ANALISIS PENGARUH KINERJA *ROUTING PROTOCOL* AODV *(AD HOC ON-DEMAND VECTOR)* DAN DSDV *(DESTINATION SEQUENCED DISTANCE VECTOR)* TERHADAP KONSUMSI ENERGI *NODE* PADA JARINGAN MANET

SKRIPSI

untuk memenuhi salah satu persyaratan

mencapai derajat Sarjana S1

Disusun oleh:

Nazarudin Faruk Assidiq

14524094

Jurusan Teknik Elektro

Fakultas Teknologi Industri

Universitas Islam Indonesia

Yogyakarta

2018

LEMBAR PENGESAHAN

LEMBAR PENGESAHAN

ANALISIS PENGARUH KINERJA *ROUTING PROTOCOL* AODV *(AD HOC ON DEMAN VECTOR)* DAN DSDV *(DESTINATION SEQUENCED DISTANCE VECTOR)* TERHADAP KONSUMSI ENERGI *NODE* PADA JARINGAN MANET

TUGAS AKHIR

Diajukan sebagai Salah Satu Syarat untuk Memperoleh
Gelar Sarjana Teknik
pada Program Studi Teknik Elektro
Fakultas Teknik Industri
Universitas Indonesia

Disusun oleh:

Nazarudin Faruk Assidiq
14524094

Yogyakarta, 13 Agustus 2018

Menyetujui

Pembimbing 1

Ida Nurcahyani,S.T,M.Eng
155240104

LEMBAR PENGESAHAN

LEMBAR PENGESAHAN

SKRIPSI

ANALISIS PENGARUH KINERJA *ROUTING PROTOCOL* AODV *(AD HOC ON DEMAND VECTOR)* DAN *DSDV (DESTINATION SEQUENCED DISTANCE VECTOR)* TERHADAP KONSUMSI ENERGI *NODE* PADA JARINGAN MANET

Dipersiapkan dan disusun oleh:

Nazarudin Faruk Assidiq

14524094

Telah dipertahankan di depan dewan penguji

Pada tanggal: 20 Agustus 2018

Susunan dewan penguji

Ketua Penguji : **Ida Nurcahyani,S.T.,M.Eng.,** _____

Anggota Penguji 1: Tito Yuwono, S.T., M.Sc., _____

Anggota Penguji 2: Elvira Sukma Wahyuni, S.Pd.,M.Eng., _____

Skripsi ini telah diterima sebagai salah satu persyaratan untuk memperoleh gelar Sarjana

Tanggal: 11 September 2018

Ketua Program Studi Teknik Elektro

Yusuf Aziz Amrullah S.T., M.Eng., Ph.D.

045240101

PERNYATAAN

PERNYATAAN

Dengan ini Saya menyatakan bahwa:

1. Skripsi ini tidak mengandung karya yang diajukan untuk memperoleh gelar kesarjanaan di suatu Perguruan Tinggi, dan sepanjang pengetahuan Saya juga tidak mengandung karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis diacu dalam naskah ini dan disebutkan dalam daftar pustaka.
2. Informasi dan materi Skripsi yang terkait hak milik, hak intelektual, dan paten merupakan milik bersama antara tiga pihak yaitu penulis, dosen pembimbing, dan Universitas Islam Indonesia. Dalam hal penggunaan informasi dan materi Skripsi terkait paten maka akan diskusikan lebih lanjut untuk mendapatkan persetujuan dari ketiga pihak tersebut diatas.

Yogyakarta, 20 Agustus 2018

Nazarudin Faruk Assidiq

ii

KATA PENGANTAR

Assalamu'alaikum warahmatullahi wabarakatuh,

Segala puji syukur penulis panjatkan ke hadirat Allah Subhanahu wa ta'ala yang telah melimpahkan rahmat dan hidayah-Nya, sehingga penulis bisa menyelesaikan skripsi ini yang berjudul "Analisis Pengaruh Kinerja Routing Protocol AODV *(Ad hoc On-Demand Vector)* dan DSDV (*Distance Sequenced Destination Vector)* Terhadap Konsumsi *Node* pada Jaringan MANET" sebagai syarat salah satu untuk bisa menempuh ujian sarjana teknik pada Fakultas Teknologi Industri (FTI) Program Studi Teknik Elektro di Universitas Islam Indonesia.

Dalam pengerjaan skripsi ini penulis menyadari bahwa banyak pihak yang sangat membantu dalam segala hal. Atas dukungan moral dan materil yang diberikan dalam penyusunan skripsi ini, penulis mengucapkan banyak terima kasih kepada:

1. **Keluarga dirumah, Basuki, Mardiah, Rera Wulandari,** dan **Roro Nur Azaza** selaku keluarga yang tiada henti-hentinya mendukung dan mendoa'kan yang terbaik untuk penulis.
2. **Bapak Yusuf Aziz Amrullah S.T., M.Eng., Ph.D.,** selaku Ketua Jurusan Teknik Elektro, Fakultas Teknik Industri, Universitas Islam Indonesia.
3. **Ibu Ida Nurcahyani, S.T., M.Eng**. Selaku Dosen Pembimbing yang telah membimbing, membantu, mendampingi, serta memberi motivasi dan arahan sehingga penulis dapat menyelesaikan tugas akhir ini.
4. **Seluruh dosen Jurusan Teknik Elektro,** atas ilmu dan bimbingan yang diberikan kepada penulis selama ini.
5. **Ryandy Ghonim Asgar, Muhammad Iqbal, Ersa Cucun Alfindo,** selaku sahabat penulis yang selama awal masa kuliah hingga sekarang selalu menghibur.
6. **Teknik Elektro UII Angkatan 14, PASTEL 14 UII, Teman Kos Pondok Keluarga** karena telah mendukung dan menemani kegiatan di kampus dan diluar kampus dari awal kuliah hingga sekarang.

ARTI LAMBANG DAN SINGKATAN

MANET	:	*Mobile Ad-hoc Network*
AODV	:	*Ad-Hoc On Demand Distance Vector*
DSDV	:	*Destination-Sequenced Distance Vector*
OLSR	:	*Optimized Link State Routing*
NS	:	*Network Simulator*
RREQ	:	*Route Request*
RREP	:	*Route Reply*
RRER	:	*Route Error*
UDP	:	*User Datagram Program*
Mbps	:	*Megabit per second*

ABSTRAK

Mobile Ad-hoc Network (MANET) merupakan jaringan nirkabel yang tidak memiliki *router* terpusat dan mempunyai banyak *node* dimana masing-masing *node* memiliki fungsi sebagai *host* untuk terhubung ke jaringan nirkabel dan juga berfungsi sebagai *mobile router* yang akan melanjukan paket data yang tersedia kepada *node* lainnya. Masing-masing *node* bersifat *mobile*, sehingga membutuhkan energi untuk saling berkomunikasi. *Routing* merupakan suatu fungsi dari lapisan *network* untuk menentukan rute dari *node* pengirim menuju ke *node* penerima. MANET memiliki tiga protokol *routing* yaitu *Table-Driven routing protocols (proactive), On- Demand routing protocols (reactive)* dan *Hybrid routing protocol*. AODV *(Ad hoc On-Demand Distance Vector)* merupakan salah satu contoh routing reaktif dan DSDV (*Destination Sequenced Distance Vector)* merupakan contoh dari routing proaktif. Penelitian ini bertujuan untuk mengetahui kinerja *routing protocol* AODV dan DSDV terhadap konsumsi energi *node* pada jaringan MANET. Paramater yang dianalisis adalah konsumsi energi dan energi yang tersisa. Penelitian ini menggunakan dua skenario yaitu penambahan *node* dan penambahan luas area. Pada hasil analisis didapatkan kesimpulan bahwa pada skenario penambahan *node* konsumsi energi pada AODV lebih hemat dari DSDV dengan selisih 0,887 *Joules* dan pada penambahan luas area konsumsi energi *node* pada AODV lebih hemat daripada DSDV dengan rasio perbandingan konsumsi energi sebesar 0,1%.

Kata kunci : MANET, AODV, DSDV, konsumsi energi, energi yang tersisa

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1
PENDAHULUAN

1.1 Latar Belakang Masalah

Kemajuan teknologi yang memberikan banyak pilihan baru dimana jaringan nirkabel dapat membangun dan mengelola jaringannya sendiri. *Mobile Ad-hoc Network* (MANET) merupakan jaringan nirkabel yang tidak memiliki *router* terpusat dan mempunyai banyak *node* dimana masing-masing *node* memiliki fungsi sebagai *host* untuk terhubung ke jaringan nirkabel dan juga berfungsi sebagai *mobile router* yang akan melanjutkan paket data yang tersedia kepada *node* lainnya. Jenis *routing protocol* yang digunakan akan sangat berpengaruh terhadap kinerja jaringan MANET.

Routing merupakan suatu fungsi dari lapisan *network* untuk menentukan rute dari *node* pengirim menuju ke *node* penerima. Peran protocol *routing* di MANET untuk mengatasi *node-node* berkomunikasi dengan sumber daya terbatas. *MANET* memiliki tiga protokol *routing* yaitu *Table Driven routing protocols (proactive), On Demand routing protocols (reactive)* dan *Hybrid routing protocol*. Beberapa karakteristik pada MANET adalah *node mobility* dan topologi yang dinamis. Pada MANET *node-node* selalu bergerak bebas, setiap *node* dapat saling berkomunikasi dalam lingkup sinyal karena masing-masing *node* dapat memancarkan sinyal dalam radius tertentu. MANET juga memiliki sifat topologi yang dinamis karena dalam MANET tidak dibutuhkannya *access point* dan *node-node* yang selalu aktif bergerak kemana saja maka topologi jaringan pada MANET tidak dapat di prediksi.

Beberapa masalah yang sering terjadi pada jaringan MANET adalah sumber energi *node* yang terbatas. Karena masing-masing *node* yang bersifat *mobile* sehingga membutuhkan energi untuk saling berkomunikasi dengan *node* lainnya. Karena ketika energi pada *node-node* habis, *node-node* tidak bisa melakukan komunikasi dengan node lainnya untuk mengirim dan menerima paket data. Maka dari itu dalam pemilihan *routing protocol* sebaiknya mempertimbangkan konsumsi energi pada *node* dahulu sehingga dapat memastikan *node-node* dapat bertahan ketika saling berkomunikasi.

Penelitian [1] mengatakan bahwa konsumsi energi dan masa pakai *routing protocol* DSDV lebih baik daripada *routing protocol* AODV karena disebabkan oleh mobilitas *node* dan jumlah paket yang diproses di setiap protokol.

*Ad Hoc On -Demand Distance Vecto*r a (AODV) merupakan kombinasi dari *Dynamic Source Routing (DSR)* dan *Destination Sequenced Distance Vector* (DSDV). AODV merupakan jenis *routing protocol on demand*. Protokol ini menggunakan mekanisme dari DSR yaitu *Route Discovery* dan *Route Maintenance*, kemudian melibatkan *hop–by-hop routing, periodic beacon, sequenced numbers* pada DSDV [2].

Penelitian ini berfokus kepada bagaimana pengaruh kinerja *routing protocol* AODV dan DSDV terhadap konsumsi energi pada jaringan *MANET* dengan menggunakan perangkat lunak *Network Simulator* NS-3.27. Penelitian ini bertujuan untuk menganalisis dan membandingkan konsumsi energi dan energi yang tersisa oleh *routing protocol* AODV dan DSDV pada jaringan MANET.

1.2 Rumusan Masalah

1. Bagaimana pengaruh *routing protocol* AODV dan DSDV terhadap konsumsi energi pada *MANET*?
2. Diantara *routing protocol* AODV dan DSDV manakah yang paling hemat energi?

1.3 Batasan Masalah

1. Penelitian pada tugas akhir ini hanya membahas tentang pengaruh konsumsi energi pada *routing protocol* AODV dan DSDV pada jaringan *MANET*. Faktor lain seperti *QoS, throughput* tidak diperhitungkan.
2. Menggunakan *routing protocol* AODV dan DSDV.
3. Jumlah *node* yang digunakan adalah 50 *dan 100 node.*
4. Parameter yang dianalisis adalah konsumsi energi dan energi yang tersisa.
5. Simulasi dilakukan menggunakan perangkat lunak *Network Simulator (NS)* 3.27.

1.4 Tujuan Penelitian

Tujuan dari penelitian ini adalah untuk mengetahui kinerja routing protocol AODV dab DSDV terhadap konsumsi energi node pada jaringan MANET.

1.5 Manfaat Penelitian

Hasil dari penelitian ini diharapkan dapat digunakan sebagai pertimbangan untuk memilih *routing protocol* yang digunakan pada jaringan MANET.

BAB 2
TINJAUAN PUSTAKA

2.1 Studi Literatur

Pada bagian tinjauan pustaka ini peneliti mengumpulkan beberapa artikel dan jurnal yang berkaitan dengan penelitian yang akan dilakukan peneliti yaitu kinerja *routing protocol* terhadap konsumsi energi. Peneliti akan menguraikan beberapa jurnal yang telah disurvei oleh peneliti.

Salah satu penelitian yang dilakukan oleh Aloysius Tri Sulistyo Putranto [3] menganalisis penggunaan energi AODV dan DSDV pada jaringan MANET dengan parameter yang digunakan adalah *death of node* dan nilai *throughput*. Pengambilan data menggunakan perangkat lunak simulator NS2. Penelitian ini menggunakan kecepatan, *node*, dan penambahan area sebagai skenario yang digunakannya. Untuk hasil analisis, pada *routing protocol* AODV lebih hemat baterai dibandingkan DSDV dalam hal kecepatan, *node*. Hal tersebut terjadi karena AODV merupakan *routing* reaktif dimana *node* yang paling banyak menggunakan baterai adalah *sourcenya*. Untuk hasil anilisis *throughputnya*, AODV memiliki *throughput* yang lebih baik daripada DSDV pada area yang lebih kecil. Tetapi untuk area yang cukup besar dan kecepatan tinggi DSDV memiliki *throughput* yang lebih baik [3]. DSDV terlihat lebih hemat ketika *node* semakin bertambah. Hal itu disebabkan karena *node* yang banyak melakukan *control message* daripada mengirimkan data [3].

Selanjutnya adalah penelitian yang dilakukan oleh Justisia Satiti, Indrarini Dyah Irawati, Leanna Vidya Yovita [4] mengenai analisis perbandingan protokol *routing* AODV dan DSDV pada *wireless sensor network*. Simulasi ini menggunakan simulator NS-2.35 berstandarkan IEEE 802.15.4 *(Zigbee)*. Skenario yang digunakan oleh peneliti adalah dengan perubahan jumlah *node*, penambahan jumlah *node* ZED yang aktif secara bersamaan [4]. Parameter yang digunakan peneliti adalah *delay*, *throughput*, *routing overhead*, dan *energy consumption*. Untuk hasil analisis adalah AODV memiliki *delay* yang lebih stabil dibandingkan DSDV ketika menghadapi kepadatan trafik. Nilai *routing overhead* yang terjadi pada AODV juga lebih kecil daripada DSDV dan *energy consumption* yang dikeluarkan oleh AODV lebih sedikit daripada DSDV. Maka pada penelitian ini dapat disimpulkan bahwa protokol *routing* AODV lebih cocok digunakan pada *wireless sensor network.*

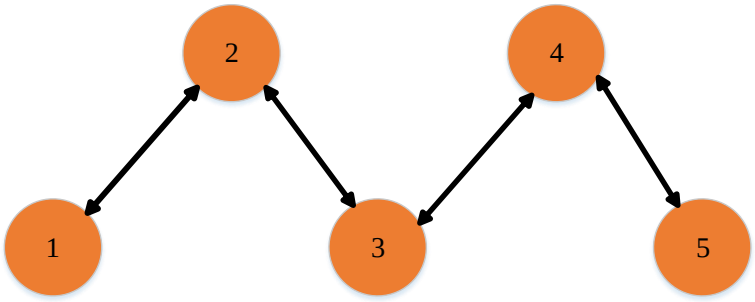
Penelitian yang dilakukan oleh Muhammad Irfan Denatama dan Doan Perdana, Ridha Muldina Negara pada tahun 2016 [5] mengenai analisis perbandingan kinerja protokol *routing* DSDV dan OLSR untuk perubahan kecepatan mobilitas pada standar IEEE 802.11ah. parameter yang diamati adalah *throughput, packet delivery ratio,* rata-rata *delay,* dan konsumsi energi. Simulasi ini menggunakan NS3. Untuk hasil analisis, pada skenario perubahana kecepatan protokol *routing* OLSR memiliki kinerja yang lebih baik daripada DSDV. Nilai rata-rata *throughput* yang didapat oleh OLSR adalah 28400 *Bps* dan DSDV sebesar 2934 *Bps*. Nilai rata-rata PDR dari OLSR lebih bagus daripada DSDV dan juga nilai rata-rata *delay* OLSR lebih kecil daripada DSDV. Untuk rasio perbandingan konsumsi energi antara OLSR dan DSDV pada skenario perubahan kecepatan *node* adalah 1,48 % [5].

Penelitian ini bertujuan untuk mengetahui konsumsi energi dan energi yang tersisa terhadap kinerja *routing protocol* AODV dan DSDV pada jaringan MANET. Penelitian ini menggunakan dua skenario yaitu penambahan *node* dan penambahan luas area. Melalui penelitian ini diharapkan dapat diketahui jenis *routing protocol* yang hemat energi diantara *routing protocol* AODV dan DSDV.

2.2 Tinjauan Teori

2.2.1 Mobile Ad-hoc Network

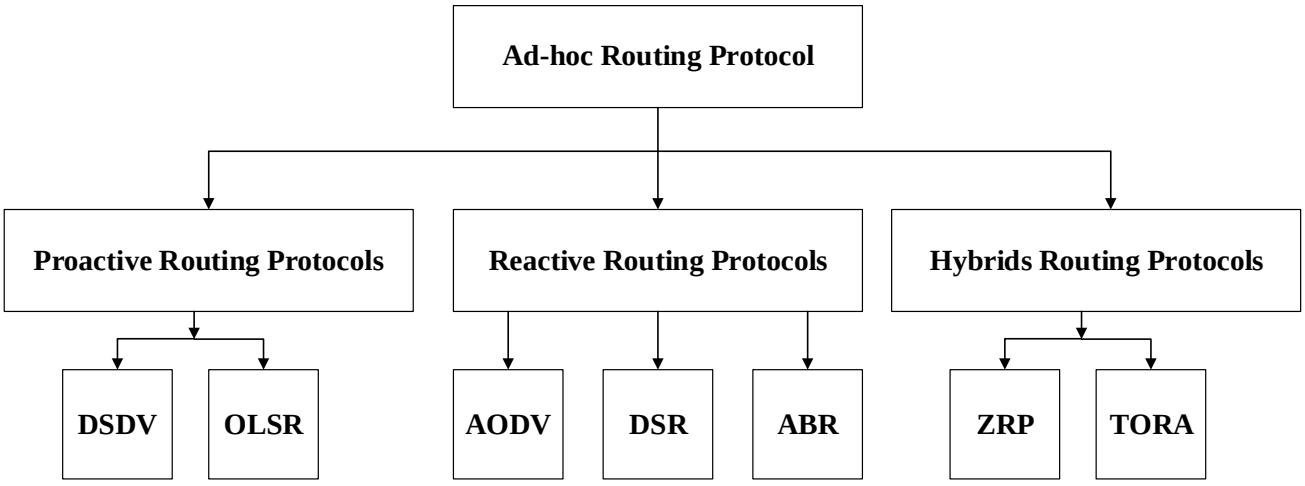
Mobile Ad-hoc Network (MANET) adalah gabungan *node-node* yang dapat saling berkomunikasi tanpa memerlukan infrastruktur. Node dapat berperan sebagai host yang terhubung ke jaringan dan node dapat berperan sebagai pengirim dan penerima untuk meneruskan informasi sehingga tidak diperlukan base station. *MANET* memiliki karakteristik topologi yang dinamis yang menyebabkan *node-node* pada *MANET* bergerak bebas kapan saja dan kemana saja. Penggunaan *routing protocol* pada MANET adalah untuk membantu lancarnya komunikasi antar *node*. Contoh pengaplikasian jaringan *Ad Hoc* pada saat kegiatan militer, dimana *node-node* digunakan sebagai penghubung menggunakan jaringan wireless seperti *celluler, wi-fi, Bluetooth*.



Gambar 2.1 *Mobile Ad Hoc Network*

2.2.2 Routing protocol

Routing merupakan suatu fungsi dari lapisan *network* untuk menentukan rute dari *node* pengirim menuju ke *node* penerima. Fungsi lain dari *routing protocol* pada *MANET* adalah dapat melakukan adaptasi terhadap perubahan topologi dan trafik yang disebabkan oleh pergerakan *node* secara acak. Pada Gambar 2.2 adalah klasifikasi routing protocol pada MANET.



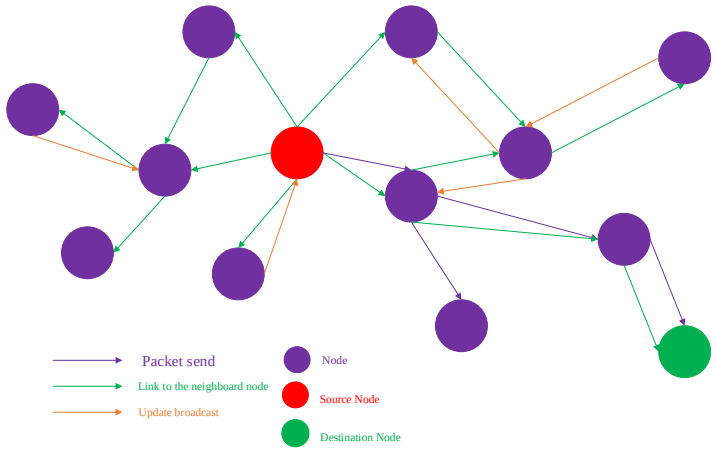
Gambar 2.2 Klasifikasi *Routing Protocol*

1. Proaktif atau *table-driven protocol*

Proaktif *routing* berupaya untuk menyediakan informasi *routing* yang selalu *up to date* di setiap *node*. Pada setiap *node* wajibkan untuk mempunyai satu atau lebih tabel untuk menyimpan informasi *routing*. Setiap *node* akan merespon perubahan dalam topologi jaringan dengan cara menyebarkan informasi yang *up to date* ke semua *node* yang ada di jaringan. Contoh dari *table-driven routing* adalah DSDV, OLSR..

Destination Sequenced Distance Vector (DSDV) merupakan *Routing Protocol routing* proaktif atau *table-driven* untuk mengatasi *routing loop*. Pada DSDV, mengirimkan pesan ke jaringan menggunakan *sequenced number*. *Sequenced number* juga digunakan pada saat adanya perubahan dalam jaringan, hal ini terjadi karena DSDV menggunakan *table routing* yang juga merupakan sifat dari *routing* proaktif yang akan selalu memperbarui informasi secara periodik.

Pada saat melakukan update rute, DSDV menggunakan time-driven dan event driven. Time-driven merupakan *node-node* akan saling bertukar informasi dengan *node-node* lainnya untuk mendapatkan informasi yang terbaru secara periodik. Sedangkan event driven terjadi ketika adanya pembaharuan yang penting, maka *node* lainnya akan digerakkan oleh trigger atau fenomena tertentu untuk mengirimkan informasi dari *routing* table yang berubah [3]. DSDV membutuhkan konsumsi energi yang cukup besar karena sifat DSDV yang merupakan *routing* proaktif dimana DSDV harus selalu memperbaharui *routing table* secara periodik. Gambar 2.3 menjelaskan cara kerja DSDV [6].



Gambar 2.3 Cara kerja DSDV

Dalam Gambar 2.3 menjelaskan cara kerja dari *routing* DSDV. DSDV dikenal dimana *node* sumber yang mengirim paket ke jalur disimpan dalam *routing table* dan

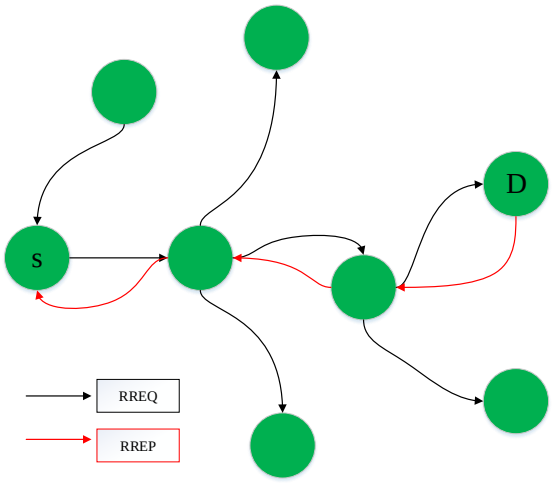
node lainnya melakukan pembaruan dari *routing table* mereka. Pada waktu pengiriman paket DSDV sangat kecil daripada *routing* reaktif karena pada routing reaktif *node* pertama melakukan pencarian rute daripada mengirimkan paket. Pada pemeliharaan tabel dalam DSDV sangat mahal ketika ukuran jaringan yang terlalu besar [6].

2. Reaktif atau *on-demand routing protocol*

Pada kelas ini, *routing* dibuat ketika *node* sumber membutuhkannya. Pada saat *node* sumber membutuhkan *routing* ke suatu *node* tujuan, maka akan dilakukan proses *route discovery* dalam jaringan. Setelah didapatkan rute, dilakukan proses *route maintenance*. Contoh *on-demand routing* adalah: AODV, DSR, ABR.

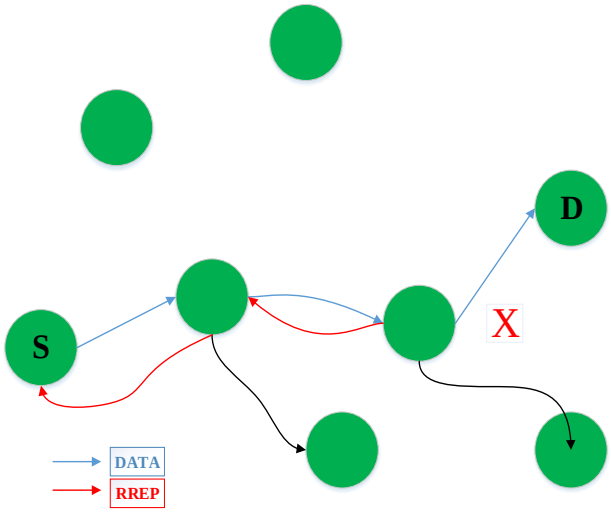
Ad hoc On demand Distance Vector (AODV) merupakan *distance vector routing protocol* reaktif yang menggunakan metodologi *hop-to-hop*. Protokol AODV hanya bekerja jika ada *request* suatu rute. Ketika sebuah *node* ingin mengetahui rute ke tujuan tertentu, maka akan tercipta permintaan *route*. Permintaan *route* akan dilanjutkan oleh *node intermediate* yang membuat rute terbalik untuk dirinya sendiri untuk tujuan. Ketika permintaan mencapai *node* dengan rute ke tujuan, maka akan menciptakan lagi balasan yang berisi berupa jumlah *hop* yang diperlukan untuk mencapai ke tujuan. Semua *node* yang berpatisipasi akan meneruskan balasan ini ke *node* sumber yang akan membuat rute menuju ke tujuan. Rute ini dibuat dari setiap *node* dari sumber ke tujuan adalah dalam keadaan *hop-by-hop* dan bukan seluruh rute seperti pada *routing* sumber. Maka dari itu sifat dari AODV sendiri adalah *single route*, yaitu paket akan dikirim hanya melalui satu rute. Jika ada kerusakan pada saat melewati rute lain, maka dilakukan pencarian rute baru. AODV memiliki *route discovery* dan *route maintenance*. Pada *Route Discovery* terdapat pesan *Route Request* (RREQ) dan *Route Reply* (RREP). *Route Maintenance* terdapat pesan *Route Error* (RRER).

Ketika ada permintaan pengiriman paket data, maka *route request* (RREQ) akan disebarkan ke *node* yang ada disekitarnya. Jika RREQ mencapai *node* dengan rute tujuan, maka selanjutnya akan diteruskan oleh *Route Reply* (RREP) ke *node* sumber. ketika *node* telah menerima RREP, maka *node* tersebut akan mengirimkan RREP lagi ke *destination sequenced number*. Jika pesan benar maka akan dilanjutkan RREP [7]. Gambar 2.3 menjelaskan cara kerja dari RREQ.



Gambar 2.4 Cara kerja RREQ

Selama pemeliharaan rute, route error akan bekerja ketika terjadinya pemutusan rute atau rute tidak dapat terdeteksi. Maka dilakukan route maintenance. RRER akan menyebarkan kesalahan rute kepada *node* sumber [7]. Gambar 2.4 adalah cara kerja RREP pada AODV.



Gambar 2.5 Cara kerja RREP

2.2.3 Network Simulator 3

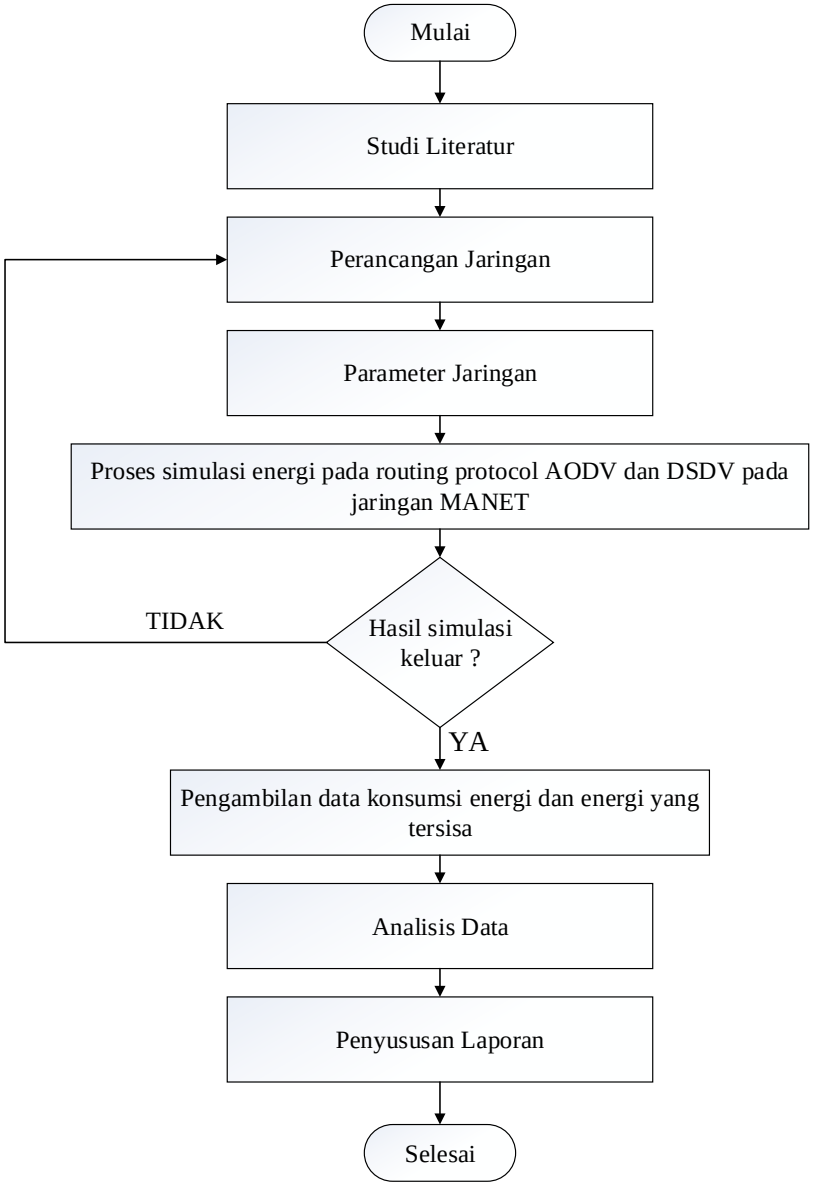
NS3 adalah *network simulator* yang bersifat *open source* yang digunakan untuk mensimulasikan jaringan komputer. Dikembangkan dengan Bahasa C++, *NS3* pertama

kali dikenalkan pada tahun 2008 [8]. Tujuan dari adanya proyek *NS3* ini adalah untuk mengembangkan, lingkungan simulasi terbuka untuk penelitian dalam bidang *networking*. *NS3* bukan merupakan kelanjutan dari *NS2*. Salah satu alasan kenapa menggunakan *NS3* adalah untuk belajar bagaimana jaringan bekerja. Banyak fitur-fitur yang terdapat pada *NS3* untuk mensimulasikan jaringan. Salah satu contoh adalah jaringan pada MANET. Karenanya, *NS3* bersifat *open source*. Maka kita bisa menambah fungsi-fungsi baru *core* dalam *NS3*. Salah satu fitur yang membedakan *NS3* dengan *tools* lainnya adalah *NS3* beroperasi pada sistem *Linux*, walaupun *support* juga pada *FreeBSD*, *Cygwin* (untuk *windows*). Untuk melihat hasil dari simulasi dengan menggunakan *NS3* dapat ditampilkan berupa grafik sehingga memudahkan ketika menganalisis hasil [8].

BAB 3
METODOLOGI

3.1 Alur Penelitian

Gambar 3.1 menjelaskan alur penelitian yang digunakan oleh peneliti. Alur penelitian yang terdiri dari studi literatur, perancangan jaringan, parameter jaringan, proses simulasi, pengambilan data, analisis data, penyusunan program. Pada tahap perancangan jaringan, akan dijelaskan lebih pada sub-bab perancangan program.



Gambar 3.1 *Flowchart* Penelitian

3.1.1 Studi Literatur

Tahap pertama adalah peneliti mencari beberapa jurnal yang terkait dengan judul penelitian untuk dijadikan acuan penelitian dan menambah wawasan peneliti pada saat merancang skenario.

3.1.2 Perancangan Skenario

Tahap kedua peneliti perancangan Skenario untuk melakukan simulasi. Perancangan ini menggunakan NS-3.27. Pada bagian perancangan program akan dijelaskan tahap-tahap perancangan program menggunakan NS-3.27. terdapat dua skenario yaitu skenario penambahan *node* dan skenario penambahan luas area. Pada skenario yang pertama adalah skenario penambahan *node*. *Node* yang akan digunakan adalah 50 *node* dan 100 *node* dengan luas area yang sama yaitu 100x100 meter. Pada skenario yang kedua adalah dengan penambahan luas area. Luas area yang digunakan adalah 300x300 meter dengan menggunakan *node* yang sama yaitu 50, dan 100 *node.*

3.1.3 Menentukan parameter jaringan

Jaringan yang digunakan peneliti adalah jaringan MANET. Parameter jaringan yang digunakan peneliti adalah konsumsi energi dan energi yang tersisa. Simulasi berjalan selama lima menit dengan energi awal yang diberikan adalah 1147 *Joules.*

3.1.4 Proses Simulasi

Pada proses simulasi ini peneliti menjalankan program dengan skenario yang telah dirancang oleh peneliti pada tahap merancang skenario. Simulasi berjalan selama lima menit dan energi awal yang diberikan adalah 1147 *Joules*.

3.1.5 Hasil Simulasi

Tahap selanjutnya adalah hasil simulasi. Setelah dilakukan simulasi dengan skenario yang digunakan oleh peneliti, jika hasil simulasi keluar maka selanjutnya adalah pengambilan data konsumsi energi dan energi yang tersisa. Jika simulasi tidak keluar, maka dilakukan proses simulasi ulang hingga hasil keluar.

3.1.6 Pengambilan data

Setelah simulasi selesai dan hasil simulasi keluar, peneliti melakukan pengambilan data dengan parameter yang sudah ditentukan sebelumnya pada tahap parameter jaringan yaitu konsumsi energi dan energi yang tersisa.

3.1.7 Analisis data

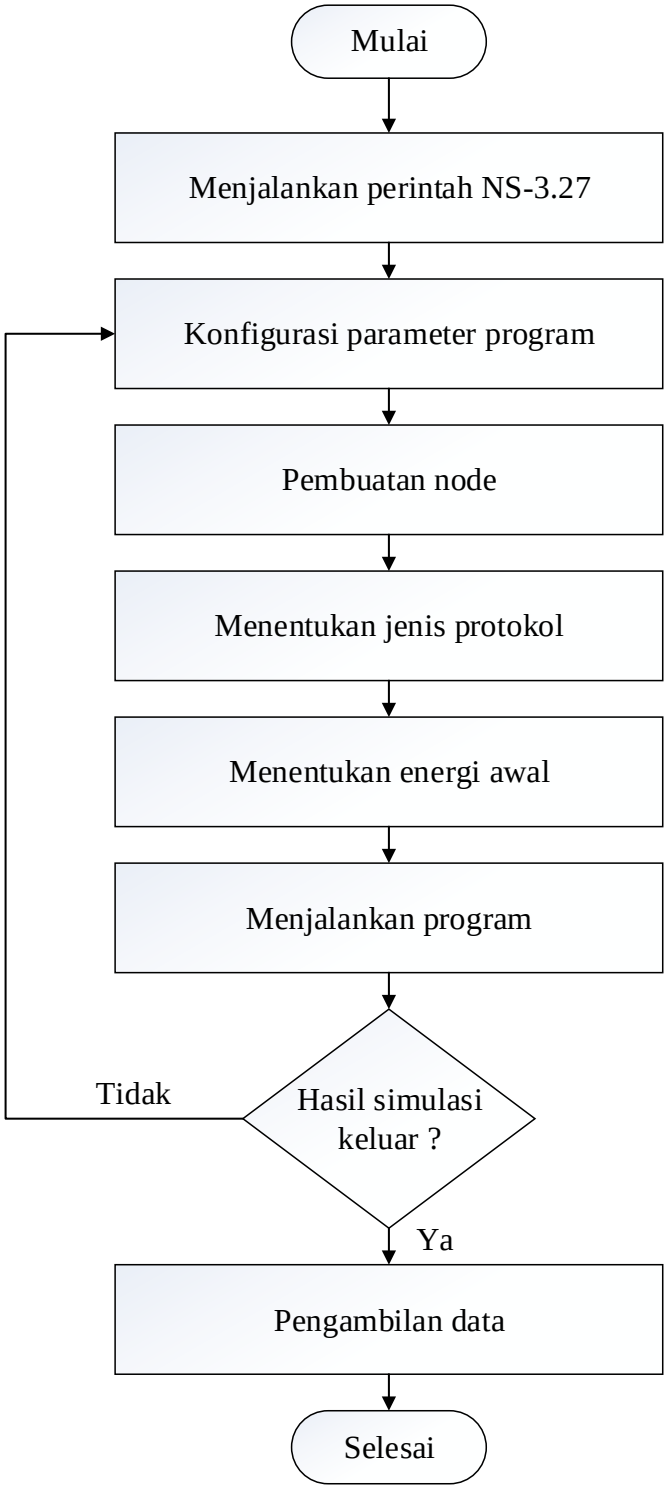
Pada tahap ini, peneliti menganalisis hasil simulasi yang keluar dengan membandingkan konsumsi energi dan energi yang tersisa dari protokol *routing* AODV dan DSDV pada jaringan MANET.

3.1.8 Penyusunan laporan

Tahap terakhir adalah penyusunan laporan dimana ketika tahap awal sampai dengan tahap terakhir telah dilakukan. Peneliti melakukan penyusunan laporan dari studi literatur yang didapatkan sampai dengan kesimpulan.

3.2 Perancangan Program

Gambar 3.2 menjelaskan alur perancangan program yang dilakukan peneliti. Penelitian ini menggunakan NS-3.27. Perancangan program yang terdiri dari menjalankan perintah NS-3.27, konfigurasi parameter jaringan, pembuatan node, menentukan jenis protokol, menentukan energi awal, lalu program dijalankan, ketika hasil simulasi keluar, kemudian lanjut ke tahap pengambilan data. Tabel 3.1 menunjukan parameter simulasi yang digunakan oleh peneliti dan Gambar 3.3 menunjukan jenis node yang digunakan oleh peneliti.



Gambar 3.2 *Flowchart* Perancangan Program

Tabel 3.3.1 Parameter Simulasi

Parameter Simulasi	**Nilai**
Jumlah n*ode*	50,100
Bandwidth	11Mbps
Simulasi area	100x100 m, 300x300 m
Jenis n*ode*	*Random way point*
Jenis protokol	*UDP*
Routing protocol	AODV, DSDV
Sistem komunikasi	*MAC/IEE 802.11b*
Maximum packet size	1024
Energi awal	1147 *Joules*
Parameter analisis	Konsumsi energi, energi yang tersisa

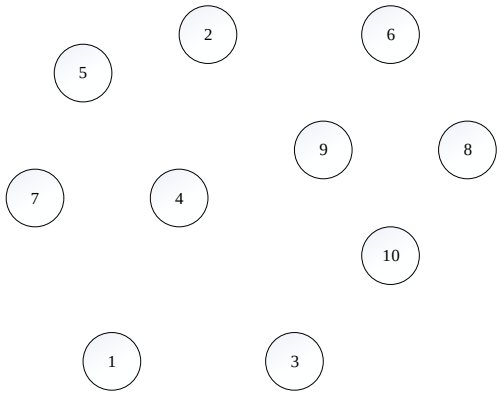
Energi awal yang digunakan untuk melakukan simulasi sebesar 1147 *Joules* Simulasi ini berjalan selama lima menit. Peneliti melakukan simulasi waktu lima menit karena dari kinerja laptop peneliti yang memiliki keterbatasan. Keterbatasan spesifikasi laptop yang digunakan oleh peneliti juga mempengaruhi simulasi yang dilakukan peneliti karena peneliti menggunakan software virtual machine untuk menjalankan dual OS dalam satu laptop sehingga RAM dan *memory* terbagi menjadi dua. Hal ini membuat kinerja laptop yang kurang dan membutuhkan waktu yang cukup lama untuk menjalankan simulasi.

3.2.1 Menentukan *node*

Pada tahap ini, *node* digunakan sebanyak 50, 100 *node* pada masing-masing *routing protocol*. Penambahan *node* pada setiap percobaan berfungsi untuk menganalisis konsumsi energi dan energi yang tersisa pada *rouing protocol* AODV dan DSDV pada jaringan MANET. Jenis node yang digunakan adalah *random waypoints.*

Random waypoints adalah salah satu jenis *node* yang paling umum digunakan dalam komunitas riset. *Node-node* akan bergerak secara acak untuk memilih tujuannya

sendiri dengan kecepatan secara acak [9]. Gambar 3.3 adalah model dari *random waypoints.*



Gambar 3.3 Model Random *Waypoints*

3.2.2 Menentukan protokol

Pada tahap menentukan protokol, peneliti menggunakan jenis protokol UDP *(User Datagram Protocol).* Pada NS3 terdapat protokol TCP dan UDP, tetapi peneliti hanya menggunakan protokol UDP.

3.2.3 Menentukan jenis jaringan

Pada proses jenis jaringan, peneliti menggunakan jaringan WLAN dengan standar 802.11b dengan *data rate* sebesar 11Mbps dan frekuensi 2.4 *GHz* hingga jangkauan 300 meter dengan lingkungan *outdoor*. Peneliti menggunakan jenis jaringan ini karena jenis ini paling umum digunakan di NS-3.27.

3.2.4 Pengambilan data

Setelah program dijalankan dan hasil simulasi didapatkan, peneliti melakukan pengambilan data. Data yang analisis oleh peneliti adalah konsumsi energi dan energi dalam bentuk grafik.

3.2.5 Konsumsi energi node

Konsumsi energi *node* adalah energi yang digunakan oleh *node* untuk saling berkomunikasi. Ketika simulasi berjalan, *node-node* pada jaringan MANET berkomunikasi dengan *node* lainnya untuk mengirim dan menerima paket data.

3.2.6 Energi yang tersisa

Energi yang tersisa adalah energi yang belum digunakan oleh *node-node* pada saat simulasi telah berakhir. Ketika simulasi berjalan, *node-node* akan bergerak untuk saling berkomunikasi mengirim data dan menerima data. Pada saat itulah peran konsumsi energi bekerja pada masing-masing *node* dan ketika simulasi berakhir, energi yang belum digunakan *node-node* adalah energi yang tersisa. Persamaan untuk menghitung energi yang tersisa adalah sebagai berikut:

$$Remaining\ energy\ =\ Initial\ energy - energy\ consumed \quad (3.1)$$

BAB 4
HASIL DAN PEMBAHASAN

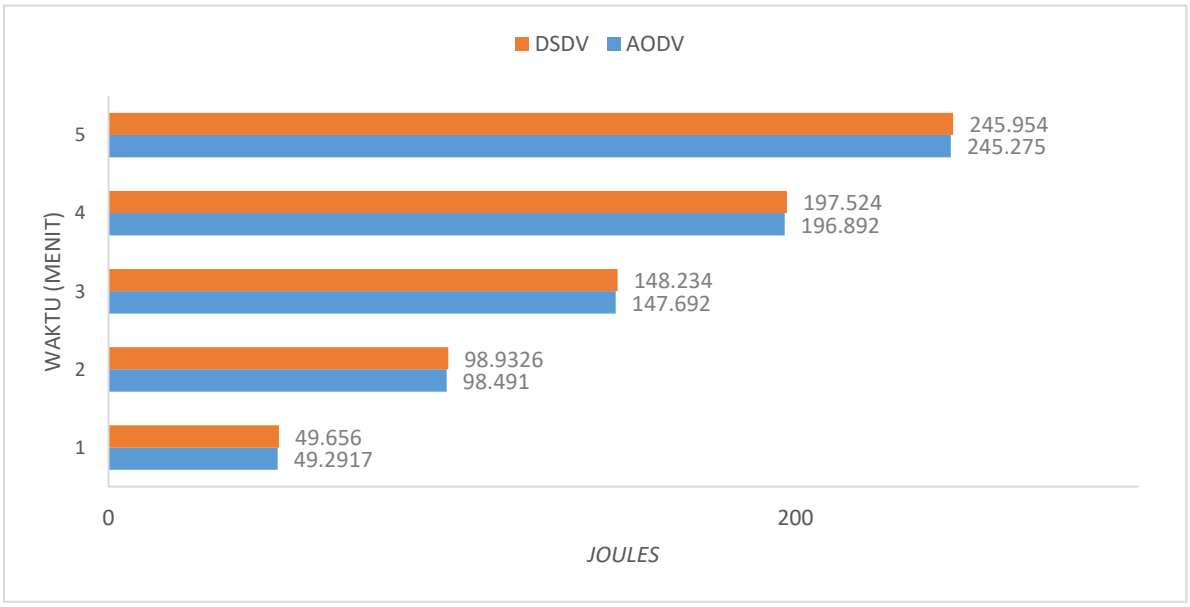
Pada bab hasil dan pembahasan ini disajikan hasil simulasi yang dilakukan oleh peneliti dengan skenario yang sudah ditentukan sebelumnya oleh peneliti. Setelah didapatkan hasil simulasi yang sesuai, selanjutnya adalah peneliti menganalisis data yang keluar berupa grafik.

Penelitian ini bertujuan untuk menganalisis dan membandingkan pengaruh konsumsi energi pada *routing protocol* AODV dan DSDV pada jaringan *MANET*. Parameter yang dianalisis oleh peneliti adalah konsumsi energi dan energi yang tersisa. Ada dua skenario yang digunakan oleh peneliti yaitu penambahan *node* dan penambahan luas area.

4.1 Skenario pertama

Skenario yang pertama adalah penambahan *node* dengan 50 dan 100 *node.* Gambar 4.1 menunjukan perbandingan keseluruhan konsumsi energi *node* dengan menggunakan 50 *node*. Selanjutnya pada Gambar 4.2 menunjukan perbandingan keseluruhan konsumsi energi *node* dengan menggunakan 100 *node.* Pada Tabel 4.1 menunjukan rasio perbandingan keseluruhan konsumsi energi *node* antara AODV dan DSDV.

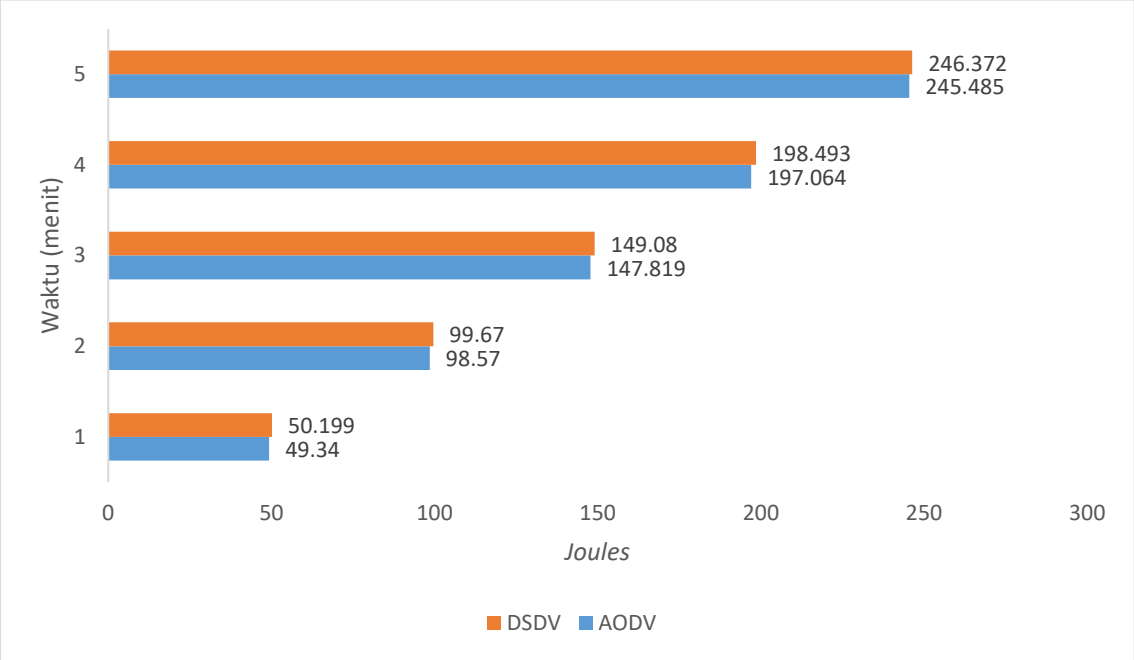
4.1.1 Konsumsi energi dengan 50 *node*



Gambar 4.1 Konsumsi energi dengan 50 *node*

Gambar 4.1 menunjukan grafik perbandingan keseluruhan konsumsi energi *node* pada *routing protocol* AODV dan DSDV dengan luas area 100x100 m dan menggunakan 50 *node*. Pada menit pertama, energi yang dikonsumsi oleh *routing* AODV adalah sebesar 49.2917 *Joules* dan energi yang dikonsumsi oleh *routing* DSDV sebesar 49,656 *Joules*. Energi yang dikonsumsi oleh *routing* AODV lebih rendah daripada DSDV dengan selisih energi sebesar 0,36 *Joules*. Setelah simulasi berjalan pada menit kedua, energi yang dikonsumsi oleh *routing* AODV adalah sebesar 98,491 *Joules* dan energi yang dikonsumsi oleh *routing* DSDV sebesar 98,9326 *Joules*. Energi yang dikonsumsi oleh *routing* AODV lebih rendah daripada DSDV dengan selisih energi sebesar 0,44 *Joules*. Setelah simulasi berjalan pada menit ketiga, energi yang dikonsumsi oleh *routing* AODV adalah sebesar 147,692 *Joules* dan energi yang dikonsumsi oleh *routing* DSDV sebesar 148,234 *Joules*. Energi yang dikonsumsi oleh *routing* AODV lebih rendah daripada DSDV dengan selisih energi sebesar 0,54 *Joules*. Setelah simulasi berjalan pada menit keempat, energi yang dikonsumsi oleh *routing* AODV adalah sebesar 196,892 *Joules* dan energi yang dikonsumsi oleh *routing* DSDV sebesar 197,524 *Joules*. Energi yang dikonsumsi oleh *routing* AODV lebih rendah daripada DSDV dengan selisih energi sebesar 0,63 *Joules*. ketika simulasi berhenti pada menit kelima, energi yang dikonsumsi oleh *routing* AODV adalah sebesar 245,275 *Joules* dan energi yang dikonsumsi oleh *routing* DSDV sebesar 245,954 *Joules*. Energi yang dikonsumsi oleh *routing* AODV lebih rendah daripada DSDV dengan selisih energi sebesar 0.68 *Joules*.

4.1.2 Konsumsi energi dengan 100 *node*



Gambar 4.2 Konsumsi energi dengan 100 *node*

Gambar 4.2 menunjukan grafik perbandingan keseluruhan konsumsi energi node pada *routing protocol* AODV dan DSDV dengan luas area 100x100m dan menggunakan 100 *node*. Setelah diamati dari simulasi berjalan pada menit pertama hingga menit kelima, didapatkan kesimpulan bahwa konsumsi energi pada *routing protocol* AODV lebih hemat daripada DSDV. Hal tersebut karena ketika simulasi berhenti pada menit kelima, total konsumsi energi yang digunakan oleh AODV sebesar 245,485 *Joules*. Sedangkan total konsumsi energi yang digunakan pada DSDV ketika simulasi berhenti sebesar 246,372 *Joules* dengan selisih sebesar 0,887 *Joules* lebih besar daripada AODV.

4.1.3 Rasio Konsumsi energi *node*

Tabel 4.1 Rasio konsumsi energi *node*

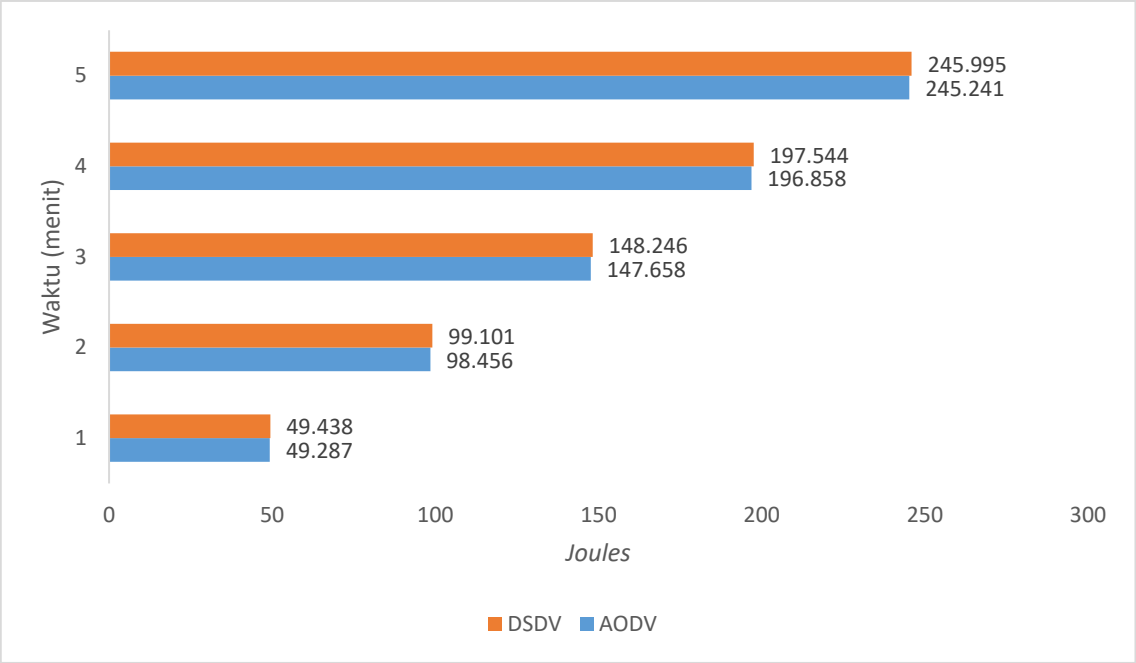
Routing protocol	***Node***	**Energi yang tersisa (Joules)**	**Rasio konsumsi energi (%)**
AODV	50	901,275	24,52
DSDV	50	901,046	24,59
AODV	100	901,515	24,54
DSDV	100	900,628	24,63

Tabel 4.1 menunjukan rasio konsumsi energi dengan skenario pertama penambahan 50 dan 100 *node* pada luas area 100x100 meter. Dapat dilihat bahwa perbandingan rasio konsumsi energi pada 50 *node* sebesar 0,07% dimana konsumsi energi *node* pada AODV lebih sedikit dari DSDV. Perbandingan rasio konsumsi energi pada 100 *node* sebesar 0,09% dimana pada AODV lebih sedikit daripada DSDV.

4.2 Skenario kedua

Skenario yang kedua adalah penambahan luas area 300x300 meter dan menggunakan 50, dan 100 *node*. Gambar 4.5 menunjukan perbandingan keseluruhan konsumsi energi *node* dengan menggunakan 50 *node*. Selanjutnya pada Gambar 4.6 menunjukan perbandingan keseluruhan konsumsi energi dengan menggunakan 100 *node*. Pada Tabel 4.2 menunjukan rasio perbandingan keseluruhan konsumsi energi *node* antara AODV dan DSDV.

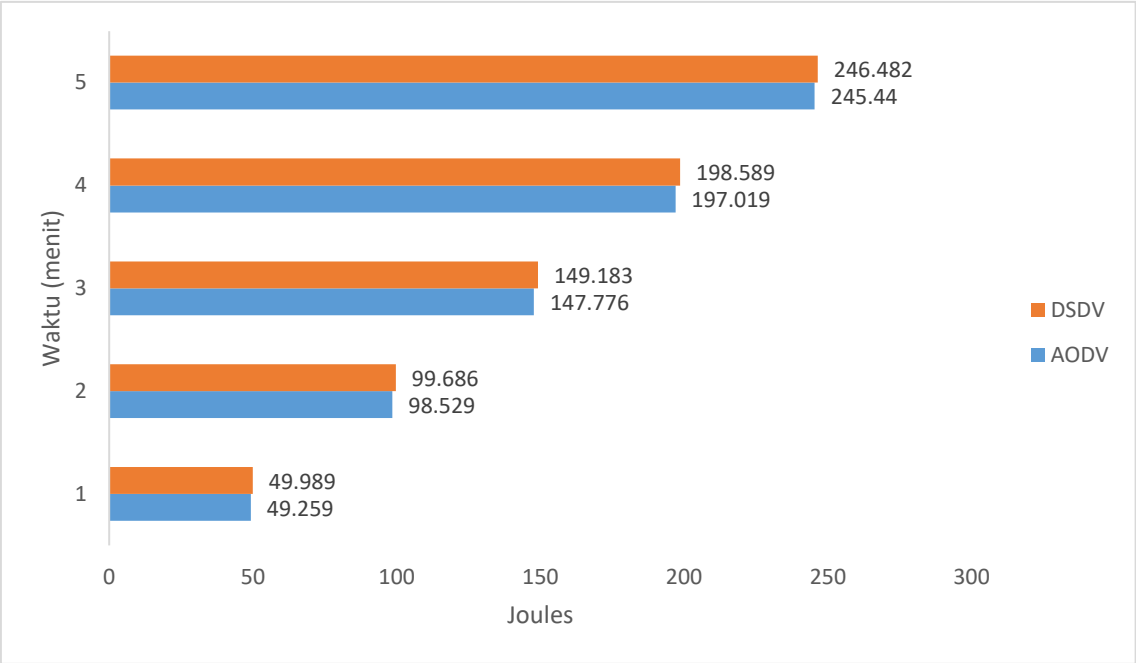
4.2.1 Konsumsi energi dengan 50 *node*



Gambar 4.3 Konsumsi energi dengan 50 *node*

Gambar 4.5 menunjukan grafik perbandingan keseluruhan konsumsi energi *node* pada *routing protocol* AODV dan DSDV dengan luas area 300x300m dan menggunakan 50 *node.* Setelah diamati dari simulasi berjalan pada menit pertama hingga menit kelima, didapatkan kesimpulan bahwa konsumsi energi pada *routing protocol* AODV lebih hemat daripada DSDV. Hal tersebut karena ketika simulasi berhenti pada menit kelima, total konsumsi energi yang digunakan oleh AODV sebesar 245,241 *Joules*. Sedangkan total konsumsi energi yang digunakan pada DSDV ketika simulasi berhenti sebesar 245,995 *Joules* dengan selisih sebesar 0,754 *Joules* lebih besar daripada AODV.

4.2.2 Konsumsi energi dengan 100 *node*



Gambar 4.4 Konsumsi energi dengan 100 *node*

Gambar 4.6 menunjukan grafik perbandingan keseluruhan konsumsi energi *node* pada *routing protocol* AODV dan DSDV dengan luas area 300x300 m dan menggunakan 100 *node*. Setelah diamati dari simulasi berjalan pada menit pertama hingga menit kelima, didapatkan kesimpulan bahwa konsumsi energi pada *routing protocol* AODV lebih hemat daripada DSDV. Hal tersebut karena ketika simulasi berhenti pada menit kelima, total konsumsi energi yang digunakan oleh AODV sebesar 245,44 *Joules*. Sedangkan total konsumsi energi yang digunakan pada DSDV ketika simulasi berhenti sebesar 246,48 *Joules* dengan selisih sebesar 1,04 *Joules* lebih besar daripada AODV.

4.2.3 Rasio konsumsi energi *node*

Tabel 4.2 Rasio konsumsi energi *node*

Routing protocol	***Node***	**Energi yang tersisa (Joules)**	**Rasio konsumsi energi (%)**
AODV	50	901,757	24,52
DSDV	50	901,005	24,59
AODV	100	901,56	24,54
DSDV	100	900,518	24,64

Tabel 4.2 menunjukan rasio konsumsi energi pada skenario kedua yaitu penambahan luas area. Dapat dilihat bahwa pada saat percobaan pertama dengan menggunakan 50 *node*, rasio perbandingan konsumsi energi antara AODV dan DSDV sebesar 0,07%. Pada percobaan kedua dengan menggunakan 100 *node*, rasio perbandingan konsumsi energi antara AODV dan DSDV sebesar 0,1%.

Konsumsi energi terjadi ketika *node-node* aktif yang bekerja mengirimkan paket data dan menerima paket data. Ketika *node-node* aktif yang digunakan semakin banyak, maka konsumsi energi yang digunakan akan semakin banyak. Luas area juga akan berpengaruh kepada konsumsi energi ketika *node-node* bekerja.

Setelah dilakukan perbandingan hasil simulasi dan analisa data yang telah dilakukan oleh peneliti, maka didapat kesimpulan bahwa konsumsi energi yang digunakan oleh *routing protocol* AODV pada jaringan MANET lebih sedikit daripada DSDV. AODV merupakan *routing protocol* reaktif dimana paket *routing* yang dihasilkan oleh AODV lebih sedikit daripada DSDV sehingga konsumsi energi yang digunakan AODV lebih sedikit daripada DSDV. DSDV terlihat lebih boros mengkonsumsi energi karena DSDV merupakan *routing* proaktif yang selalu melakukan *update routing table* secara periodik untuk menjaga suatu topologi jaringan dan rute *node*.

BAB 5
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Setelah melakukan penelitian ini, maka dapat diambil kesimpulan bahwa :

1. Dalam konsumsi energi, *routing protocol* AODV lebih hemat daripada DSDV pada jaringan MANET. DSDV terlihat lebih boros karena sifat DSDV yang merupakan *routing* proaktif dimana *routing* proaktif akan selalu memperbaharui *routing table* secara periodik untuk menjaga suatu topologi jaringan dan rute *node*.
2. Pada skenario penambahan 100 *node*, konsumsi energi *node* pada AODV lebih hemat daripada DSDV dengan selisih sebesar 0,887 *Joules* dan pada skenario penambahan luas area dengan menggunakan 100 *node*, rasio perbandingan konsumsi energi *node* antara AODV dan DSDV sebesar 0,1%.

5.2 Saran

Adapun saran-saran yang dapat digunakan pada penelitian kedepannya adalah penelitian ini hanya sebatas membandingkan konsumsi energi yang digunakan oleh *routing protocol* AODV dan DSDV pada jaringan MANET. Untuk selanjutnya bisa dibandingkan kembali menggunakan *routing protocol hybrid* dengan jaringan yang lain.

DAFTAR PUSTAKA

[1] H. Oudani, S. Krit, L. El Maimouni, and J. Laassiri, "Energy Consumption in Wireless Sensor Network : Simulation and Compartative Study of Flat and Hierarchical Routing," *IADIS Int. J. Comput. Sci. Inf. Syst.*, vol. 12, no. 1, pp. 109–125, 2017.

[2] P. S. Karadge and S. V Sankpal, "A Performance Comparison of Energy Efficient AODV Protocols in Mobile Ad hoc Networks," *Int. J. Adv. Res. Comput. Commun. Eng.*, vol. 2, no. 1, pp. 1000–1004, 2013.

[3] Aloysius Tri Sulistyo Putranto, "ANALISIS PENGGUNAAN ENERGY AODV DAN DSDV PADA MOBILE AD HOC NETWORK," pp. 1–85, 2016.

[4] J. Satiti, I. D. Irawati, and L. V. Yovita, "Analisis Perbandingan Performansi Protokol Routing Aodv Dan Dsdv Pada Wireless Sensor Network," *Prodi S1 Tek. Telekomun. Fak. Tek. Elektro, Univ. Telkom*, vol. 2, no. 2, pp. 1–6, 2015.

[5] R. M. N. Muhammad Irfan Denatama, Doan Perdana, "Analisis Perbandingan Kinerja Protokol Routing DSDV dan OLSR Untuk Perubahan Kecepatan Mobilitas pada Standar IEEE 802.11ah," vol. 8, no. 2, pp. 100–106, 2016.

[6] B. Swami and R. Singh, "Simulation Based Comparison Between OWL and DSDV," *Procedia Technol.*, vol. 24, pp. 1575–1580, 2016.

[7] D. U. Purba, R. Primananda, and K. Amron, "Analisis Kinerja Protokol Ad Hoc On-Demand Distance Vector (AODV) dan Fisheye State Routing (FSR) pada Mobile Ad Hoc Network," *Pengemb. Teknol. Inf. dn Ilmu Komput.*, vol. 2, no. 7, pp. 2626–2634, 2018.

[8] D. Irawan, "Simulasi Model Jaringan Mobile Ad-Hoc (Manet) Dengan Ns-3," *Badan Pengkaj. dan Penerapan Teknol. Jakarta. J. Konf. Nas. Sist. dan Inform. 2011; Bali, Novemb. 12, 2011.*, pp. 335–339, 2011.

[9] N. Mehta and M. Shah, "Performance Evaluation of Efficient Routing Protocols in Delay Tolerant Network under Different Human Mobility Models," *Int. J. Appl. or Innov. Eng. Manag.*, vol. 8, no. 1, pp. 169–178, 2015.

LAMPIRAN

Pada penelitian ini menggunakan perangkat lunak NS-3.27. berikut adalah salah satu kodingan *routing protocol* AODV pada jaringan MANET yang dikerjakan pada tugas akhir :

```
/*
 * EnergyExample.cc
 *
 *  Created on: Apr 16, 2018
 *      Author: nazarfaruq
 */
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/mobility-module.h"
#include "ns3/config-store-module.h"
#include "ns3/wifi-module.h"
#include "ns3/energy-module.h"
#include "ns3/internet-module.h"
#include "ns3/netanim-module.h"
#include "ns3/aodv-helper.h"
#include "ns3/netanim-module.h"
#include "ns3/flow-monitor-module.h"


#include <iostream>
#include <fstream>
#include <vector>
#include <string>

using namespace ns3;

NS_LOG_COMPONENT_DEFINE ("EnergyExample");

static inline std::string
PrintReceivedPacket (Address& from)
{
  InetSocketAddress iaddr = InetSocketAddress::ConvertFrom (from);

  std::basic_ostringstream<char> oss;
  oss << "--\nReceived packet! Socket: " << iaddr.GetIpv4 ()
      << " port: " << iaddr.GetPort ()
```

```
  std::basic_ostringstream<char> oss;
  oss << "--\nReceived packet! Socket: " << iaddr.GetIpv4 ()
      << " port: " << iaddr.GetPort ()
      << " at time = " << Simulator::Now ().GetSeconds ()
      << "\n--";

  return oss.str ();
}

/**
 * \param socket Pointer to socket.
 *
 * Packet receiving sink.
 */
void
ReceivePacket (Ptr<Socket> socket)
{
  Ptr<Packet> packet;
  Address from;
  while ((packet = socket->RecvFrom (from)))
    {
      if (packet->GetSize () > 0)
        {
          NS_LOG_UNCOND (PrintReceivedPacket (from));
        }
    }
}

/**
 * \param socket Pointer to socket.
 * \param pktSize Packet size.
 * \param n Pointer to node.
 * \param pktCount Number of packets to generate.
 * \param pktInterval Packet sending interval.
 *
 * Traffic generator.
 */
static void
GenerateTraffic (Ptr<Socket> socket, uint32_t pktSize, Ptr<Node> n,
                 uint32_t pktCount, Time pktInterval)
{
  if (pktCount > 0)
    {
      socket->Send (Create<Packet> (pktSize));
      Simulator::Schedule (pktInterval, &GenerateTraffic, socket, pktSize, n,
                           pktCount - 1, pktInterval);
    }
  else
    {
      socket->Close ();
    }
}

/// Trace function for remaining energy at node.
void
RemainingEnergy (double oldValue, double remainingEnergy)
{
  NS_LOG_UNCOND (Simulator::Now ().GetSeconds ()
                 << "s Current remaining energy = " << remainingEnergy << "J");
}

/// Trace function for total energy consumption at node.
void
TotalEnergy (double oldValue, double totalEnergy)
{
  NS_LOG_UNCOND (Simulator::Now ().GetSeconds ()
                 << "s Total energy consumed by radio = " << totalEnergy << "J");
}

int
main (int argc, char *argv[])
{
```

```
/*
LogComponentEnable ("EnergySource", LOG_LEVEL_DEBUG);
LogComponentEnable ("BasicEnergySource", LOG_LEVEL_DEBUG);
LogComponentEnable ("DeviceEnergyModel", LOG_LEVEL_DEBUG);
LogComponentEnable ("WifiRadioEnergyModel", LOG_LEVEL_DEBUG);
 */

std::string phyMode ("DsssRate11Mbps");
double Prss = -80;            // dBm
uint32_t PpacketSize = 1024;   // bytes
bool verbose = false;

// simulation parameters
uint32_t numPackets = 10000;  // number of packets to send
double interval = 1;          // seconds
double startTime = 300.0;       // seconds
double distanceToRx = 100.0;  // meters
/*
 * This is a magic number used to set the transmit power, based on other
 * configuration.
 */
double offset = 81;

CommandLine cmd;
cmd.AddValue ("phyMode", "Wifi Phy mode", phyMode);
cmd.AddValue ("Prss", "Intended primary RSS (dBm)", Prss);
cmd.AddValue ("PpacketSize", "size of application packet sent", PpacketSize);
cmd.AddValue ("numPackets", "Total number of packets to send", numPackets);
cmd.AddValue ("startTime", "Simulation start time", startTime);
cmd.AddValue ("distanceToRx", "X-Axis distance between nodes", distanceToRx);
cmd.AddValue ("verbose", "Turn on all device log components", verbose);
cmd.Parse (argc, argv);

// Convert to time object
Time interPacketInterval = Seconds (interval);

// disable fragmentation for frames below 2200 bytes
Config::SetDefault ("ns3::WifiRemoteStationManager::FragmentationThreshold",
                    StringValue ("2200"));
// turn off RTS/CTS for frames below 2200 bytes
Config::SetDefault ("ns3::WifiRemoteStationManager::RtsCtsThreshold",
                    StringValue ("2200"));
// Fix non-unicast data rate to be the same as that of unicast
Config::SetDefault ("ns3::WifiRemoteStationManager::NonUnicastMode",
                    StringValue (phyMode));

NodeContainer c;
c.Create (50);     // create 2 nodes
NodeContainer networkNodes;
networkNodes.Add (c.Get (0));
networkNodes.Add (c.Get (1));
networkNodes.Add (c.Get (2));
networkNodes.Add (c.Get (3));
networkNodes.Add (c.Get (4));
networkNodes.Add (c.Get (5));
networkNodes.Add (c.Get (6));
networkNodes.Add (c.Get (7));
networkNodes.Add (c.Get (8));
networkNodes.Add (c.Get (9));
networkNodes.Add (c.Get (10));
networkNodes.Add (c.Get (11));
networkNodes.Add (c.Get (12));
networkNodes.Add (c.Get (13));
networkNodes.Add (c.Get (14));
networkNodes.Add (c.Get (15));
networkNodes.Add (c.Get (16));
networkNodes.Add (c.Get (17));
networkNodes.Add (c.Get (18));
networkNodes.Add (c.Get (19));
networkNodes.Add (c.Get (20));
networkNodes.Add (c.Get (21));
networkNodes.Add (c.Get (22));
```

```
networkNodes.Add (c.Get (23));
networkNodes.Add (c.Get (24));
networkNodes.Add (c.Get (25));
networkNodes.Add (c.Get (26));
networkNodes.Add (c.Get (27));
networkNodes.Add (c.Get (28));
networkNodes.Add (c.Get (29));
networkNodes.Add (c.Get (30));
networkNodes.Add (c.Get (31));
networkNodes.Add (c.Get (32));
networkNodes.Add (c.Get (33));
networkNodes.Add (c.Get (34));
networkNodes.Add (c.Get (35));
networkNodes.Add (c.Get (36));
networkNodes.Add (c.Get (37));
networkNodes.Add (c.Get (38));
networkNodes.Add (c.Get (39));
networkNodes.Add (c.Get (40));
networkNodes.Add (c.Get (41));
networkNodes.Add (c.Get (42));
networkNodes.Add (c.Get (43));
networkNodes.Add (c.Get (44));
networkNodes.Add (c.Get (45));
networkNodes.Add (c.Get (46));
networkNodes.Add (c.Get (47));
networkNodes.Add (c.Get (48));
networkNodes.Add (c.Get (49));

// The below set of helpers will help us to put together the wifi NICs we want
WifiHelper wifi;
if (verbose)
  {
    wifi.EnableLogComponents ();
  }
wifi.SetStandard (WIFI_PHY_STANDARD_80211b);

/** Wifi PHY **/
/***************************************************************************/
YansWifiPhyHelper wifiPhy = YansWifiPhyHelper::Default ();
wifiPhy.Set ("RxGain", DoubleValue (-10));
wifiPhy.Set ("TxGain", DoubleValue (offset + Prss));
wifiPhy.Set ("CcaMode1Threshold", DoubleValue (0.0));
/***************************************************************************/

/** wifi channel **/
YansWifiChannelHelper wifiChannel;
wifiChannel.SetPropagationDelay ("ns3::ConstantSpeedPropagationDelayModel");
wifiChannel.AddPropagationLoss ("ns3::FriisPropagationLossModel");
// create wifi channel
Ptr<YansWifiChannel> wifiChannelPtr = wifiChannel.Create ();
wifiPhy.SetChannel (wifiChannelPtr);

/** MAC layer **/
// Add a MAC and disable rate control
WifiMacHelper wifiMac;
wifi.SetRemoteStationManager ("ns3::ConstantRateWifiManager", "DataMode",
                              StringValue (phyMode), "ControlMode",
                              StringValue (phyMode));
// Set it to ad-hoc mode
wifiMac.SetType ("ns3::AdhocWifiMac");

/** install PHY + MAC **/
NetDeviceContainer devices = wifi.Install (wifiPhy, wifiMac, networkNodes);

Ptr<ListPositionAllocator> waypoint1 =CreateObject<ListPositionAllocator> ();
waypoint1 = CreateObject<ListPositionAllocator> ();
waypoint1->Add (Vector (100.0, 0.0, 0.0));
waypoint1->Add (Vector (200.0, 0.0, 0.0));
waypoint1->Add (Vector (100.0, 50.0, 0.0));
waypoint1->Add (Vector (200.0, 50.0, 0.0));
```

```
/** mobility **/
MobilityHelper mobility;

              mobility.SetPositionAllocator ("ns3::GridPositionAllocator",
                                             "MinX", DoubleValue (75.0),
                                             "MinY", DoubleValue (0.0),
                                             "DeltaX", DoubleValue (75.0),
                                             "DeltaY", DoubleValue (0.0),
                                             "GridWidth", UintegerValue (4));

              mobility.SetMobilityModel ("ns3::RandomWaypointMobilityModel","Speed", StringValue
              ("ns3::UniformRandomVariable[Min=5.0|Max=10.0]"),"Pause", StringValue
              ("ns3::ConstantRandomVariable[Constant=0.0]"),"PositionAllocator", PointerValue(waypoint1));


Ptr<ListPositionAllocator> positionAlloc = CreateObject<ListPositionAllocator> ();
positionAlloc->Add (Vector (20.0, 20.0, 0.0));
positionAlloc->Add (Vector (4 * distanceToRx, 30.0, 15.0));
mobility.SetPositionAllocator (positionAlloc);
mobility.Install (networkNodes);



/** Energy Model **/
/*****************************************************************************/
/* energy source */
BasicEnergySourceHelper basicSourceHelper;
// configure energy source
basicSourceHelper.Set ("BasicEnergySourceInitialEnergyJ", DoubleValue (300));
// install source
EnergySourceContainer sources = basicSourceHelper.Install (c);
/* device energy model */
WifiRadioEnergyModelHelper radioEnergyHelper;
// configure radio energy model
radioEnergyHelper.Set ("TxCurrentA", DoubleValue (0.0174));
// install device model
DeviceEnergyModelContainer deviceModels = radioEnergyHelper.Install (devices, sources);
// install battery lifetime

/*****************************************************************************/


AodvHelper aodv;
Ipv4StaticRoutingHelper ipv4RoutingHelper;

Ipv4ListRoutingHelper list1;
list1.Add (ipv4RoutingHelper, 10);

Ipv4ListRoutingHelper list2;
list2.Add (aodv, 20);

/** Internet stack **/
  InternetStackHelper internet;
  internet.SetRoutingHelper (aodv);
  internet.Install (networkNodes);


Ipv4AddressHelper ipv4;
NS_LOG_INFO ("Assign IP Addresses.");
ipv4.SetBase ("10.1.1.0", "255.255.255.0");
Ipv4InterfaceContainer i = ipv4.Assign (devices);

TypeId tid = TypeId::LookupByName ("ns3::UdpSocketFactory");
Ptr<Socket> recvSink = Socket::CreateSocket (networkNodes.Get (49), tid);  // node 1, receiver
InetSocketAddress local = InetSocketAddress (Ipv4Address::GetAny (), 80);
recvSink->Bind (local);
recvSink->SetRecvCallback (MakeCallback (&ReceivePacket));



Ptr<Socket> source = Socket::CreateSocket (networkNodes.Get (0), tid);    // node 0, sender
```

```
InetSocketAddress remote = InetSocketAddress (Ipv4Address::GetBroadcast (), 80);
source->SetAllowBroadcast (true);
source->Connect (remote);

/** connect trace sources **/
/***************************************************************************/
// all sources are connected to node 1
// energy source
Ptr<BasicEnergySource> basicSourcePtr = DynamicCast<BasicEnergySource> (sources.Get (49));
basicSourcePtr->TraceConnectWithoutContext ("RemainingEnergy", MakeCallback (&RemainingEnergy));
// device energy model
Ptr<DeviceEnergyModel> basicRadioModelPtr =
  basicSourcePtr->FindDeviceEnergyModels ("ns3::WifiRadioEnergyModel").Get (0);
NS_ASSERT (basicRadioModelPtr != NULL);
basicRadioModelPtr->TraceConnectWithoutContext ("TotalEnergyConsumption", MakeCallback (&TotalEnergy));
/***************************************************************************/


/** simulation setup **/
// start traffic
Simulator::Schedule (Seconds (startTime), &GenerateTraffic, source, PpacketSize,
                     networkNodes.Get (0), numPackets, interPacketInterval);

//install FlowMonitor
  FlowMonitorHelper flowmon;
  Ptr<FlowMonitor> monitor;
  monitor = flowmon.InstallAll();

Simulator::Stop (Seconds (300.0));

AnimationInterface anim ("EnergyExample.xml");

Simulator::Run ();
Simulator::Destroy ();

return 0;
```