± 10-15 % untuk hasil uji coba *testbed*.
7. AODV-BR dapat diimplementasikan di VMeS dikarenakan konsep yang ditawarkan pada AODV-BR sama dengan AODV tradisional, dimana teknik routing yang digunakan berdasarkan *on-demand*. AODV-BR akan bekerja saat dibutuhkan saja dan sangat cocok untuk sistem komunikasi yang memiliki *data rate* rendah seperti VMeS.

5.2 Saran

Berdasarkan hasil perancangan dan pengujian terhadap kinerja dari protokol routing AODV-BR, dapat diberikan beberapa saran yang berguna untuk penelitian selanjutnya pada komunikasi VMeS.

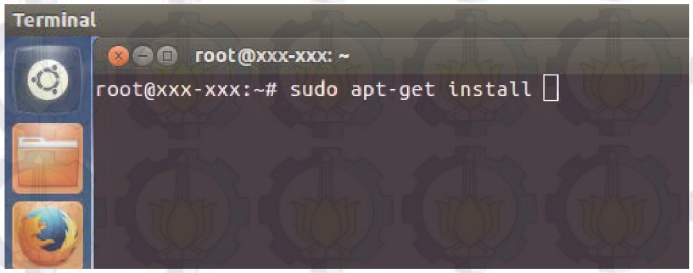
1. Skema routing yang diuji coba pada komunikasi VMeS ini menggunakan protokol MAC pada lapisan datalink IEEE 802.11 dengan skema *Distributed Coordination Function (DFC)* untuk *medium access control*-nya. Perlu dirancang dan diuji coba pada AX.25 dengan mencoba menerapkan mikanisme *Request To Send / Clear to Send (RTS/CTS)* yang ada pada 802.11 terhadap AX.25.
2. Uji coba pada MAC 802.11 sangat efektif, dengan penggunaan *frame* 1 % daru total *frame* pada 802.11, dan AX.25 10%. Untuk penelitian selanjutnya tidak perlu mereduksi paket utama pada protokol AODV-BR dengan alasan tidak dibutuhkannya kapasitas *frame* yang tinggi untuk jalankan protokol AODV-BR pada AX.25.
3. Penelitian terhadap protokol routing AODV-BR ini masih sampai pada simulasi secara tool simulasi dan *testbed*, dan masih sangat perlu dilakukan penelitian secara riil pada perangkat VMeS yang sebenarnya.
4. Untuk analisa kinerja pada protokol *routing* AODV-BR selanjutnya, perlu penambahan parameter *QoS* yang dijadikan parameter ukur agar dapat memberikan kontribusi analisa yang lebih relevan terhadap kinerja dari protokol AODV-BR ini.

LAMPIRAN

1. *Steep by steep* Instalasi NS di Ubuntu 12.04

Dibawah ini adalah langkah-langkah melakukan instalasi dan konfigurasi dari *Network Simulator 2.35* yang digunakan untuk simulasi Protokol AODV-BR. Untuk langkah-langkah detailnya adalah sebagai berikut:

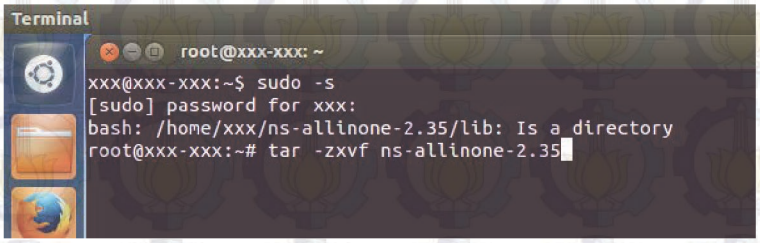
- Pastikan PC atau laptop sudah terinstal Sistem Operasi Linux Ubuntu 12.04 atau versi di atasnya seperti 12.10, 13.04, 13.10, 14.04, 14.10, dan saat ini sudah rilis 15.04. Repository dari OS tersebut sudah terupdate, bisa saja menggunakan official servernya Ubuntu atau pada server mirror lokal yang ada di Indonesia.
- Perangkat harus terhubung dengan jaringan internet pada saat proses instalasi, dikarenakan terdapat beberapa software pendukung yang dilakukan instalasi secara online. Untuk melakukan instalasi secara online digunakan script dasar pada Ubuntu seperti *$sudo apt-get install <nama software>* yang dapat dieksekusi melalui terminal (ctrl + T) pada Ubuntu.



Gambar 7.1 Script dasar Ubuntu untuk melakukan instalasi software

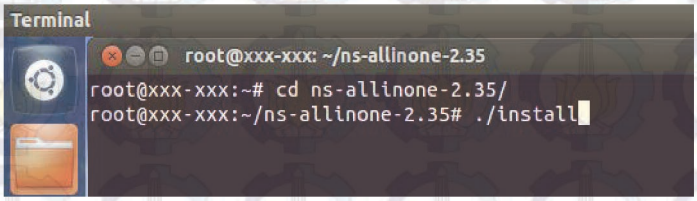
- Sebelum menginstall NS 2.35, lakukan instalasi software atau tool tambahan agar semua fungsi pada NS 2.35 dapat berjalan secara efektif pada Ubuntu 12.04. Sotware yang perlu diinstal sebelum melakukan instalasi NS adalah seperti; *tcl8.5-dev, tk8.5-dev, gcc-4.4, g++-4.4, build-essential, autoconf, automake, perl, xgraph, libxt-dev, lib11-dev,* dan *libxmu-dev*.
- Siapkan software *ns-allinone-2.35.tar.gz* yang bisa diunduh secara gratis pada situs resminya NS yaitu pada alamat www.nsnam.isi.edu.

- Letakkan file software NS pada direktori /Home, kemudian lakukan proses ekstraksi dari paket software NS dengan mengeksekusi perintahnya melalui terminal (ctrl + T). Ketikkan secara beruntun script berikut ini;
  - sudo –s
  - tar -zxvf ns-allinone-2.35.tar.gz



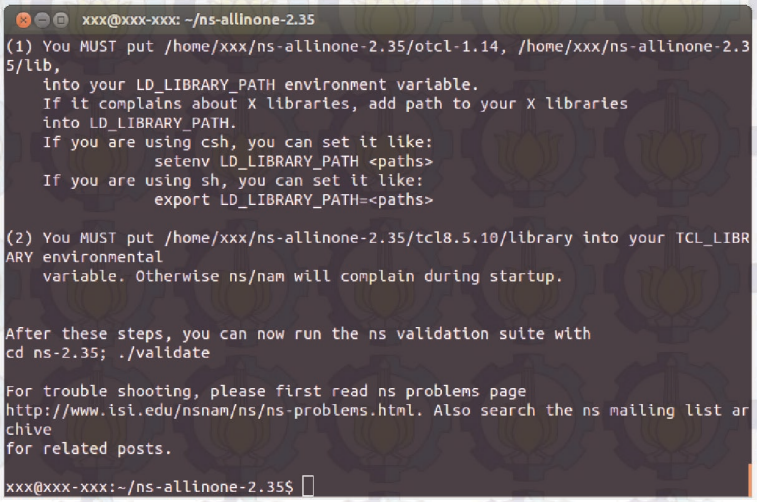
Gambar 7.2 Proses ekstraksi paket NS 2.35

- Setelah proses ekstraksi sudah selesai, arahkan direkrori di terminal pada folder hasil ekstrak dari ns-allinone-2.35, kemudian lakukan instalasi dengan mengeksekusi perintah berikut di terminal.
  - cd ns-allinone-2.35
  - ./install



Gambar 7.3 Proses instalasi NS 2.35

- Apabila proses instalasi tidak terjadi error, maka pada akhir *log* dari proses instalasi yang muncul diterminal akan memberikan informasi rekomendasi untuk mengubah variable LD_LIBRARY_PATH. Ini bertujuan untuk mengarahkan software atau tool yang tadinya diinstall sebelum melakukan instalasi NS, akan disingkronkan dan diarahkan ke direktori dimana NS terinstall. Untuk melakukan perubahan tersebut, dapat dilakukan dengan cara masuk ke folder /HOME, kemudian tekan tombol ctrl+h untuk membuka file yang disembunyikan oleh sistem operasi pada direktori tersebut. File yang disembunyikan ini diperlukan untuk mengubah alamat dari direktori yang direkomendasikan oleh NS.



Gambar 7.4 *Log* akhir dari instalasi NS

- Cari dan buka file dengan nama *.bashrc* yang secara default akan dibuka menggunakan editor gedit pada Ubuntu.
- Kemudian tambahkan script variabel berikut pada akhir dari file *.bashrc* tadi.

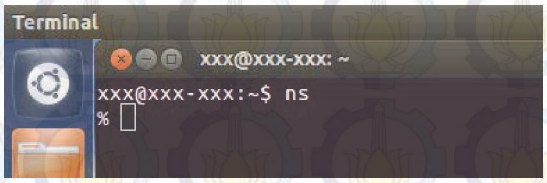
```
PATH=$PATH:/home/xxx/ns-allinone-2.35/bin:/home/xxx/ns-allinone-2.35/tcl8.5.10/unix:/home/xxx/ns-allinone-2.35/tk8.5.10/unix

LD_LIBRARY_PATH=$LD_LIBRARY_PATH: /home/xxx/ns-allinone-2.35/lib

TCL_LIBRARY=$TCL_LIBRARY:/home/xxx/ns-allinone-2.35/tcl8.5.10/library

export PATH
export LD_LIBRARY_PATH
export TCL_LIBRARY
```

- Edit nama user sesuai dengan user yang digunakan. Pada penelitian ini digunakan nama user "xxx". Setelah selesai melakukan pengeditan, simpanlah hasil edit yang telah dilakukan.
- Langkah selanjutnya adalah melakukan testing terhadap NS, apakah NS sudah berhasil diinstal atau tidak. Langkah ini bisa dilakukan dengan cara ketik NS pada terminal, jika hasilnya "%" berarti NS sudah terinstall dan bisa dijalankan.



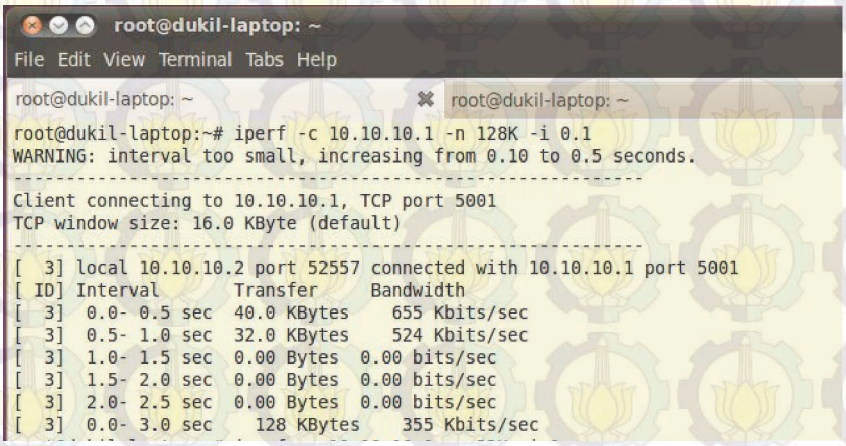
Gambar 7.5 *Testing* NS

- Proses akhir dari instalasi dan konfigurasi dari Network Simulator 2.35 ini adalah melakukan validasi terhadap seluruh perubahan yang dilakukan saat instalasi serta untuk mensinkronkan antar *tool* yang dibutuhkan untuk membangun script pada NS. Langkah ini dapat dilakukan dengan cara mengarahkan direktori terminal kedalam direktori NS *$cd ns-allinone-2.35/ns-2.35* kemudian tekan enter maka secara langsung direktori yang terdapat pada terminal akan mengarah pada folder /ns-2.35. Untuk melakukan proses validasi terhadap NS, ketiklah perintah *$./validate* pada terminal, maka dengan perintah tersebut secara otomatis akan melakukan validasi terhadap semua *tool* yang digunakan oleh NS. Proses validasi membutuhkan waktu yang lumayan lama, bisa diperkirakan 30 menit atau bisa jadi 60 menit sesuai dengan performa PC atau laptop yang digunakan.

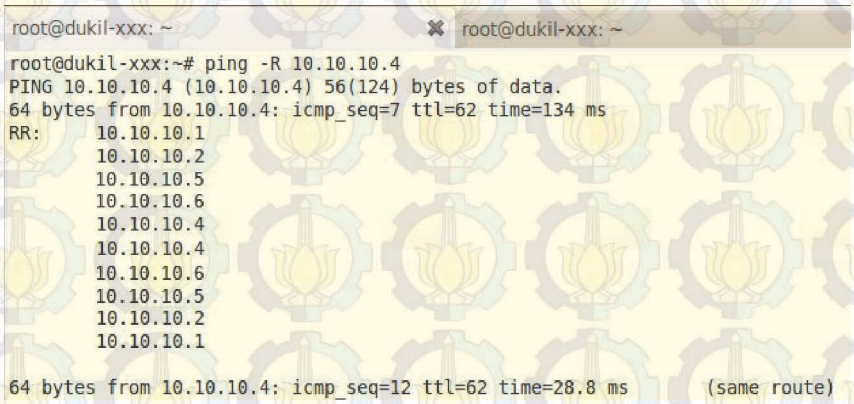
2. Gambar tampilan Iperf pada sisi penerima.

```
^Croot@dukil-xxx:~# iperf -s
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[  4] local 10.10.10.1 port 5001 connected with 10.10.10.2 port 42062
[ ID] Interval       Transfer     Bandwidth
[  4]  0.0- 2.3 sec  32.0 KBytes    114 Kbits/sec
[  5] local 10.10.10.1 port 5001 connected with 10.10.10.2 port 45501
[  5]  0.0- 8.8 sec  32.0 KBytes  29.7 Kbits/sec
[  4] local 10.10.10.1 port 5001 connected with 10.10.10.2 port 45502
[  4]  0.0- 1.5 sec  32.0 KBytes    176 Kbits/sec
[  5] local 10.10.10.1 port 5001 connected with 10.10.10.2 port 45503
[  5]  0.0- 2.4 sec  32.0 KBytes    109 Kbits/sec
[  4] local 10.10.10.1 port 5001 connected with 10.10.10.2 port 45504
[  4]  0.0-17.9 sec  32.0 KBytes  14.7 Kbits/sec
[  5] local 10.10.10.1 port 5001 connected with 10.10.10.2 port 45505
[  5]  0.0- 1.8 sec  32.0 KBytes    144 Kbits/sec
[  4] local 10.10.10.1 port 5001 connected with 10.10.10.2 port 45506
[  4]  0.0- 1.3 sec  32.0 KBytes    199 Kbits/sec
[  5] local 10.10.10.1 port 5001 connected with 10.10.10.2 port 45507
[  5]  0.0- 3.5 sec  32.0 KBytes  75.9 Kbits/sec
[  4] local 10.10.10.1 port 5001 connected with 10.10.10.2 port 45508
```

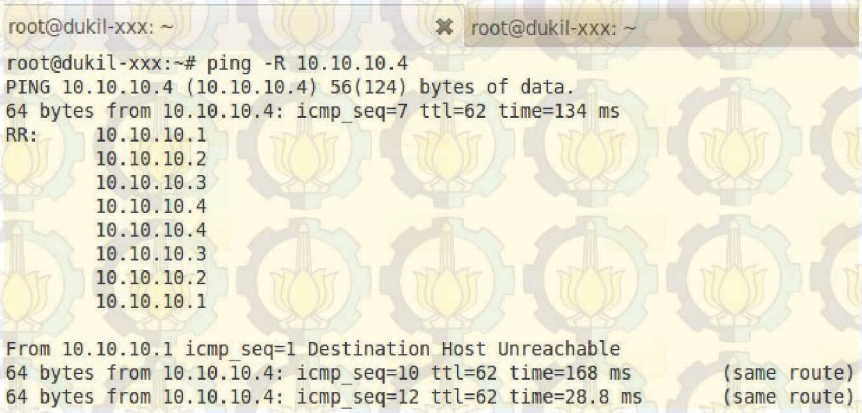
3. **Gambar tampilan Iperf pada sisi pengirim**



4. **Ping –R untuk 4 hop alternatif *route***



5. **Ping –R untuk 4 hop *primary route***



6. Script TCL Topologi VMeS

```
# Simulasi 4-5 node Pada Protokol Routing AODV-BR

# Define options
set val(chan)     Channel/WirelessChannel    ;# Type Chanel
set val(prop)     Propagation/TwoRayGround   ;# Model radio-propagation
set val(netif)    Phy/WirelessPhy            ;# Jenis Interface Jaringan
set val(mac)      Mac/802_11                 ;# Protokol MAC / Datalink
set val(ifq)      Queue/DropTail/PriQueue    ;# Type Interface Antrian
set val(ll)       LL                         ;# Type Layer Link
set val(ant)      Antenna/OmniAntenna        ;# Model Antena
set val(ifqlen)   30                         ;# Max. Paket pada ifq
set val(nn)       5                          ;# Jumlah Mobile Node
set val(rp)       AODV                       ;# Jenis Protokol Routing
set val(x)        500                        ;# Dimensi X pada Pemetaan
set val(y)        450                        ;# Dimensi Y pada Pemetaan
set val(stop)    500                ;# Batas Waktu Simulasi Selesai (ms)

      set ns           [new Simulator]
      set tracefd      [open dukil.tr w]
      set windowVsTime2 [open dukil.tr w]
      set namtrace     [open dukil.nam w]

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

      # Mengatur Pemetaan Objek
      set topo      [new Topography]

      $topo load_flatgrid $val(x) $val(y)

      create-god $val(nn)

      #
      #  membuat mobile node nn dan melampirkan pada saluran
      #

      # Konfigurasi Node
            $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) \
                  -channelType $val(chan) \
                  -topoInstance $topo \
                  -agentTrace ON \
                  -routerTrace ON \
                  -macTrace OFF \
                  -movementTrace ON

          for {set i 0} {$i < $val(nn) } { incr i } {
              set node_($i) [$ns node]
```

```
                $node_($i) set X_ [ expr 10+round(rand()*490) ]
                $node_($i) set Y_ [ expr 10+round(rand()*390) ]
                $node_($i) set Z_ 0.0
            }

            for {set i 0} {$i < $val(nn) } { incr i } {
                $ns at [ expr 1+round(rand()*60) ] "$node_($i) setdest [
expr  10+round(rand()*490)  ]  [  expr  10+round(rand()*390)  ]  [  expr
2+round(rand()*15) ]"
             }

        #Konfigurasi Posisi Awal node mobile
        $node_(0) set X_ 10.0
        $node_(0) set Y_ 10.0
        $node_(0) set Z_ 0.0


        $node_(1) set X_ 75.0
        $node_(1) set Y_ 75.0
        $node_(1) set Z_ 0.0

        $node_(2) set X_ 150.0
        $node_(2) set Y_ 150.0
        $node_(2) set Z_ 0.0

        $node_(3) set X_ 225.0
        $node_(3) set Y_ 225.0
        $node_(3) set Z_ 0.0

        $node_(4) set X_ 300.0
        $node_(4) set Y_ 300.0
        $node_(4) set Z_ 0.0


        # Arah pergerakan Node
        $ns at 10.0 "$node_(0) setdest 10.0 10.0 0.0"
        $ns at 12.0 "$node_(1) setdest 100.0 90.0 3.0"
        $ns at 12.0 "$node_(2) setdest 125.0 140.0 3.0"
        $ns at 12.0 "$node_(3) setdest 260.0 240.0 3.0"
        $ns at 15.0 "$node_(4) setdest 300.0 300.0 10.0"
        #$ns at 60.0 "$node_(5) setdest 150.0 70.0 2.0"
        #$ns at 90.0 "$node_(6) setdest 380.0 150.0 8.0"
        #$ns at 42.0 "$node_(7) setdest 200.0 100.0 15.0"
        # $ns at 55.0 "$node_(8) setdest 50.0 275.0 5.0"
        # $ns at 19.0 "$node_(9) setdest 250.0 250.0 7.0"
        # $ns at 90.0 "$node_(10) setdest 150.0 150.0 20.0"

        # Mengatur Koneksi menggunakan agent TCP antar node pengirim dan
penerima
        set tcp [new Agent/TCP/Newreno]
        $tcp set class_ 2
        set sink [new Agent/TCPSink]
        $ns attach-agent $node_(0) $tcp
        $ns attach-agent $node_(4) $sink
        $ns connect $tcp $sink
        set cbr [new Application/Traffic/CBR]
```

```
        $cbr attach-agent $tcp
        $cbr set packetSize_ 128K
        $cbr set random_ false
        $ns at 10.0 "$cbr start"
        $ns at 480.0 "$cbr stop"

        #set tcp [new Agent/TCP/Newreno]
        #$tcp set class_ 2
        #set sink [new Agent/TCPSink]
        #$ns attach-agent $node_(5) $tcp
        #$ns attach-agent $node_(0) $sink
        #$ns connect $tcp $sink
        #set ftp [new Application/FTP]
        #$ftp attach-agent $tcp
        #$ns at 10.0 "$ftp start"

        # Printing the window size
        proc plotWindow {tcpSource file} {
        global ns
        set time 0.1
        set now [$ns now]
        set cwnd [$tcpSource set cwnd_]
        puts $file "$now $cwnd"
        $ns at [expr $now+$time] "plotWindow $tcpSource $file" }
        $ns at 10.1 "plotWindow $tcp $windowVsTime2"

        # Mendefinisikan posisi node pada ANIMASI (nam)
        for {set i 0} {$i < $val(nn)} { incr i } {

        # Memberikan ukuran besar 30 pada node di file nam
        $ns initial_node_pos $node_($i) 30
        }

        # Konfigurasi akhir sumulasi
        for {set i 0} {$i < $val(nn) } { incr i } {
            $ns at $val(stop) "$node_($i) reset";
        }

        # Konfigurasi akhir simulasi dan animasi
        $ns at $val(stop) "$ns nam-end-wireless $val(stop)"
        $ns at $val(stop) "stop"
        $ns at 490 "puts \"end simulation\" ; $ns halt"
        proc stop {} {
            global ns tracefd namtrace
            $ns flush-trace
            close $tracefd
            close $namtrace
            #exec nam dukil.nam &
            #exit 0
        }

$ns run
```