TUGAS AKHIR – EE 184801

IMPLEMENTASI *RANDOM WALK ROUTING ALGORITHM* PADA SIMULASI JARINGAN SENSOR NIRKABEL

Muhammad Danang Retzafakhri
NRP 07111440000119

Dosen Pembimbing
Sri Rahayu, S.T., M.Kom.

DEPARTEMEN TEKNIK ELEKTRO
Fakultas Teknologi Elektro
Institut Teknologi Sepuluh Nopember
Surabaya 2019

TUGAS AKHIR – EE 184801

IMPLEMENTASI *RANDOM WALK ROUTING ALGORITHM* PADA SIMULASI JARINGAN SENSOR NIRKABEL

Muhammad Danang Retzafakhri
NRP 07111440000119

Dosen Pembimbing
Sri Rahayu, S.T., M.Kom.

DEPARTEMEN TEKNIK ELEKTRO
Fakultas Teknologi Elektro
Institut Teknologi Sepuluh Nopember
Surabaya 2019

FINAL PROJECT – EE 184801

RANDOM WALK ROUTING ALGORITHM IMPLEMENTATION ON *WIRELESS* SENSOR NETWORK SIMULATION

Muhammad Danang Retzafakhri
NRP 07111440000119

Advisors
Sri Rahayu, S.T., M.Kom.

DEPARTMENT OF ELECTRICAL ENGINEERING
Faculty of Electrical Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2019

PERNYATAAN KEASLIAN
TUGAS AKHIR

Dengan ini saya menyatakan bahwa isi keseluruhan Tugas akhir saya dengan judul "**IMPLEMENTASI *RANDOM WALK ROUTING* ALGORITHM PADA SIMULASI JARINGAN SENSOR NIRKABEL**" adalah benar-benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diijinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka.

Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Surabaya, Juli 2019

Muhammad Danang Retzafakhri
NRP. 07111440000119

Halaman ini sengaja dikosongkan

IMPLEMENTASI *RANDOM WALK ROUTING ALGORITHM* PADA SIMULASI JARINGAN SENSOR NIRKABEL

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Teknik
Pada
Bidang Studi Teknik Telekomunikasi dan Multimedia
Departemen Teknik Elektro
Institut Teknologi Sepuluh Nopember

Menyetujui :

Dosen Pembimbing

Sri Rahayu, S.T., M.Kom.
NIP. 196802281997022001

KEMENTERIAN RISET TEKNOLOGI DAN PENDIDIKAN TINGGI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
FAKULTAS TEKNOLOGI ELEKTRO
DEPARTEMEN TEKNIK ELEKTRO

SURABAYA
JULI, 2019

Halaman ini sengaja dikosongkan

IMPLEMENTASI *RANDOM WALK ROUTING* ALGORITHM PADA SIMULASI JARINGAN SENSOR NIRKABEL

Nama mahasiswa : Muhammad Danang Retzafakhri
NRP : 07111440000119
Dosen Pembimbing : Sri Rahayu, S.T., M.Kom.

Abstrak:

Wireless Sensor Network (WSN) merupakan suatu jaringan nirkabel yang menghubungkan beberapa sensor (sensor *node*) yang terletak di lokasi berbeda untuk membentuk sebuah sistem (*plan*) untuk keperluan tertentu. Sensor-sensor ini saling berkomunikasi dan bekerja sama untuk menginformasikan data-data sesuai sepesifikasinya atau difungsikan sebagai sebuah *node* untuk keperluan *routing*. Berbagai protokol *routing* memiliki prinsip yang berbeda terkait jalur lintasan yang dipilih, ada jalur yang sudah ditentukan sebelumnya (*Single Path*), jalur terpendek (*Shortest Path*) ataupun tergantung kondisi yang ada (*Random Path*). Tentu masing-masing pilihan memiliki konsekuensi tersendiri terkait kinerja transmisi data dari pengirim (*Source Node*) menuju *node* penerima (*Sink Node*). Dalam Tugas Akhir ini, dilakukan analisa penerapan algoritma *Random walk* pada *Routing Protocol* untuk jaringan sensor nirkabel (WSN), mulai dari pembuatan simulasi jaringan WSN, pembangkitan trafik pada jaringan tersebut, pemprosesan data parameter dan kemudian dilakukan analisa kinerjanya. Untuk penyelesaian tugas akhir ini digunakan alat bantu perangkat lunak berupa *Network simulator 2* dan *Tracegraph*. Adapun parameter kinerja jaringan WSN yang dibahas meliputi *delay*, *throughput* dan *packet delivery ratio* dengan nilai masing-masing sebesar 0.60377 detik, 122.0614 kbps dan 90.21577% pada protokol *transport* TCP. Dan *delay*, *throughput* dan *packet delivery ratio* masing-masing sebesar 2.8492 detik, 49.4563 kbps dan 77.8976% pada protokol *transport* UDP. Dengan hasil tugas akhir ini diharapkan bisa diperoleh gambaran performansi *Random walk Routing Protocol* pada jaringan WSN untuk mengirimkan data dalam kondisi *traffic setting* tertentu.

Kata kunci: *Wireless Sensor Network* (WSN), *Random walk*, *Routing*, Performansi.

Halaman ini sengaja dikosongkan

RANDOM WALK ROUTING ALGORITHM IMPLEMENTATION ON *WIRELESS* SENSOR NETWORK SIMULATION

Student Name : Muhammad Danang Retzafakhri
NRP : 07111440000119
Advisor : Sri Rahayu, S.T., M.Kom.

Abstract:

Wireless Sensor Network (WSN) is a wireless network that connects several sensors (sensor nodes) located in different locations to form a system (plan) for certain purposes. These sensors communicate with each other and work together to inform the data according to their specifications or function as a node for routing needs. Various routing protocols have different principles related to the chosen path, there are paths that have been predetermined (Single Path), the shortest path (Shortest Path) or depending on the conditions (Random Path). Of course each choice has its own consequences related to the performance of data transmission from the sender (Source Node) to the receiving node (Sink Node). In this Final Project, an analysis of the Random walk algorithm on the Routing Protocol for wireless sensor networks (WSN) is analyzed, starting from the WSN network simulation, traffic generation on the network, processing of parameter data and then analyzing its performance. For the completion of this final project, software tools in the form of Network simulator 2 and Tracegraph are used. The WSN network performance parameters that will be discussed include packet delivery ratio, delay, and throughput with its each value is 0.60377 second, 122.0614 kbps and 90.21577% on TCP transport protocol. And ratio, delay, and throughput with its each value is 2.8492 second, 49.4563 kbps and 77.8976% on UDP transport protocol With the results of this final project, it is expected that an overview of the Random Walk Routing Protocol performance on the WSN network can be obtained to transmit data in certain traffic conditions.

Key Word: *Wireless Sensor Network* (WSN), *Random walk*, *Routing*, Performance.

Halaman ini sengaja dikosongkan

KATA PENGANTAR

Puji Syukur penulis panjatkan kehadirat Allah SWT atas segala Rahmat, Karunia, Pertolongan, Ilmu, dan Petunjuk yang telah dilimpahkan-Nya sehingga penulis mampu menyelesaikan tugas akhir dengan judul "**IMPLEMENTASI *RANDOM WALK ROUTING* ALGORITHM PADA SIMULASI JARINGAN SENSOR NIRKABEL**". Selaian disusun sebagai salah satu persyarat untuk menyelesaikan jenjang pendidikan S1 pada Bidang Telekomunikasi, Depatemen Teknik Elektro, Fakultas Teknologi Elektro, Institut Teknologi Sepuluh Nopember, tugas akhir ini didedikasikan untuk bangsa Indonesia dalam rangka menyongsong 100 tahun kemerdekaan Republik Indonesia. Besar harapan penulis agar tugas akhir ini dapat bermanfaat pada perkembangan teknologi kendaraan listrik yang ramah lingkungan dalam upaya mewujudkan kemandirian energi sebagai bentuk kemerdekaan yang sesungguhnya. Penulis menyadari bahwa masih terdapat banyak kekuranga didalamnya. Oleh karena itu, penulis berharap kritik dan saran untuk penelitian berikutnya yang lebih baik

Dalam pengerjaan tugas akhir ini, penulisan banyak mendapat dukungan dalam bentuk apapun dari berbagai pihak. Oleh sebab itu, seiring dengan selesainya buku tugas akhir ini,penulis ingin mengucapkan terimakasih kepada:

1. Ibu Trindawati dan Bapak Suswanto selaku kedua orang tua yang senantiasa ada untuk memberi semangat, motivasi, dukungan, dan banyak hal lainnya yang sangat berharga dalam hidup saya.
2. Seluruh kelurga besar mulai dari adik Fijar serta sepupu yang juga memberikan dukungan dan doa.
3. Ibu Sri Rahayu selaku dosen pembimbing yang telah memberikan arahan, ilmu, bimbingan, serta perhatian selama proses penyelesaian tugas akhir ini.
4. Teman-teman seperjuangan e54 dan e55 yang telah menemani dan memberikan dukungan selama masa kuliah sampai penyusunan tugas akhir ini.
5. Dosen dan Teman-teman dari bidang studi Telekomunikasi Multimedia mulai B301 hingga B306 yang telah memberikan banyak wawasan dan sharing ilmu hingga saya dapat memahami mata kuliah sampai kelulusan.
6. Keluarga besar penghuni SDR1 23 yang selalu memberikan motivasi agar segera dapat menyelesaikan perkuliahan.

7. Teman-teman seperjuangan beserta seluruh pihak yang tidak dapat disebutkan satu persatu.

. Penulis mengucapkan terimakasih atas doa dan dukungan yang diberikan selama ini. Penulis juga meminta maaf apabila terdapat kesalahan maupun kekurangan pada tugas akhir ini. Semoga tugas akhir ini dapat bermanfaat bagi mahasiswa teknik elektro ITS pada khususnya dan para pembaca pada umumnya.

Surabaya, Juli 2019

Muhammad Danang Retzafakhri

DAFTAR ISI

Halaman ini sengaja dikosongkan

DAFTAR GAMBAR

DAFTAR TABEL

Halaman ini sengaja dikosongkan

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Wireless Sensor Network (WSN) merupakan suatu jaringan nirkabel yang menghubungkan beberapa sensor (sensor *node*) yang terletak di lokasi berbeda untuk membentuk sebuah sistem (jaringan) untuk keperluan tertentu (*plan*). Sensor-sensor ini saling berkomunikasi dan bekerja sama untuk menginformasikan data-data sesuai sepesifikasinya atau difungsikan sebagai *node-node* untuk keperluan *routing*.

Kemajuan teknologi WSN yang pesat tidak lepas dari fakta bahwa banyak prosesor bukan hanya berada di dalam sebuah PC/Laptop, namun juga tersebar di beberapa lokasi yang berbeda, sehingga perlu dibuatkan sistem jaringan agar sensor-sensor tersebut dapat di kelola dengan baik. WSN banyak diterapkan untuk aplikasi militer, kesehatan, *remote control*, *chip robotic*, alat komunikasi, dan mesin – mesin industri yang telah terintegrasi dengan sensor. Dengan adanya teknologi WSN, kita dapat dengan mudah memonitor dan mengontrol temperature, kelembapan, kondisi cahaya, level derau, pergerakan suatu objek dan sebagainya, dari jarak jauh.

Hal yang penting dalam pengelolaan komunikasi sensor-sensor pada WSN adalah bagaimana menjamin ketersediaan jalur komunikasi antar sensor dalam sebuah konsep *routing*. *Routing* sendiri merupakan proses untuk memilih jalur lintasan (path) yang harus dilalui selama transmisi paket. Dalam konsep *static routing* informasi *routing* sudah dikonfigurasi dalam tabel *routing* telah ditetapkan oleh administrator, sehingga memiliki sifat yang tetap. Hanya saja jenis *routing* ini memiliki kelemahan yaitu, jika salah satu jalur mati/*down* atau terjadi perubahan jaringan (tidak sesuai dengan tabel *routing*), maka lalu lintas paket menjadi tidak dapat dialihkan ke jalur lain dan harus mengkonfigurasi kembali tabel *routing* yang ada. Untuk mengatasi permasalahan tersebut diperlukan konsep *routing* yang jalurnya tidak selalu sama (berubah-ubah). *Routing* dengan jalur berubah-ubah tersebut merupakan salah satu dari jenis *routing* dinamis. Salah satu dari *routing* dinamis yang dapat digunakan untuk konsep pemilihan jalur-jalur atau rute-rute transmisi data adalah *routing* dengan algoritma *random walk*.

1.2 Permasalahan dan Batasannya

Perumusan masalah pada tugas akhir tugas akhir ini adalah:

1. Membuat simulasi jaringan *node* WSN menggunakan *software tools.*
2. Perancangan jaringan WSN dibuat dengan menggunakan NS2.
3. Jumlah *node* yang digunakan dalam simulasi WSN adalah 30.
4. Ukuran Paket yang digunakan 1500 bytes.
5. Algoritma *network routing* yang digunakan adalah *Random Walk*.
6. Protokol Transport yang digunakan TCP.
7. Protokol Aplikasi yang digunakan FTP.
8. Parameter kinerja yang diamati meliputi *delay, throughput* dan *packet delivery ratio.*
9. Protokol *network routing* pembanding yang digunakan AODV dan *Waypoint.*
10. Protokol Transport pembanding yang digunakan UDP.

1.3 Tujuan

Tugas akhir tugas akhir ini bertujuan untuk:

1. Mengukur kinerja penerapan algoritma *Random walk routing* pada jaringan WSN.
2. Membandingkan kinerja *Random walk routing* dengan algoritma *routing* lainnya pada jaringan WSN.

1.4 Metodologi

Metodologi yang digunakan dalam pengerjaan tugas akhir ini adalah sebagai berikut:

1. Studi Literatur

   Studi literatur dilakukan dengan mencari dan mempelajari beberapa buku, paper, dan jurnal baik skala nasional maupun internasional yang dapat menunjang tugas akhir ini. Pada tahap ini akan mempelajari beberapa materi utama yaitu dasar – dasar pada WSN dan *Routing* pada WSN terutama teknik *Random walk Routing*. Dan juga mempelajari beberapa *software-tool* yang akan digunakan pada tugas akhir ini yaitu NS2 dan Tracegrah 2.02.

2. Desain Sistem WSN

Desain sistem dilakukan untuk menentukan suatu luasan area yang digunakan dalam jaringan WSN. Bentuk topologi jaringan juga ditentukan untuk penempatan *node-node* yang ada pada jaringan WSN. Pada tahap ini dilakukan juga penentuan jumlah *node* yang akan digunakan pada area jaringan WSN dalam bentuk topologi yang sudah ditentukan sebelumnya.

3. Simulasi *Routing* WSN
   Simulasi *routing* WSN dilakukan dengan software NS2, hal pertama yang dilakukan adalah melakukan pengaturan pada software untuk menentukan topologi, parameter input, dan pembangkitan trafik sehingga WSN dapat berjalan. Dilakukan juga penerapan *Random walk* pada simulasi WSN tersebut. Hasil simulasi pada NS2 digunakan pada Tracegraph 2.02 untuk mendapatkan parameter-parameter output.

4. Analisa Data dan Hasil Simulasi
   Dalam tahap ini dilakukan pengambilan parameter output dari hasil simulasi NS2 yang kemudian dianalisa dengan Tracegraph 2.02. Hasil simulasi kemudian dianalisa pada tahap Pengolahan Hasil Simulasi dengan diperhitungkan juga secara teoritis. Kemudian tahap berikutnya melakukan komparasi terhadap hasil simulasi dengan pengaturan parameter yang lain.

5. Penyelesaian Laporan
   Pada tahap ini semua rangkaian kegiatan tugas akhir dibukukan dalam bentuk laporan. Tahap ini sekaligus sebagai syarat kelulusan dari mata kuliah tugas akhir.

1.5 Sistematika Penulisan

Sistematika penulisan dalam tugas akhir ini terdiri atas lima bab dengan uraian sebagai berikut :

Bab 1 : Pendahuluan
Pada bab ini dibahas penjelasan mengenai latar belakang, permasalahan dan batasan masalah, tujuan, metode tugas akhir, sistematika pembahasan, dan relevansi terhadap tugas akhir.

Bab 2 : Tinjauan Pustaka

Pada bab ini dibahas mengenai penelitian sebelumnya dan dasar teori yang digunakan untuk menunjang penyusunan tugas akhir ini. Teori yang digunakan diantaranya penjelasan WSN, *Routing,* Algoritma *Random walk* dan *software tools* yang digunakan.

Bab 3 : Perancangan Sistem

Pada bab ini dibahas mengenai desain, perancangan, dan pemodelan suatu jaringan sensor nirkabel yang akan disimulasikan.

Bab 4 : Hasil Pengujian dan Analisa

Pada bab ini dibahas mengenai hasil simulasi dan analisis dari performa algoritma *routing Random walk* dengan beberapa pengaturan parameter yang berbeda.

Bab 5 : Penutup

Bab ini berisi tentang kesimpulan dari hasil analisis yang dilakukan dan saran untuk pengembangan penelitian berikutnya.

1.6 Relevansi

Penelitian diharapkan dapat memberikan manfaat, yaitu:

1. Menjadi referensi tentang gambaran performansi algoritma *Random Walk Routing* pada jaringan WSN dan perbedaannya terhadap kinerja penerapan algoritma *routing* lainnya.
2. Menjadi referensi bagi mahasiswa yang akan mengerjakan penelitian dengan topik *routing* pada jaringan WSN.

BAB 2
TINJAUAN PUSTAKA

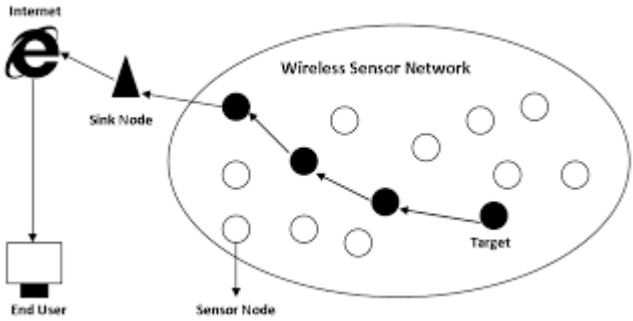
2.1 Jaringan Sensor Nirkabel

Sensor adalah suatu *device* yang berfungsi untuk mengkonversi besaran fisis ke besaran fisis lain seperti listrik. Kumpulan dari beberapa *wireless* sensor jika masing-masing diletakkan secara special dan diatur konfigurasinya, dapat disebut dengan WSN (*Wireless Sensor Network*).

WSN (*Wireless sensor network*) merupakan jaringan *wireless* alat yang menggunakan sensor untuk memonitor fisik atau kondisi lingkungan sekitar, seperti suhu, suara, getaran, gelombang elektromagnetik, tekanan, gerakan, dan lain-lain. Perkembangan dari WSN sebenarnya sudah dimulai dari kebutuhan dalam bidang militer seperti pemantauan pada saat perang di medan perang. Tapi sekarang WSN sudah digunakan dalam bidang industri dan penggunaan untuk kemudahan masyarakat sipil, melingkupi pengawasan dan pengontrolan proses dalam industri, mesin pengawasan kesehatan, pemantau kondisi lingkungan, aplikasi untuk kesehatan, otomatisasi pada rumah, dan pengaturan pada lalu lintas.

Dalam penambahan pada satu atau lebih suatu sensor, masing-masing *node* dalam WSN biasanya dilengkapi dengan radio tranciever atau alat komunikasi *wireless* lainnya, mikro-kontroler kecil, dan sumber energi, biasanya baterai. Untuk ukuran *node* sensor pada WSN memiliki kisaran *node* sensor yang bisa mencapai besar dari sebuah kotak sepatu hingga seukuran debu. Aplikasi dan penggunaan dari WSN ada banyak dan bervariasi, tapi umumnya adalah untuk *monitoring, tracking* dan *controlling*. Penerapan spesifik dari WSN misalnya adalah pengontrolan reaktor nuklir, pendeteksi api, dan *monitoring* lalu lintas.

Kemampuan sensor pada WSN secara luas membuat penggunaan WSN untuk melakukan *monitoring* banyak digunakan. WSN dapat digunakan dengan sensor sederhana yang me-*monitoring* suatu fenomena sedangkan untuk yang kompleks maka setiap WSN akan mempunyai lebih dari satu sensor sehingga WSN ini akan dapat melakukan banyak *monitoring* suatu fenomena. Jika WSN ini dihubungkan ke *gateway* yang dapat mengakses internet maka WSN ini dapat diakses dan berkolaborasi dengan sistem lain.



Gambar 2. 1 Jaringan *Node* WSN [13]

2.1.1. Karakteristik WSN

Beberapa karakteristik dari suatu WSN antara lain :

a. Daya atau *Power* yang terbatas

Jaringan WSN memiliki daya yang terbatas dikarenakan ukuran dari alat yang digunakan sangatlah kecil. Sehingga setiap sensor harus memiliki kemampuan untuk menyimpan daya secara efektif atau mengolahnya agar dapat bertahan dalam waktu yang lama.

b. Kemampuan untuk bertahan

Kemampuan untuk bertahan yang dimiliki WSN adalah agar dapat bertahan pada lingkungan yang tidak mudah untuk dijangkau dan di kontrol secara terus menerus. Sehingga WSN dapat bertahan pada kondisi yang sangat ekstrim pada saat digunakan.

c. Kemampuan untuk mengatasi kesalahan *node*

Kemampuan dari WSN untuk mengatasi kesalahan yang terjadi pada *node-node* yang ada pada jaringannya sehingga meningkatkan efektifitas dari jaringan WSN.

d. Mobilitas dari *Node*

Node dari jaringan WSN memiliki mobilitas sesuai dengan penempatan sensor *node* tersebut. Sehingga bila ditempatkan pada benda yang bergerak maka akan mengikuti mobilitas dari benda tersebut.

e. Topologi jaringan yang dinamis

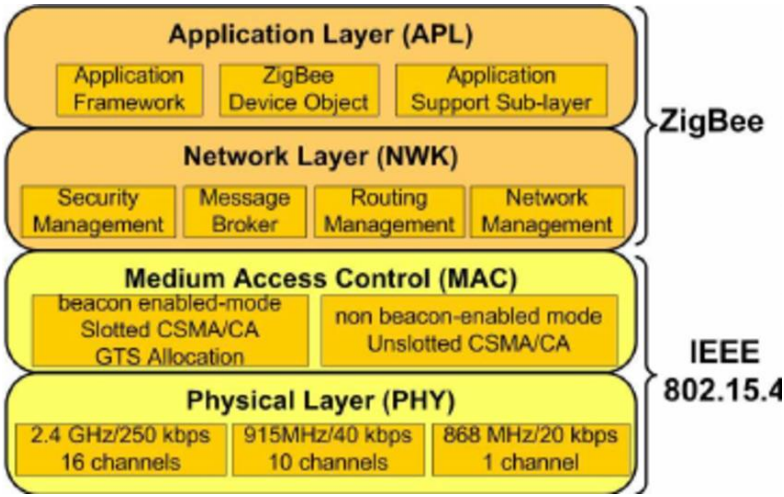
Topologi jaringan dinamis yang selalu berubah menyesuaikan lokasi atau benda dimana sensor *node* di letakkan.

f. Penyebaran dengan skala besar

Penyebaran sensor *node* jaringan WSN sangat besar untuk me-*monitor* banyak tempat maupun benda yang jumlahnya tidak sedikit.

2.1.2. Spesifikasi dan Standar WSN

Dalam Pengoperasiannya pada *Layer 2 OSI Layer*, *Wireless* Sensor menggunakan standar komunkasi *wireless* yaitu IEEE 802.15.4. Protokol IEEE 802.15.4 ini merupakan salah satu macam dari protokol-protokol pada WPAN (*Wireless* Personal Area Networks), salah satu contoh dari WPAN yang lainnya adalah bluetooth. Protokol IEEE 802.15.4 ini merupakan standar untuk gelombang radio (RF). Protokol ini bekerja pada *data rate* yang rendah agar baterai bisa tahan lama, dan sederhana. Suatu *device* yang menggunakan protokol ini, dapat terkoneksi dengan baik pada radius maksimal 10 m dan dengan *data rate* maksimal 250 Kbit/s dengan alat lainnya.



Gambar 2. 2 Arsitektur IEEE 802.15.4 [14]

2.1.3. Hardware WSN

Salah satu tantangan dalam bidang WSN adalah pembuatan sensor *node* (*mote*) yang hemat energi dan sekecil mungkin. *Mote* diibaratkan seperti komputer kecil, dengan kemampuan dasar seperti bagian tampilan dan komponen-komponen lain yang ada di dalamnya. Komponen-

komponen tersebut biasanya terdiri dari satu unit pemroses dengan kemampuan komputasi dan memori yang terbatas, sensor (yaitu sensor dengan kondisi yang khusus sesuai dengan penomena yang diindera), alat komunikasi (biasanya radio *trancievер*), sumber tenaga yang kecil, biasanya berbentuk baterai. Kemungkinan yang lain adalah sebuah modul pengolahan energi, dan alat komunikasi cadangan/kedua.

Sebuah *Mote* umumnya terdiri dari *Transceiver, Mikrokontroler, Power Source, External Memory* dan Sensor.

Gambar 2. 3 *Mote* WSN [13]

2.1.4. Software WSN

WSN juga membutuhkan perangkat lunak untuk mengontrol aktivitas-aktivitasnya dan membuat setiap perangkatnya dapat dimanfaatkan secara optimal, seperti sistem operasi dan aplikasi. Sistem operasi berfungsi menyediakan fungsi-fungsi dasar untuk mengatur kerja dari perangkat keras dan perangkat lunak yang berjalan di atasnya, termasuk menyediakan antarmuka pemrograman untuk pengembangan aplikasi WSN.

Beberapa system operasi WSN yang banyak digunakan diantaranya adalah *TinyOS, Contiki, Nano-RK, LiteOS dan RTOS*. Sistem operasi di atas dilengkapi antarmuka pemrograman aplikasi (API), komponen *software*, modul bawaan yang memudahkan pengembang aplikasi menulis aplikasi WSN-nya sendiri. Diantara bahasa

pemrograman yang banyak digunakan yaitu *C*, *C++*, *Java* dan *nesC* (*nesC* adalah bahasa pemrograman yang menggunakan dialek bahasa *C*, dan digunakan untuk menulis aplikasi di sistem operasi *TinyOS*). Adapun pengembangan aplikasi WSN secara umum mengikuti prinsip pengembangan aplikasi pada sistem tertanam (*embedded system*) dimana pengembang aplikasi menulis aplikasinya menggunakan komputer lain, kemudian meng-kompilasi dan menyambung kode aplikasi tersebut dengan sejumlah pustaka yang dibutuhkan termasuk penyambungan dengan kode sistem operasi, proses ini menghasilkan sebuah image yang telah berisi kode aplikasi dan sistem operasi yang siap di upload ke *Mote* untuk dieksekusi.

2.2 Protokol *Routing*

Protokol *routing* adalah serangkaian kerangka kerja dalam komunikasi *wireless* yang mengatur mulai dari proses pengiriman pesan mulai, mekanisme pembaruan tabel rute (*routing* tabel), serta penggunaan sumber daya. Protokol *routing* dibagi menjadi 2 jenis yaitu protokol *routing* statis dan protokol *routing* dinamis.

2.2.1 Protokol *Routing* Statis

Static routing adalah metode *routing* yang tabel jaringannya dibuat secara manual oleh administrator jaringan. *Static routing* mengharuskan admin untuk merubah route atau memasukkan command secara manual di *router* tiap kali terjadi perubahan jalur. *Router* meneruskan paket dari sebuah *network* ke *network* yang lainnya berdasarkan rute (seperti rute pada bis kota) yang ditentukan oleh administrator. Rute pada *static routing* tidak berubah, kecuali jika diubah secara manual oleh administrator.

Rute pada *routing* statis ditentukan dengan penggunaan tabel *routing* yang di konfigurasi oleh administrator jaringan. Tabel *routing* adalah sebuah tabel yang berisikan seluruh informasi *node* yang terhubung dengan *node* tersebut sehingga *node* yang satu bisa berkomunikasi atau mengirimkan paket melalui *node* yang terhubung hingga mencapai *node* tujuan. Dengan begitu pengiriman paket tidak akan bisa melalui *node* lain dalam jaringan tersebut yang tidak termasuk dalam pengaturan tabel *routing*.

Keuntungan dari *routing* statis adalah sebagai berikut :

- Meringankan kinerja prosesor *router*, karena *router* hanya mengupdate sekali saja ip tabel yang ada.
- Tidak ada *bandwidth* yang digunakan untuk pertukaran informasi dari tabel isi *routing* pada saat pengiriman paket.
- *Routing* statis lebih aman dibandingkan *routing* dinamis, karena *static router* menyediakan control penuh pada *routing* tabelnya.
- *Routing* Statis kebal dari segala usaha *hacker* untuk men-*spoof* dengan tujuan membajak trafik.
- Analisa kesalahan pada topologi jaringan lebih cepat diketahui.
- Pengiriman paket data yang lebih cepat karena jalur-jalur (path) sudah di ketahui terlebih dahulu.

Sementara kelemahan dari *routing* statis adalah sebagai berikut :

- Administrator jaringan harus mengetahui semua informasi dari masing-masing *router* yang digunakan.
- Hanya dapat digunakan untuk jaringan berskala kecil.
- Admisnistrasinya cukup rumit dibanding *routing* dinamis, terlebih jika banyak *router* yang harus dikonfigurasi secara manual.
- Rentan terhadap kesalahan saat entri data *routing* statis yang dilakukan secara manual.
- Selalu menggunakan rute yang sama yang kemungkinan bukan rute terbaik.
- Jika route berubah, *static router* harus diupdate secara manual.
- Konfigurasi *static routing* memiliki kompleksitas yang bergantung pada jumlah network yang terhubung.
- Jumlah *gateway* terbatas.

2.2.2 Protokol *Routing* Dinamis

Routing dinamis, juga disebut adaptif *routing*, yaitu suatu *routing* yang memiliki dan membuat tabel *routing* secara otomatis, dengan mendengarkan lalu lintas jaringan dan juga dengan saling berhubungan antara *router* lainnya. Protokol *routing* mengatur *router router* sehingga dapat berkomunikasi satu dengan yang lain dan saling memberikan informasi satu dengan yang lain dan saling memberikan informasi *routing* yang dapat mengubah isi *forwarding* tabel, tergantung keadaan

jaringannya. Dengan cara ini, *router- router* mengetahui keadaan jaringan yang terakhir dan mampu meneruskan data ke arah yang benar. Dengan kata lain, *routing* dinamik adalah proses pengisian data *routing* di tabel *routing* secara otomatis. *Dynamic router* mempelajari sendiri Rute yang terbaik yang akan ditempuhnya untuk meneruskan paket dari sebuah *network* ke *network* lainnya. Administrator tidak menentukan rute yang harus ditempuh oleh paket-paket tersebut. Administrator hanya menentukan bagaimana cara *router* mempelajari paket, dan kemudian *router* mempelajarinya sendiri. Rute pada *dynamic routing* berubah, sesuai dengan pelajaran yang didapatkan oleh *router*.

Kelebihan dari *routing* dinamis adalah sebagai berikut :

- Hanya mengenalkan alamat yang terhubung langsung dengan *router*nya (kaki-kakinya).
- Tidak perlu mengetahui semua alamat *network* yang ada.
- Bila terjadi penambahan suatu *network* baru tidak perlu semua *router* mengkonfigurasi. Hanya *router-router* yang berkaitan yang perlu di konfigurasi.

Sementara kelemahan dari *routing* dinamis adalah sebagai berikut :

- Beban kerja *router* lebih berat karena selalu memperbarui tabel *routing* pada setiap waktu tertentu atau bahkan setiap pengiriman paket berlangsung.
- Kecepatan pengenalan dan kelengkapan tabel *routing* terbilang lama karena *router* membroadcast ke semua *router* sampai ada yang cocok. Sehingga setelah konfigurasi harus menunggu beberapa saat agar setiap *router* mendapat semua alamat *router* yang ada.

2.2.2.1 Klasifikasi *Routing* Dinamis

Klasifikasi *routing* dinamis protokol *routing* dibagi menjadi dua, yaitu proaktif dan reaktif. Pada proaktif, *node* dalam jaringan melakukan pembaharuan terhadap tabel *routing*nya secara periodik (tabel driven), sedangkan reaktif hanya melakukan pembaharuan saat dibutuhkan/diminta saja (*on-demand*). Contoh protokol *routing* proaktif adalah OLSR, sedangkan contoh yang reaktif adalah AODV dan DSR. Kalsifikasi dari sifat *routing* dinamis tersebut dijelaskan pada sub bab berikutnya.

2.2.2.2 Sifat Proaktif dan Reaktif

a. Proaktif

Pada protokol *routing* proaktif , tabel *routing* (tabel berisi jumlah *node* dan rute *node*) diperbarui secara periodik. Hal tersebut mengakibatkan proses pengiriman pesan pada protokol proaktif sangat cepat dan kecilnya *Control Message* pengiriman pesan. Beberapa contoh dari protokol proaktif antara lain :

- OLSR (*Optimized Link State Protocol*)
- B.A.T.M.A.N (*Better Approach To Mesh Adhoc Network*)
- DSDV (*Destination Sequentced Distance-Vector*)
- BABEL
- FSR (*Fisheye State Routing*)

b. Reaktif

Pada protokol *routing* proaktif, rute baru dicari jika ada permintaan (*request*) untuk mengirimkan pesan. Keuntungan dari protokol reaktif adalah hemat sumber daya dan efisien paket *broadcast* karena hanya dikirim ketika mencari rute. Beberapa contoh protokol *routing* reaktif antara lain :

- AODV (*Adhoc On Demand Distance Vector*)
- DSR (*Dynamic Source Routing*)
- DYMO (*Dinamic MANET On Demand*)

c. Hybrid

Ada satu lagi sifat dari *routing* dinamis yaitu sifat gabungan dari sifat aktif dan reaktif. Sifat ini adalah sifat *hybrid*. Prinsip kerja protokol *routing* hybrid adalah menggabungkan mekanisme protokol proaktif dan protokol reaktif. Untuk mencari rute, mekanisme protokol reaktif digunakan yakni sesuai permintaan (request). Sedangkan untuk mekanisme memperbarui dan mempertahankan rute yang sudah dibangun, digunakan mekanisme protokol proaktif yakni dengan secara periodik memastikan rute yang telah dibangun. Beberapa contoh protokol *routing hybrid* antara lain :

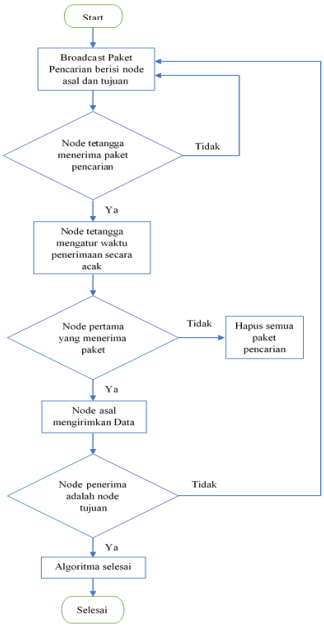
- ZRP (*Zone Routing Protocol*)
- HWMP (*Hybrid Wireless Mesh Protocol*)

2.3 *Random Walk*

Random walk merupakan sebuah teori dalam probabilitas yang menyatakan bahwa suatu pergerakkan bersifat acak (*random)*. *Random walk* termasuk dalam suatu proses stokastik yang bersifat diskret. Dalam *Random walk*, probabilitas untuk bergerak naik maupun turun adalah sama. Begitupun dengan *Random walk* yang simetrik dimana *Random walk* mempunyai probabilitas yang sama untuk dua nilai yang berbeda.

2.3.1 Algoritma *Random walk*

Algoritma *Random walk* adalah algoritma yang menyediakan jalur acak dalam grafik. *Random walk* berarti bahwa kita mulai dari satu *node*, memilih tetangga untuk menavigasi secara acak atau berdasarkan distribusi probabilitas yang disediakan, dan kemudian melakukan hal yang sama dari *node* itu, menghasilkan jalur yang tetap seperti daftar.



Gambar 2. 4 Diagram Algoritma *Random Walk*

Rumus dasar *Random walk* dapat ditulis sebagai berikut :

$$S_n = S_0 + \sum_{i=1}^{n} X_i$$

Dengan penjelasan sebagai berikut :

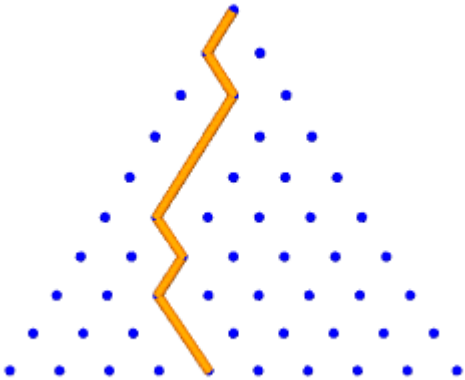
S : *State Space* (status), yang merupakan kumpulan nilai-nilai yang mungkin dari proses stokastik X_i

X_i : Variabel *Random* Bernoulli yang dapat bernilai +1 atau -1

Pembahasan selanjutnya mengenai *random walk* akan diawali dengan sebuah aplikasi. Sebuah aplikasi yang sederhana dari *random walk* adalah dalam *routing* atau penentuan jalur pengiriman data pada suatu jaringan, yaitu dimana suatu kondisi awal pengiriman data pada *node* tertentu dilambangkan dengan *S.* Pada interval diskret (misalnya pengiriman paket per detik), pemilihan dari jalur *routing* tersebut terhadap *node* tetangga yang berada dalam jangkauan memiliki nilai yang sama. Maka nilai kondisi pada *node* ke *n* atau *S(n)* adalah *S(n+1)=S(n)+1* atau *S(n+1)=S(n-1).* Oleh sebab itu maka peluang untuk *node* tetangga akan dipilih sebagai *node* berikutnya adalah sama besar.

2.3.2 Contoh *Routing Random walk*

Dalam *routing* berbasis *random walk*, paket data diteruskan ke salah satu *node* tetangga yang dipilih secara acak. Hal ini dapat dicapai dengan dua fase. Pada fase pertama dari pencarian *node* tetangga, paket pencarian *node* tetangga disebarkan dan sebagai balasannya seluruh *node* tetangga membalas dengan *ID* mereka. Setelah waktu tertentu ketika semua atau kebanyakan *node* tetangga diketahui sebagai *node* penerusan, *node* akan memilih salah satu tetangga secara acak sebagai *node* hop selanjutnya dan kemudian mengirim paket ke *node* tersebut. Begitu seterusnya hingga paket tiba pada *node* tujuan. Dari algoritma tersebut, dapat diketahui bahwa *Random walk* memiliki kemiripan dengan protokol *routing* dinamis dan reaktif.



Gambar 2. 5 Contoh *Routing Random Walk* [15]

2.4 Algoritma *Routing* Pembanding

Dengan penggunaan algoritma *routing Random walk* maka dibutuhkan beberapa algoritma *routing* lain sebagai pembanding performansi algoritma *routing Random walk* tersebut. Pada subbab ini akan dibahas mengenai 2 contoh protokol *routing* pembanding. Dengan 1 protokol *routing* mewakili protokol *routing* statis dan 1 lagi protokol *routing* yang mewakili protokol *routing* dinamis. Kedua *routing* tersebut adalah *routing* Waypoint mewakili *routing* statis dan AODV mewakili *routing* dinamis.

2.4.1. *Waypoint* (Statis)

Protokol *routing* Waypoint adalah protokol yang sering digunakan sebagai protokol *routing* statis. Protokol *routing* ini digunakan dengan cara mengatur tabel *routing* pada jaringan yang digunakan. Dengan pengaturan *routing* tabel maka jalur yang dilalui dapat ditentukan sejak awal sehingga jalur *routing* atau pengiriman data paketnya tidak menggunakan jalur lain selain jalur yang ditentukan sejak awal.

2.4.2. AODV

Adhoc On Demand Distance-Vector Routing atau disingkat AODV adalah salah satu protokol *routing* yang bersifat reaktif. Pada

AODV, pencarian rute (*routing*) hanya dilakukan jika terdapat permintaan, dalam hal ini permintaan untuk pengiriman pesan. Selain itu, AODV menerapkan algoritma *Distance-Vector*, yakni hanya menyimpan informasi tentang hop berikutnya ke tetangga terdekat dan panjang jalur ke semua tujuan yang diketahui.

Terdapat beberapa paket yang terlibat dalam protokol *routing* ini yaitu *Route Request* (RREQ), *Route Replay* (RREP), *Route Error* (RERR), dan HELLO MESSAGE. Proses *routing* pada protokol AODV dibagi menjadi beberapa bagian sebagai berikut.

a. *Route Discovery*

Pencarian rute (route discovery) dilakukan jika ada *node* yang ingin mengirimkan pesan ke *node* lain. Proses ini dimulai dengan pengiriman paket RREQ ke semua *node* ke jaringan hingga paket RREQ diterima oleh tujuan.

b. *Reverse Path Setup*

Ketika RREQ diterima *node* yang bukan tujuan, maka *node* tersebut akan meneruskan paket tersebut ke *node* lain. Selain meneruskan paket RREQ, *node* tersebut juga akan mencatat *reverse path* yang merupakan jalur asal paket RREQ yang diterima *node* tersebut. Ketika *node* tujuan telah ditemukan, maka paket RREP akan sampai ke *node* sumber melalui kumpulan *reverse path* tersebut.

c. *Route Maintenance* dan *Link Breakage*

Dikarenakan pada jaringan MANET *node* bebas bergerak, maka *node* bisa saja keluar dari jaringan. Hal tersebut mengakibatkan jalur/koneksi terputus atau yang disebut *link breakage*. Cara mengetahui bahwa *ada link breakage* adalah jika suatu *node* tidak menerima HELLO MESSAGE dalam interval tertentu (hello interval). *Route maintenance* pada AODV juga menggunakan HELLO MESSAGE.

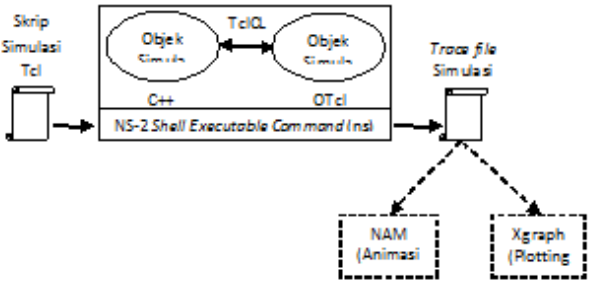
2.5 *Network simulator* Versi 2 (NS2)

Network simulator (NS2) adalah alat simulasi jaringan yang bersifat *open source* yang banyak digunakan dalam mempelajari struktur dinamik dari jaringan komunikasi. Simulasi dari jaringan nirkabel dan protokol (seperti algoritma *routing*, TCP, dan UDP) dapat diselesaikan dengan baik dengan simulator ini. Beberapa keuntungan menggunakan *network*

simulator sebagai perangkat lunak simulasi adalah *network simulator* dilengkapi dengan *tool* validasi, pembuatan simulasi dengan menggunakan *network simulator* jauh lebih mudah daripada menggunakan *software developer* seperti *Delphi* atau *C++*, *network simulator* bersifat *open source* di bawah GPL (*Gnu Public License*). *Network simulator* ini dapat digunakan pada sistem operasi *windows* dan sistem operasi *linux*.

2.5.1. Arsitektur Dasar

NS2 terdiri dari dari 2 bahasa utama yaitu *C++* dan *Objectoriented Tool Command Language* (Otcl). Apabila *C++* mendefinisikan mekanisme internal dari objek simulasi, maka Otcl menyusun simulasi dengan mengumpulkan dan mengatur objek. *C++* dan Otcl terhubung oleh TclCl. Arsitektur dasar dari NS dapat digambarkan seperti berikut:



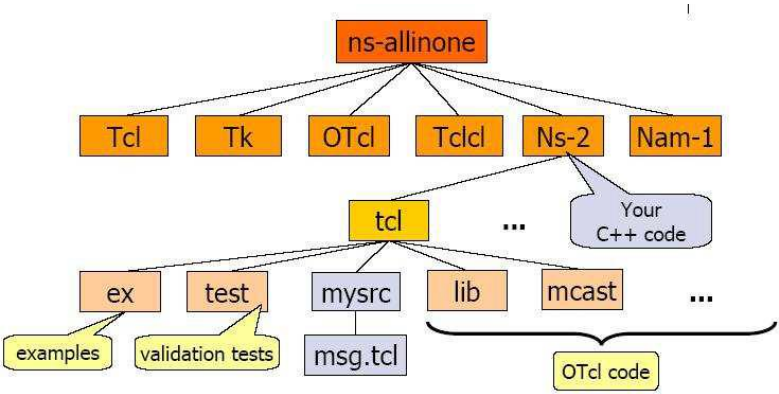
Gambar 2. 6 Arsitektur NS2 [16]

Setelah simulasi, output dari hasil simulasi NS2 berupa *text-based* dan *animation-based*. Untuk menginterpretasi hasil *output* secara grafis dan interaktif, digunakan sebuah tool seperti NAM (*Network Animator*) dan *Xgraph*. Untuk melakukan analisa *behavior* dari jaringan, pengguna dapat mengekstrak bagian yang relevan dari hasil yang *text-based* dan mengubah ke dalam bentuk yang dapat dipahami.

2.5.2. Komponen Pembangun NS2

Pembangun aplikasi NS2 sehingga dapat bekerja sebagaimana mestinya, terdiri dari beberapa komponen. Komponen-komponen tersebut akan saling bekerja sama sehingga simulasi pada NS2 dapat

dijalankan. Komponen pembangun NS2 dapat dilihat seperti gambar dibawah ini :



Gambar 2. 7 Komponen Pembangun NS2 [16]

Keterangan :

Tcl (*Tool command language*) : *Scripting programming* untuk konfigurasi *network simulator*

Otcl (*Object* Tcl) : Tcl *Interpreter* yang melakukan inisiasi *event scheduler*, membangun topologi jaringan berbasis objek serta memberitahu sumber trafik saat memulai dan mengakhiri pengiriman paket melalui *event scheduler*.

TK : Tool Kit

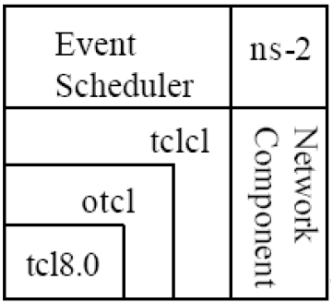
Tclcl : merupakan bahasa pemrograman untuk menyediakan linkage antara *C++* dan OTcl berupa *class hierarchy, object instantiation, variable binding* dan *command dispatching.*

NS2 : *Network simulator* versi 2

Nam (*Network animator*) : NAM menyediakan interpretasi visual dari topologi jaringan yang dibuat.

2.5.3. Hubungan antar Komponen Pembangun NS2

Gambar dibawah ini merepresentasikan struktur umum hubungan antar komponen pembangun NS2.



Gambar 2. 8 Hubungan Antar Komponen NS2 [16]

Pada deskripsi ini pengguna NS berada pada pojok kiri bawah, melakukan desain dan menjalankan simulasi dalam bahasa Tcl. Dalam simulasi, pengguna memanggil dan menggunakan objek simulator pada *library* Otcl. *Event Scheduler* dan sebagian besar *network component* pada NS ditulis dalam bahasa *C++*. Ini diakses oleh Otcl melalui Otcl *linkage* yang diimplementasikan dengan menggunakan Tclcl.

2.6 *Quality of Service*

Quality of Service (QOS) dapat dikatakan sebagai satu terminologi yang digunakan untuk mendefinisikan karkateristik suatu layanan (*service)* jaringan guna mengetahui seberapa baik kualitas dari layanan tersebut. Kualitas layanan diukur dengan standar dari *Telecommunications and Internet Protocol Harmonization Over Networks* (TIPHON). Dalam tugas akhir ini parameter QoS yang akan dianalisa adalah *delay, throughput* dan *packet delivery ratio.*

2.6.1. *Delay*

Delay didefinisikan sebagai selisih waktu pengiriman sebuah paket saat dikirimkan dengan saat paket tersebut diterima pada *node* tujuan. *Delay* disebut juga dengan istilah *latency* terdiri dari beberapa faktor penundaan yaitu *propagation delay* atau *transmission delay* yaitu penundaan akibat waktu tempuh paket selama dalam saluran transmisi

yang *bandwidth* nya berbeda-beda, *queuing delay* yaitu waktu antrian paket sebelum dilewatkan pada saluran transmisi dan lainnya. Pada tugas akhir ini *delay* yang diukur adalah *end to end delay* (*delay* keseluruhan dan sudah di rata-rata) yang dinyatakan dalam satuan detik (s).

2.6.2. *Throughput*

Throughput adalah total paket data yang diterima dengan sukses dibagi waktu pengiriman data. Pada umumnya Throughput merepresentasikan kecepatan pengiriman data dari suatu sistem komunikasi.

Perhitungan *throughput* sebagai berikut :

$$T = \frac{S}{W}$$

Dengan penjelasan sebagai berikut :

T : *Throughput (bit/sec)*
S : Jumlah data yang dikirim (*bit*)
W : Waktu pengiriman data (*sec*)

2.6.3. *Packet Delivery Ratio*

Packet Delivery Ratio (PDR) adalah perbandingan antara paket yang diterima dengan paket yang dikirim. PDR dapat mendeskripsikan rasio paket-paket yang berhasil dikirimkan menuju *node* penerima selama proses pengiriman data.

Perhitungan *Packet Delivery Ratio* (PDR) sebagai berikut :

$$PDR = \frac{PK_R}{PK_S} x100\%$$

Dengan penjelasan sebagai berikut :

PDR : *Packet Delivery Ratio* (%)
PK_R : Paket yang diterima oleh *node* penerima
PK_S : Paket yang dikirim oleh *node* pengirim

2.7 *Tracegraph* Versi 2

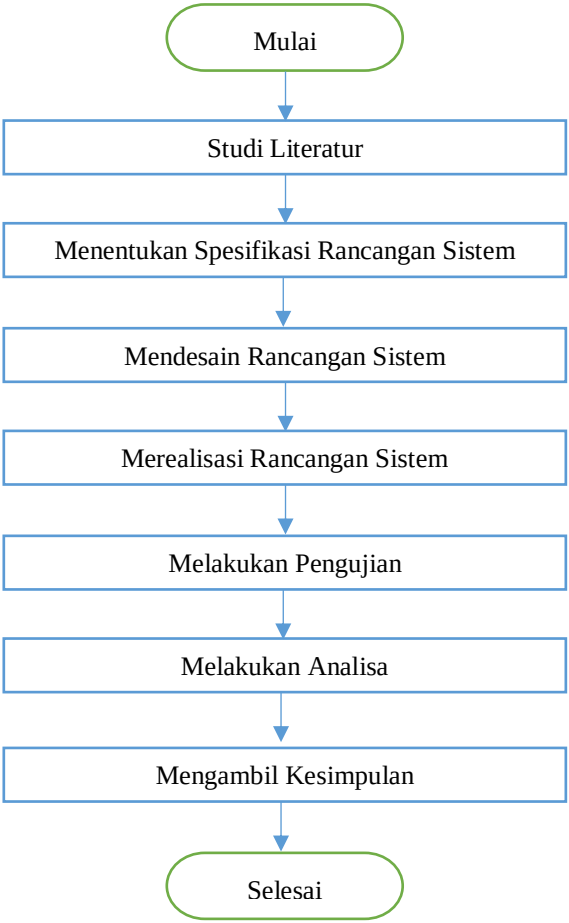
Setelah skenario simulasi dijalankan pada simulator NS2, didapat *trace file* yang merupakan *log* kejadian sepanjang simulasi. *Trace file* ini kemudian diproses menggunakan suatu *tool* yaitu *Tracegraph*. *Tracegraph* adalah *freeware* yang berfungsi sebagai *analyzer trace file* hasil *output* dari simulator NS2. *Tracegraph* dibuat oleh Jaroslaw Malek dari *Wroclaw University of Technology*, Polandia pada tahun 2001 dan masih dikembangkan hingga saat ini. *Tracegraph* menerima *input* berupa *trace file* dan dapat menganalisis semua format *trace file* dari NS2. *Tracegraph* dapat berjalan di atas sistem operasi *Windows, Linux,* dan *MAC*. Untuk dapat menjalankan *Tracegraph*, diperlukan program *Matlab* versi 6.1 keatas. Hasil analisis *Tracegraph* dapat berupa data (teks) maupun grafik (2D dan 3D). Informasi yang dapat diberikan *Tracegraph* antara lain informasi *throughput, delay, jitter*, dan lain-lain.

Halaman ini sengaja dikosongkan

BAB 3
METODOLOGI PENELITIAN

3.1 Diagram Alur Penelitian

Penelitian ini dimulai dengan mengikuti diagram alur yang terdiri dari beberapa tahap. Tahapan-tahapan tersebut direpresentasikan pada diagram alur penelitian sebagai berikut.



Gambar 3. 1 Diagram Alur Penelitian

Pada tugas akhir ini, simulasi jaringan WSN dengan implementasi algoritma *Random walk* dilakukan dengan simulator jaringan NS2. Yang dijelaskan pada bab ini. Perangkat yang digunakan untuk melakukan simulasi ini berupa sebuah PC/Laptop Dell Inspiron 7447 dengan *Operating System* (OS) Ubuntu Versi 14.04 LTS yang sudah diinstal pada Virtual Box.

3.2 Spesifikasi Rancangan Sistem

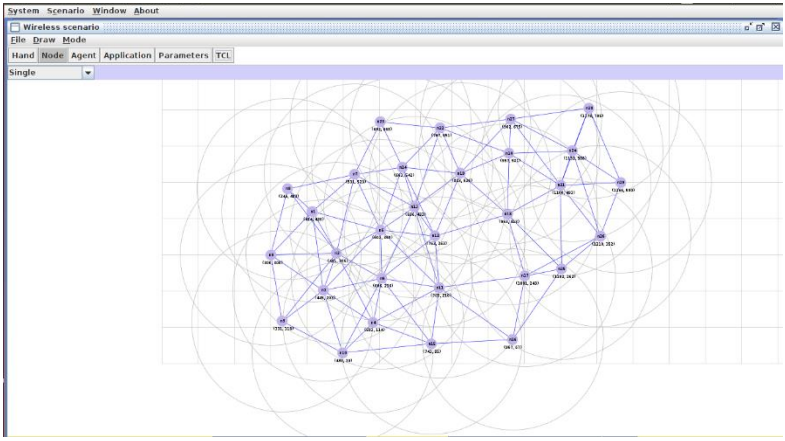
Pada penelitian ini akan dilakukan pengujian sebanyak 2 kali dengan waktu simulasi yang berbeda yaitu 5 detik dan 15 detik. Dengan spesifikasi jaringan secara umum dilakukan pada daerah simulasi seluas 1300 x 800 meter dengan tipe jaringan berupa *wireless*. Pada jaringan ini akan terdapat sebanyak 30 *node* sensor yang diletakkan secara acak dengan 1 *node* sebagai *node* awal (*source*) dan 1 *node* lainnya sebagai tujuan (*destination*). Untuk ukuran paket yang digunakan memiliki ukuran sebesar 1500 *bytes.* Jenis protokol *routing* yang digunakan adalah *Random Walk* sebagai protokol utama dengan protokol *routing* AODV dan *Waypoint* sebagai pembanding. Adapun rincian dari spesifikasi jaringan WSN dijelaskan pada Tabel 3.1 berikut ini.

Tabel 3. 1 Spesifikasi Rancangan Sistem

No.	Parameter	Nilai	Satuan
1	Luas Area	1300 x 800	Meter
2	Jumlah *Node* Sensor	30	
3	Protokol Mac	802.11	
	Message Size	1500	bytes
4	Waktu Simulasi	5, 15	Sekon
5	*Protokol Routing (Utama)*	Random Walk (RW)	
6	*Protokol Routing (Pembanding)*	AODV, *Waypoint* (WP)	
7	*Radius Jangkauan Sensor*	250	Meter
8	Protokol Transport / Transmisi (Utama)	TCP	
9	Protokol Transport / Transmisi (Pembanding)	UDP	
10	Protokol Aplikasi / Transfer Data	FTP	
11	Tipe Jaringan	*Wireless*	

3.3 Desain Rancangan Sistem WSN

Mendesain rancangan sistem jaringan WSN menggunakan NS2, dilakukan dengan membuat program dengan bahasa tcl. Untuk memudahkan dalam visualisasi desain topologi jaringan, digunakan juga aplikasi berbasis *java* yaitu *Network simulator Generator* (NSG). Aplikasi tersebut digunakan untuk men-*generate file* tcl yang nantinya digunakan untuk memperlihatkan simulasi jaringan WSN.

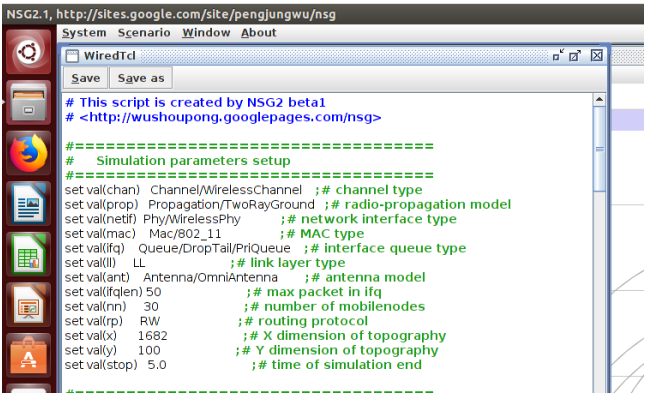


Gambar 3. 2 Tampilan NSG

Pada tampilan tatap muka aplikasi NSG terdapat banyak menu diantaranya *System, Scenario, Window* dan *About*. Menu utama yang digunakan adalah *Scenario* dimana dialam menu tersebut berisi 2 skenario yang dapat dibangkitkan dari aplikasi NSG ini. Skenario dalam menu *Scenario* yang dapat dibangkitkan terdiri dari *Wired Scenario* untuk skenario simulasi dengan jaringan bertipe kabel dan *Wireless Scenario* untuk skenario simulasi dengan jaringan bertipe nirkabel.

3.3.1. Men-*generate file* tcl

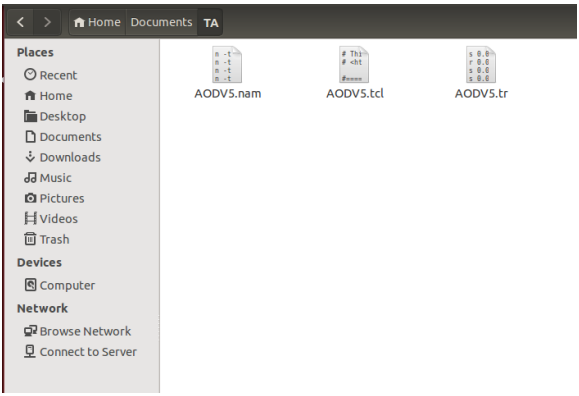
Dalam men-generate *file* tcl, maka dilakukan dengan merancang topologi dan spesifikasi sesuai tabel spesifikasi jaringan yang sudah dijelaskan pada subbab sebelumnya. Setelah mengatur spesifikasi sesuai yang diinginkan pada NSG, save *file* tcl pada pilihan *user interface* NSG dan *file* tcl dapat di *generate* dan disimpan pada folder yang diinginkan.



Gambar 3. 3 *Generate File* Tcl

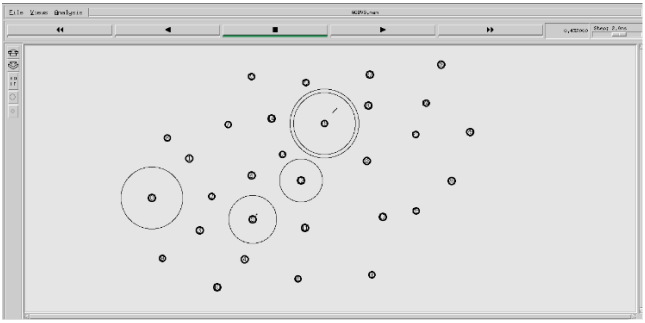
3.3.2. Hasil Simulasi

Hasil simulasi dari *file* tcl yang sudah di generate berupa *file* network animator dengan format .nam yang otomatis ter-generate pada folder yang sama dengan *file* tcl. *File* nam akan menunjukkan visualisasi dari simulasi yang dilakukan dengan pengaturan *file* tcl tersebut. Selain *file* nam, *file* trace dengan format .tr juga otomatis ter-generate pada folder yang sama dengan *file* tcl dan nam. *File* trace dengan format .tr merupakan *file* berisi hasil simulasi *routing* yang telah dilakukan.



Gambar 3. 4 *Folder* Penyimpanan *File* Simulasi

Dengan menjalankan *file* tcl yang terdapat pada *folder* hasil simulasi, maka *file* NAM akan otomatis di eksekusi dan akan menampilkan visual dari simulasi yang telah di-generate. Tampilan simulasi pada NAM dapat berupa aliran-aliran data yang dikrimkan di jaringan, jalur-jalur atau rute-rute yang dipilih untuk pengiriman data, dimana paket akan *drop* atau *loss*, dan lain sebagainya yang ada pada jaringan selama masih bisa di visualisasikan oleh *Network Animator* (NAM).



Gambar 3. 5 Tampilan Simulasi *Network Animator* (NAM)

3.4 Skenario Pengujian

Pengujian simulasi akan dilakukan dalam 3 skenario berdasarkan penggunaan protokol *routing Random walk* dengan 2 protokol pembanding yaitu *Waypoint* dan AODV. Dari ketiga simulasi dengan protokol *routing* tersebut hasilnya akan dibandingkan satu sama lain untuk mengetahui bagaimana performansi protokol *routing random walk* terhadap protokol *routing* lain.

3.4.1 Lokasi Node Sensor

Dalam tugas akhir ini digunakan *node* sensor sebanyak 30 *node* yang diletakkan pada area yang dibatasi dengan luasan sebesar 1300 meter x 800 meter. Distribusi lokasi *node* tertera pada Tabel 3.2 dibawah ini. Dengan lokasi tersebut akan diharapkan hasil *routing* dengan jalur alternatif yang banyak sehingga jalur *routing*-nya dapat terlihat pada tampilan simulasi. Dari distribusi *node* tersebut juga dipilih agar tidak terjadi penumpukan *node* hanya pada suatu area ataupun sebaliknya. Seluruh 30 *node* sensor yang digunakan memiliki radius jangkauan sejauh 250 meter.

Tabel 3. 2 Koordinat Node Sensor

No.	*Node*	Koordinat	
		x	y
1	*Node*-0	300	300
2	*Node*-1	414	420
3	*Node*-2	481	305
4	*Node*-3	445	203
5	*Node*-4	582	114
6	*Node*-5	606	236
7	*Node*-6	603	369
8	*Node*-7	531	523
9	*Node*-8	346	483
10	*Node*-9	331	118
11	*Node*-10	498	30
12	*Node*-11	765	210
13	*Node*-12	753	353
14	*Node*-13	696	433
15	*Node*-14	663	542
16	*Node*-15	743	55
17	*Node*-16	967	67
18	*Node*-17	1001	243
19	*Node*-18	953	413
20	*Node*-19	823	526
21	*Node*-20	957	581
22	*Node*-21	1100	493
23	*Node*-22	767	651
24	*Node*-23	602	668
25	*Node*-24	1132	588
26	*Node*-25	1102	262
27	*Node*-26	1210	352
28	*Node*-27	962	675
29	*Node*-28	1178	705
30	*Node*-29	1266	500

3.4.2 Pengujian Menggunakan Protokol *Random walk*

Pengujian ini dilakukan dengan menggunakan ukuran paket sebesar 1500 *bytes* dan mengatur protokol *routing* yang digunakan dengan protokol *random walk*. Adapun langkah-langkah pengujiannya adalah sebagai berikut :

1. Membuka aplikasi NSG melalui terminal OS Ubuntu.
2. Pada aplikasi NSG, pilih menu *scenario* kemudian pilih *new wireless scenario.*

3. Kemudian pada menu *Node* pilih *Single*, lalu pada area peletakkan *node*, letakkan *node* sesuai dengan distribusi *node* pada Tabel 3.2.
4. Pada menu *Parameters*, pilih tab *Simulation* untuk mengatur waktu simulasi yaitu 5 detik (s) sekaligus mengatur nama file nam dan juga file trace yang nantinya dihasilkan.
5. Masih pada menu *Parameters*, pilih tab *Wireless* untuk mengatur protokol *routing* yang digunakan yaitu *Random Walk*. Sementara untuk pengaturan lain tidak perlu dilakukan perubahan.
6. Menentukan *node* sumber (*source*) dan *node* tujuan (*destination*). Untuk tahap pertama dipilih *node* 0 sebagai *source* dan *node* 1 sebagai *destination*.
7. Menentukan *node* sumber dan tujuan dengan penggunaan tcp atau udp sekaligus menentukan ukuran paket. Penggunaan TCP atau UDP dapat dipilih pada menu *Agent* pada NSG. TCP atau UDP di-*attach* pada *node* sumber dan tujuan yang dipilih.
8. Melakukan pembangkitan paket yang akan dikirim dengan cara *attach* FTP pada *node* sumber yang sudah dipilih dengan pengaturan TCP atau UDP. Pengaturan ini dilakukan dengan memilih menu *Application* pada NSG.
9. Melakukan generate *file* tcl yang sudah diatur pada langkah sebelumnya (kompilasi program). Hal ini dapat dilakukan dengan memilih menu TCL pada layar NSG dan kemudian pilih save untuk men-*generate* dan menyimpan *file* tcl pada folder yang diinginkan.
10. Menjalankan simulasi pada Network Animator (NAM) dengan *command* "ns <nama *file* tcl>" pada terminal OS Ubuntu. Dalam tugas akhir ini nama *file* tcl untuk penggunaan protokol *routing Random Walk* adalah RW.tcl.
11. Mengganti setting waktu simulasi menjadi 15 detik (s)
12. Ulangi langkah 5-10
13. Ganti *node destination* menjadi *node* 2
14. Ulangi langkah 4-11
15. Ganti lagi *node destination* menjadi *node* 3
16. Ulangi langkah 4-11
17. Dan ganti lagi *node destination* menjadi *node* 4 dan berikutnya sampai *node* 29
18. Ulangi langkah 4-11

Langkah-langkah pengujian diatas akan menghasilkan *file* tcl, *file* nam dan *file* tr dalam satu folder penyimpanan. Nantinya *file* tr yang berisikan hasil simulasi akan dianalisa.

3.4.3 Pengujian Menggunakan Protokol Waypoint

Pengujian kedua dilakukan dengan mengatur protokol *routing* yang digunakan dengan protokol waypoint (jalur *routing* yang ditentukan). Ukuran paket masih sama seperti pada *Random Walk* yaitu 1500 *bytes.* Karena ditujukan sebagai pembanding, *node-node* yang dipilih hanya beberapa *node destination*. *Node* yang dipilih adalah *node-9, node-16, node-22, node-28* dan *node-29.* Adapun langkah-langkah pengujiannya adalah sebgai berikut :

1. Membuka aplikasi NSG melalui terminal OS Ubuntu.
2. Pada aplikasi NSG, pilih menu *scenario* kemudian pilih *new wireless scenario.*
3. Kemudian pada menu *Node* pilih *Single,* lalu pada area peletakkan *node,* letakkan *node* sesuai dengan distribusi *node* pada Tabel 3.2.
4. Pada menu *Parameters,* pilih tab *Simulation* untuk mengatur waktu simulasi yaitu 5 detik (s) sekaligus mengatur nama file nam dan juga file trace yang nantinya dihasilkan.
5. Masih pada menu *Parameters,* pilih tab *Wireless* untuk mengatur protokol *routing* yang digunakan yaitu *Waypoint.* Sementara untuk pengaturan lain tidak perlu dilakukan perubahan.
6. Menentukan *node* sumber (*source*) dan *node* tujuan (*destination*). Untuk tahap pertama dipilih *node* 0 sebagai *source* dan *node* 9 sebagai *destination*.
7. Menentukan *node* sumber dan tujuan dengan penggunaan tcp atau udp sekaligus menentukan ukuran paket. Penggunaan TCP atau UDP dapat dipilih pada menu *Agent* pada NSG. TCP atau UDP di-*attach* pada *node* sumber dan tujuan yang dipilih.
8. Melakukan pembangkitan paket yang akan dikirim dengan cara *attach* FTP pada *node* sumber yang sudah dipilih dengan pengaturan TCP atau UDP. Pengaturan ini dilakukan dengan memilih menu *Application* pada NSG.
9. Melakukan generate *file* tcl yang sudah diatur pada langkah sebelumnya (kompilasi program). Hal ini dapat dilakukan dengan

memilih menu TCL pada layar NSG dan kemudian pilih save untuk men-*generate* dan menyimpan *file* tcl pada folder yang diinginkan.

10. Menjalankan simulasi pada Network Animator (NAM) dengan *command* "ns <nama *file* tcl>" pada terminal OS Ubuntu. Dalam tugas akhir ini nama *file* tcl untuk penggunaan protokol *routing Waypoint* adalah Waypoint.tcl.
11. Mengganti setting waktu simulasi menjadi 15 detik (s)
12. Ulangi langkah 5-10
13. Ganti *node destination* menjadi *node* 16
14. Ulangi langkah 4-11
15. Ganti lagi *node destination* menjadi *node* 22
16. Ulangi langkah 4-11
17. Ganti lagi *node destination* menjadi *node* 28
18. Ulangi langkah 4-11
19. Ganti lagi *node destination* menjadi *node* 29
20. Ulangi langkah 4-11

Dari langkah-langkah pengujian diatas maka akan didapat *file* tcl, *file* nam dan *file* tr. Dimana nantinya *file* tr yang berisikan hasil simulasi akan dianalisa.

3.4.4 Pengujian Menggunakan Protokol AODV

Pengujian pembanding yang terakhir adalah mengatur protokol *routing* yang digunakan dengan protokol AODV (*Ad hoc On-Demand Distance Vector*). Karena juga merupakan protokol pembanding, maka *node-node* yang dipilih sebagai *node source* dan *destination* sama seperti percobaan sebelumnya. Begitupun ukuran paket yang digunakan yaitu 1500 *bytes.* Adapun langkah-langkah pengujiannya adalah sebagai berikut :

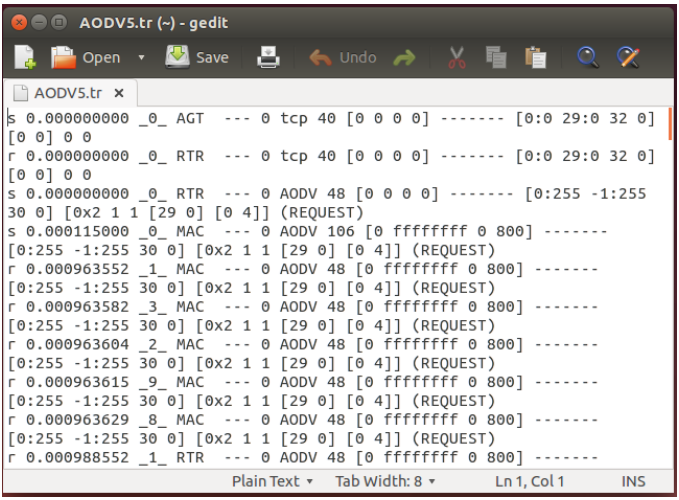
1. Membuka aplikasi NSG melalui terminal OS Ubuntu.
2. Pada aplikasi NSG, pilih menu *scenario* kemudian pilih *new wireless scenario.*
3. Kemudian pada menu *Node* pilih *Single*, lalu pada area peletakkan *node*, letakkan *node* sesuai dengan distribusi *node* pada Tabel 3.2.

4. Pada menu *Parameters*, pilih tab *Simulation* untuk mengatur waktu simulasi yaitu 5 detik (s) sekaligus mengatur nama file nam dan juga file trace yang nantinya dihasilkan.
5. Masih pada menu *Parameters*, pilih tab *Wireless* untuk mengatur protokol *routing* yang digunakan yaitu *Waypoint*. Sementara untuk pengaturan lain tidak perlu dilakukan perubahan.
6. Menentukan *node* sumber (*source*) dan *node* tujuan (*destination*). Untuk tahap pertama dipilih *node* 0 sebagai *source* dan *node* 9 sebagai *destination*.
7. Menentukan *node* sumber dan tujuan dengan penggunaan tcp atau udp sekaligus menentukan ukuran paket. Penggunaan TCP atau UDP dapat dipilih pada menu *Agent* pada NSG. TCP atau UDP di-*attach* pada *node* sumber dan tujuan yang dipilih.
8. Melakukan pembangkitan paket yang akan dikirim dengan cara *attach* FTP pada *node* sumber yang sudah dipilih dengan pengaturan TCP atau UDP. Pengaturan ini dilakukan dengan memilih menu *Application* pada NSG.
9. Melakukan generate *file* tcl yang sudah diatur pada langkah sebelumnya (kompilasi program). Hal ini dapat dilakukan dengan memilih menu TCL pada layar NSG dan kemudian pilih save untuk men-*generate* dan menyimpan *file* tcl pada folder yang diinginkan.
10. Menjalankan simulasi pada Network Animator (NAM) dengan *command* "ns <nama *file* tcl>" pada terminal OS Ubuntu. Dalam tugas akhir ini nama *file* tcl untuk penggunaan protokol AODV adalah AODV.tcl.
11. Mengganti setting waktu simulasi menjadi 15 detik (s)
12. Ulangi langkah 5-10
13. Ganti *node destination* menjadi *node* 16
14. Ulangi langkah 4-11
15. Ganti lagi *node destination* menjadi *node* 22
16. Ulangi langkah 4-11
17. Ganti lagi *node destination* menjadi *node* 28
18. Ulangi langkah 4-11
19. Ganti lagi *node destination* menjadi *node* 29
20. Ulangi langkah 4-11

Masih sama seperti pengujian sebelumnya, skenario pengujian diatas menghasilkan *file* tcl, *file* nam dan *file* tr. *File* tr yang berisikan hasil simulasi akan dianalisa pada pengolahan hasil *file trace*.

3.5 Pengolahan Hasil *File* Trace

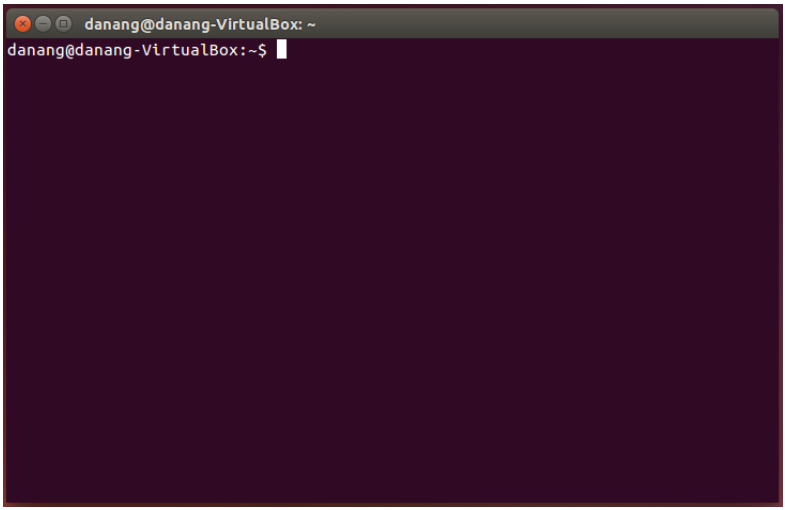
Hasil simulasi menggunakan software *Network simulator* versi 2 (NS2) dari skenario pengujian membentuk *file* hasil *trace* yang berisi karakteristik dari sistem jaringan. *File* yang terbentuk berformat tr yang secara otomatis terbentuk dan berada di dalam satu folder dengan *file* eksekusi (.tcl). Analisa hasil simulasi pengujian terdapat pada *file* berformat tr dapat dilakukan dengan cara memanggil *file* tersebut melalui terminal dengan perintah gedit. Tampilan *file* .tr yang otomatis terbentuk dalam folder simulasi dapat dilihat pada gambar dibawah ini.



Gambar 3. 6 Tampilan Isi File .tr

Untuk melakukan pengamatan simulasi, dapat dilakukan dengan NAM. Setelah sebelumnya mengatur spesifikasi jaringan, *file* akan tersimpan pada folder yang telah dipilih dengan format tcl. Dalam menjalankan *file* berformat tcl tersebut, terminal pada OS Ubuntu 14.04

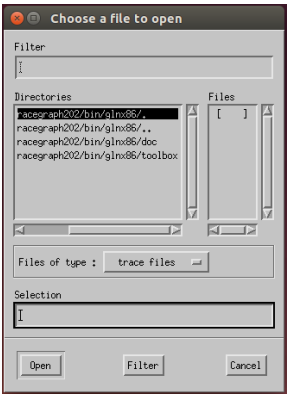
LTS harus dibuka terlebih dahulu. Kemudian menuju folder penyimpanan *file* tcl pada terminal tersebut.



Gambar 3. 7 Tampilan Terminal OS Ubuntu

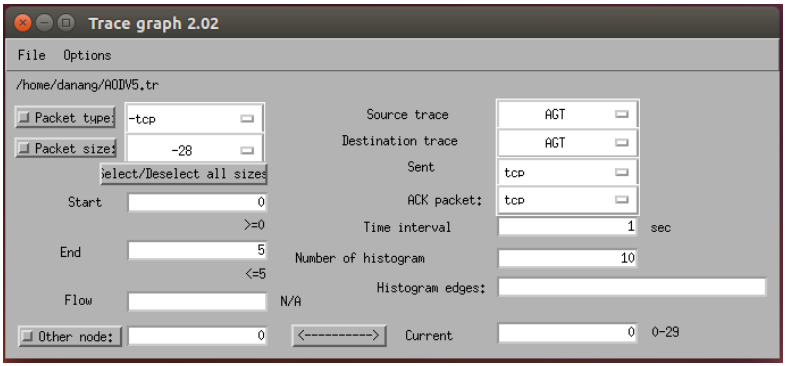
Setelah menuju folder penyimpanan *file* tcl pada terminal, kemudian mengetik ns <nama *file*>. Kemudian akan muncul tampilan animasi (NAM) jaringan yang telah dibuat. Pada NAM dapat melihat kondisi aliran data paket, topologi jaringan dan waktu yang digunakan, semua dalam bentuk simulasi. Pada NAM juga dapat dilakukan pelambatan simulasi yang dilakukan untuk memperjelas pemilihan jalur *routing* yang digunakan.

Dari *file* format tr tersebut digunakan kembali dengan *software* Tracegraph 2.02 untuk memperoleh hasil nilai parameter dan graph. Langkah awal analisa hasil simulasi dengan *software* Tracegraph 2.02, melalui terminal yang dipanggil dengan mengarahkan ke lokasi *file software* Tracegraph 2.02. Berikut adalah tampilan awal Tracegraph:



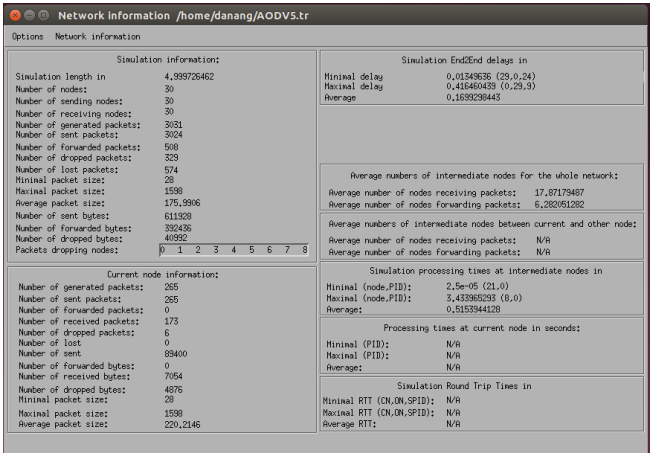
Gambar 3. 8 Tampilan Awal Tracegraph

Dari hasil simulasi skenario pengujian yang dilakukan pada *Network simulator* versi 2 (NS2) maka dapat dilakukan analisa dengan software Tracegraph 2.02. Berikut merupakan tampilan karakteristik system jaringan yang berhasil di analisa dengan Tracegraph.



Gambar 3. 9 Tampilan Spesifikasi Jaringan Pada Tracegraph

Hasil informasi lengkap tentang skenario pengujian yang telah disimulasikan mengenai kinerja jaringan ditampilkan secara numerik pada jendela Tracegraph. Berikut tampilan informasi lengkap :



Gambar 3. 10 Tampilan Hasil Simulasi Pada Tracegraph

Berdasarkan parameter pengujian, dilakukan analisa terhadap ketiga jenis *routing* dengan kondisi simulasi yang sudah ditentukan sebelumnya. Selanjutnya dilihat hasil perhitungan dari *delay*, *throughout* dan *packet delivery ratio* (PDR) sebagai perbandingan performansi dari ketiga protokol *routing* tersebut.

BAB 4
HASIL PENGUJIAN DAN ANALISA

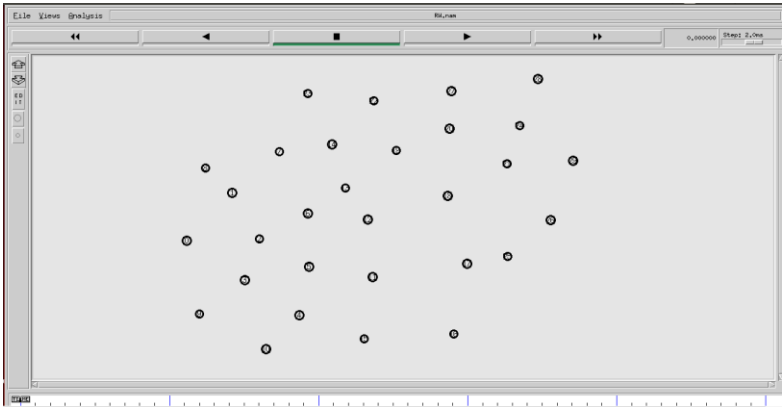
Hasil dari penelitian akan dijelaskan dalam 4 subbab besar yakni dari hasil pengujian simulasi, hasil pengukuran protokol random walk, hasil pengukuran protokol pembanding dan pembahasan. Hasil pengujian simulasi terdiri dari hasil simulasi masing-masing protokol *routing* yang diuji. Protokol pembanding berisi performansi protokol *routing* pembanding yang digunakan. Dan pembahasan merupakan pembahasan performansi *routing random walk* terhadap protokol *routing* pembanding.

4.1 Hasil Pengujian Simulasi

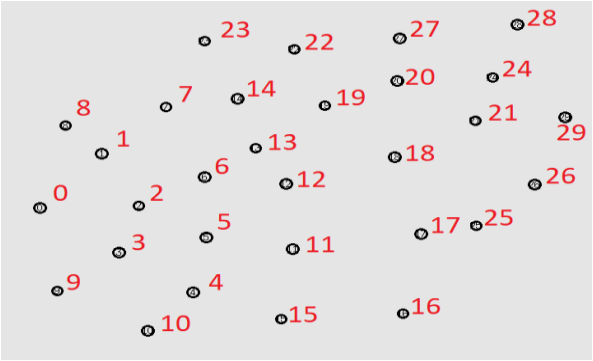
Berikut adalah hasil pengujian simulasi dengan NS2 yang telah dilakukan.

4.1.1 Hasil Tampilan *Node-Node* Sensor

Berikut adalah hasil simulasi letak lokasi *node* pada tampilan *Network Animator* (NAM) dengan letak *node* yang ditentukan sesuai tabel koordinat *node*.



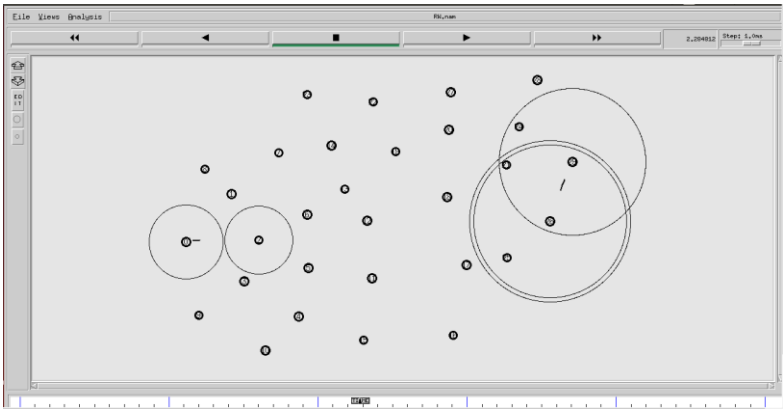
Gambar 4. 1 Sebaran Posisi *Node-Node* Sensor



Gambar 4. 2 Penomoran *Node* Sensor

4.1.2 Hasil *Running* Simulasi Protokol *Random Walk*

Sebagai gambaran awal bagaimana simulasi berjalan pada NAM, berikut adalah contoh running simulasi menggunakan protokol *random walk* dengan NS2 yang telah dilakukan.



Gambar 4. 3 Contoh *Running* Protokol *Random Walk*

Dari *running* simulasi dengan *Network Animator* (NAM) dapat dilihat bagaimana pergerakan *node* (jika *node* bergerak), pergerakan transmisi data, pemilihan jalur-jalur atau rute-rute transmisi data (*routing*) dan lain sebagainya.

4.2 Hasil Pengukuran Parameter Protokol *Random Walk*

Pada sub-bab ini dilakukan pengambilan data-data parameter yang diujikan pada simulasi yang telah dilakukan. Parameter yang digunakan yaitu *delay*, *throughput* dan PDR.

4.2.1 Hasil Pengukuran *Delay*

Berikut ini adalah hasil pengukuran *delay* pada protokol *routing random walk* dengan skenario pengujian yang menggunakan TCP. Waktu simulasi 5 detik dan 15 detik digunakan sebagai skenario I dan II. Pengukuran semua *node* tujuan (1 s/d 29) dengan *node* 0 sebagai *node* asal (*source*), dan hasilnya dapat dilihat pada Tabel 4.1.

Tabel 4. 1 Rata-Rata *Delay* Protokol *Random Walk* Dengan TCP

No	Tujuan	Delay (s)		Rata-rata
		Pengujian I (5 detik)	Pengujian II (15 detik)	
1	*Node*-1	0.1565	0.1611	0.1588
2	*Node*-2	0.1562	0.1611	0.15865
3	*Node*-3	0.1543	0.1613	0.1578
4	*Node*-4	0.3051	0.3193	0.3122
5	*Node*-5	0.1062	0.258	0.1821
6	*Node*-6	0.3039	0.319	0.31145
7	*Node*-7	0.2926	0.3186	0.3056
8	*Node*-8	0.1537	0.161	0.15735
9	*Node*-9	0.1551	0.1607	0.1579
10	*Node*-10	0.3049	0.3193	0.3121
11	*Node*-11	0.4436	0.4816	0.4626
12	*Node*-12	0.4282	0.471	0.4496
13	*Node*-13	0.4049	0.4689	0.4369
14	*Node*-14	0.4251	0.4693	0.4472
15	*Node*-15	0.4169	0.4724	0.44465
16	*Node*-16	0.657	1.1807	0.91885
17	*Node*-17	0.5018	0.7107	0.60625
18	*Node*-18	0.5136	0.7049	0.60925
19	*Node*-19	0.5226	0.6973	0.60995
20	*Node*-20	0.3003	0.3262	0.31325
21	*Node*-21	0.4555	0.3544	0.40495
22	*Node*-22	0.5677	0.6942	0.63095
23	*Node*-23	0.4457	0.5911	0.5184
24	*Node*-24	0.5169	0.404	0.46045
25	*Node*-25	0.2979	0.3601	0.329
26	*Node*-26	0.4597	0.3879	0.4238
27	*Node*-27	0.2061	0.2938	0.24995
28	*Node*-28	0.6193	0.4923	0.5558
29	*Node*-29	0.6868	0.8239	0.75535

Sementara berikut ini adalah hasil pengukuran *delay* pada protokol *routing Random Walk* dengan menggunakan UDP.

Tabel 4. 2 Rata-Rata *Delay* Protokol *Random Walk* Dengan UDP

No	Tujuan	Delay (s)		Rata-rata
		Pengujian I (5 detik)	Pengujian II (15 detik)	
1	*Node* -1	0.378	0.597	0.4875
2	*Node* -2	0.378	0.597	0.4875
3	*Node* -3	0.368	0.593	0.4805
4	*Node* -4	1.15	1.356	1.253
5	*Node* -5	1.245	1.577	1.411
6	*Node* -6	1.182	1.452	1.317
7	*Node* -7	1.138	1.351	1.2445
8	*Node* -8	0.423	0.689	0.556
9	*Node* -9	0.38	0.599	0.4895
10	*Node* -10	0.924	1.121	1.0225
11	*Node* -11	1.055	1.777	1.416
12	*Node* -12	1.084	1.658	1.371
13	*Node* -13	1.617	2.017	1.817
14	*Node* -14	1.492	2.137	1.8145
15	*Node* -15	1.372	2.233	1.8025
16	*Node* -16	2.092	3.595	2.8435
17	*Node* -17	1.901	2.82	2.3605
18	*Node* -18	1.933	2.567	2.25
19	*Node* -19	2.032	3.724	2.878
20	*Node* -20	1.543	2.235	1.889
21	*Node* -21	1.674	2.346	2.01
22	*Node* -22	2.104	3.308	2.706
23	*Node* -23	1.899	2.891	2.395
24	*Node* -24	1.985	2.992	2.4885
25	*Node* -25	1.455	3.588	2.5215
26	*Node* -26	1.731	2.782	2.2565
27	*Node* -27	1.353	2.48	1.9165
28	*Node* -28	1.45	6.207	3.8285
29	*Node* -29	1.641	7.116	4.3785

4.2.2 Hasil Pengukuran *Throughput*

Pengukuran *throughput* pada protokol *routing random walk* dengan protokol transport TCP ke *node* tujuan. Percobaan dilakukan dengan 2 skenario simulasi yaitu 5 detik dan 15 detik. Pengukuran dilakukan sebanyak 2 kali, ke semua *node* tujuan (1 s/d 29) dan hasilnya dapat dilihat pada Tabel 4.3.

Tabel 4. 3 Rata-Rata *Throughput* Protokol *Random Walk* TCP

No	Tujuan	*Throughput* (kbps)		Rata-rata
		Pengujian I (5 detik)	Pengujian II (15 detik)	
1	*Node*-1	119.015	120.633	119.824
2	*Node*-2	119.775	120.633	120.204
3	*Node*-3	116.072	119.447	117.7595
4	*Node*-4	119.749	120.63	120.1895
5	*Node*-5	51.449	98.16	74.8045
6	*Node*-6	120.08	120.724	120.402
7	*Node*-7	117.056	121.02	119.038
8	*Node*-8	115.304	120.047	117.6755
9	*Node*-9	116.343	119.689	118.016
10	*Node*-10	120.375	121.064	120.7195
11	*Node*-11	119.886	118.799	119.3425
12	*Node*-12	119.694	121.042	120.368
13	*Node*-13	116.432	121.062	118.747
14	*Node*-14	117.809	120.535	119.172
15	*Node*-15	120.444	120.59	120.517
16	*Node*-16	124.312	94.823	109.5675
17	*Node*-17	86.444	104.315	95.3795
18	*Node*-18	112.536	108.204	110.37
19	*Node*-19	115.942	123.094	119.518
20	*Node*-20	122.202	122.612	122.407
21	*Node*-21	128.56	128.178	128.369
22	*Node*-22	113.848	123.015	118.4315
23	*Node*-23	68.589	103.692	86.1405
24	*Node*-24	134.7	130.884	132.792
25	*Node*-25	122.04	126.496	124.268
26	*Node*-26	123.912	125.82	124.866
27	*Node*-27	122.702	128.833	125.7675
28	*Node*-28	139.555	138.46	139.0075
29	*Node*-29	136.374	114.195	125.2845

Berikut ini adalah pengukuran *throughput* pada protokol *routing Random Walk* dengan UDP. Hasilnya dapat dilihat pada tabel dibawah ini.

Tabel 4. 4 Rata-Rata *Throughput* Protokol *Random Walk* UDP

No	Tujuan	*Throughput* (kbps)		Rata-rata
		Pengujian I (5 detik)	Pengujian II (15 detik)	
1	*Node* -1	103.12	103.783	103.4515
2	*Node* -2	103.12	103.783	103.4515
3	*Node* -3	102.98	102.98	102.98
4	*Node* -4	47.073	48.46	47.7665
5	*Node* -5	56.234	58.902	57.568
6	*Node* -6	52.778	51.335	52.0565
7	*Node* -7	48.218	48.531	48.3745
8	*Node* -8	68.024	69.704	68.864
9	*Node* -9	103.12	103.783	103.4515
10	*Node* -10	82.356	81.927	82.1415
11	*Node* -11	76.778	77.779	77.2785
12	*Node* -12	43.35	45.451	44.4005
13	*Node* -13	29.715	30.646	30.1805
14	*Node* -14	68.778	70.091	69.4345
15	*Node* -15	29.991	31.113	30.552
16	*Node* -16	31.316	38.021	34.6685
17	*Node* -17	35.333	34.456	34.8945
18	*Node* -18	42.762	43.982	43.372
19	*Node* -19	27.285	26.933	27.109
20	*Node* -20	34.436	36.656	35.546
21	*Node* -21	29.935	30.871	30.403
22	*Node* -22	25.856	25.669	25.7625
23	*Node* -23	22.256	23.334	22.795
24	*Node* -24	37.269	40.321	38.795
25	*Node* -25	39.978	43.617	41.7975
26	*Node* -26	42.123	40.032	41.0775
27	*Node* -27	40.086	38.877	39.4815
28	*Node* -28	56.004	46.181	51.0925
29	*Node* -29	34.554	30.059	32.3065

4.2.3 Hasil Pengukuran PDR

Hasil *packet delivery ratio* (PDR) pada protokol *routing random walk* diuji pada waktu simulasi 5 detik dan 15 detik dengan protokol transport TCP. Paket data yang dikirim ke semua *node* tujuan (1 s/d 29) selanjutnya diukur nilai PDR-nya dan hasilnya dapat dilihat pada Tabel 4.5.

Tabel 4. 5 Rata-Rata PDR Protokol *Random Walk* TCP

No	Tujuan	Packet Delivery Ratio (%)		Rata-rata
		Pengujian I (5 detik)	Pengujian II (15 detik)	
1	*Node*-1	99.2558	99.7556	99.5057
2	*Node*-2	99.2549	99.7556	99.50525
3	*Node*-3	99.2662	99.7567	99.51145
4	*Node*-4	97.5219	98.9943	98.2581
5	*Node*-5	95.5462	98.7904	97.1683
6	*Node*-6	98.0985	99.1899	98.6442
7	*Node*-7	97.2586	98.3721	97.81535
8	*Node*-8	99.2285	99.7567	99.4926
9	*Node*-9	99.2644	99.7571	99.51075
10	*Node*-10	98.0847	99.2688	98.67675
11	*Node*-11	97.6587	98.6556	98.15715
12	*Node*-12	98.4184	99.3807	98.89955
13	*Node*-13	96.5571	97.93	97.24355
14	*Node*-14	97.7335	98.6425	98.188
15	*Node*-15	97.9656	99.1134	98.5395
16	*Node*-16	84.3779	85.6255	85.0017
17	*Node*-17	92.327	92.7033	92.51515
18	*Node*-18	92.4652	93.7196	93.0924
19	*Node*-19	91.8227	91.1795	91.5011
20	*Node*-20	90.7226	88.4706	89.5966
21	*Node*-21	88.9502	89.5392	89.2447
22	*Node*-22	89.6005	90.9031	90.2518
23	*Node*-23	93.4626	97.4823	95.47245
24	*Node*-24	86.2059	88.5255	87.3657
25	*Node*-25	90.2006	88.7467	89.47365
26	*Node*-26	88.4593	88.7823	88.6208
27	*Node*-27	89.9602	89.11	89.5351
28	*Node*-28	87.5923	88.0007	87.7965
29	*Node*-29	88.4781	88.5581	88.5181

Setelah melakukan pengukuran PDR dengan TCP pada protokol *routing Random Walk*, selanjutnya dilakukan pengukuran PDR dengan menggunakan UDP. Dan hasilnya dapat dilihat pada tabel dibawah ini.

Tabel 4. 6 Rata-Rata PDR Protokol *Random Walk* UDP

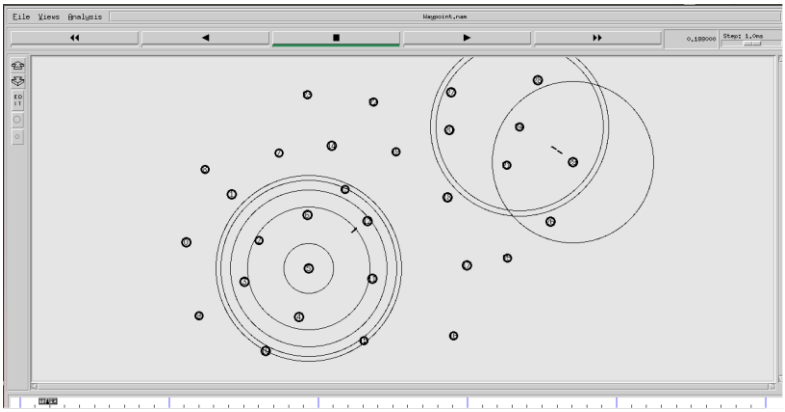
No	Tujuan	Packet Delivery Ratio (%)		Rata-rata
		Pengujian I (5 detik)	Pengujian II (15 detik)	
1	*Node* -1	95.642	95.911	95.7765
2	*Node* -2	95.643	95.912	95.7775
3	*Node* -3	94.457	94.498	94.4775
4	*Node* -4	81.771	82.637	82.204
5	*Node* -5	80.732	81.881	81.3065
6	*Node* -6	82.823	83.735	83.279
7	*Node* -7	80.447	82.45	81.4485
8	*Node* -8	85.889	86.653	86.271
9	*Node* -9	95.642	95.911	95.7765
10	*Node* -10	89.902	90.023	89.9625
11	*Node* -11	83.324	82.435	82.8795
12	*Node* -12	80.818	81.912	81.365
13	*Node* -13	78.236	78.894	78.565
14	*Node* -14	77.792	78.872	78.332
15	*Node* -15	73.956	74.321	74.1385
16	*Node* -16	72.702	72.834	72.768
17	*Node* -17	73.717	72.771	73.244
18	*Node* -18	74.567	76.292	75.4295
19	*Node* -19	75.866	76.514	76.19
20	*Node* -20	79.892	80.391	80.1415
21	*Node* -21	78.223	77.652	77.9375
22	*Node* -22	75.706	76.733	76.2195
23	*Node* -23	72.134	73.142	72.638
24	*Node* -24	71.521	72.017	71.769
25	*Node* -25	70.54	71.844	71.192
26	*Node* -26	70.587	71.234	70.9105
27	*Node* -27	72.101	73.346	72.7235
28	*Node* -28	71.032	73.533	72.2825
29	*Node* -29	72.258	72.625	72.4415

4.3 Hasil Pengukuran Protokol Pembanding

Sebagai pembanding kinerja protokol *random walk*, dilakukan pengujian menggunakan 2 buah protokol yang mewakili mekanisme pemilihan jalur secara statis (protokol *waypoint*) dan protokol reaktif yang biasa digunakan dalam jaringan-jaringan yang bersifat *adhoc* (protokol AODV). Karena hanya sebagai pembanding maka hanya dilakukan pengujian terhadap beberapa *node* tujuan saja yaitu *node* 9, *node* 16, *node* 22, *node* 28 dan *node* 29.

4.3.1 Hasil Pengukuran Protokol *Waypoint*

Dengan menggunakan jumlah dan posisi koordinat *node* yang sama dengan *random walk*, pengukuran dilakukan kembali namun protokolnya diganti dengan protokol *waypoint* dan diperoleh contoh hasil visualnya bisa dilihat pada Gambar 4.4 berikut.



Gambar 4. 4 Contoh *Running* Protokol *Waypoint*

a. Hasil Pengukuran I Protokol *Waypoint*

Pada bagian ini merupakan hasil pengukuran parameter *delay*, *throughput* dan PDR pada protokol *routing waypoint* dengan skenario pengujian I yaitu waktu simulasi 5 detik dengan protokol transport TCP dan UDP. *Node* tujuan yang dipilih sesuai dengan awal sub bab 4.3. Seluruh pengukuran dilakukan dengan *node* 0 sebagai *node* asal (*source*). Hasil dapat dilihat pada Tabel 4.4 untuk protokol transport TCP.

Tabel 4. 7 Performansi Pengujian I (5 detik) *Waypoint* TCP

No	Tujuan	Jarak (Hop)	Delay	*Throughput*	PDR
1	*Node*-9	1	0.1785	20.371	92.5831
2	*Node*-16	4	0.2834	24.172	89.3734
3	*Node*-22	4	0.2834	24.172	89.3734
4	*Node*-28	7	0.4959	42.302	90.5812
5	*Node*-29	6	0.4251	36.259	90.7464

Sementara hasil dari penggunaan protokol transport UDP dapat dilihat pada tabel berikut ini. Hasil dari penggunaan UDP ini nantinya juga akan digunakan sebagai perbandingan manakah yang memberikan hasil yang lebih baik.

Tabel 4. 8 Performansi Pengujian I (5 detik) *Waypoint* UDP

No	Tujuan	Jarak (Hop)	Delay	*Throughput*	PDR
1	*Node*-9	1	0.2785	22.375	91.8315
2	*Node*-16	4	0.4832	26.183	78.7343
3	*Node*-22	4	0.4832	26.183	78.3347
4	*Node*-28	7	0.9598	45.342	59.5833
5	*Node*-29	6	0.7326	39.279	61.4642

b. Hasil Pengukuran II Protokol *Waypoint*

Masih sama dengan bagian sebelumnya, bagian ini merupakan hasil pengukuran parameter *delay, throughput* dan PDR pada protokol *routing waypoint* dengan TCP. Namun skenario yang digunakan berbeda yaitu waktu simulasi 15 detik lebih lama 3 kali lipat dari waktu simulasi pengukuran sebelumnya yang menggunakan waktu simulasi 5 detik. Pengukuran ini digunakan sebagai skenario pengujian II yang nantinya digunakan sebagai komparasi terhadap skenario pengujian I dan hasil dari penggunaan protokol transport TCP dapat dilihat pada Tabel 4.5 dibawah ini.

Tabel 4. 9 Performansi Pengujian II (15 detik) *Waypoint* TCP

No	Tujuan	Jarak (Hop)	Delay	*Throughput*	PDR
1	*Node*-9	1	0.6987	34.736	91.3579
2	*Node*-16	4	0.9651	42.714	90.9378
3	*Node*-22	4	0.9651	42.714	90.9378
4	*Node*-28	7	1.6889	74.750	88.1356
5	*Node*-29	6	1.4477	64.072	88.3064

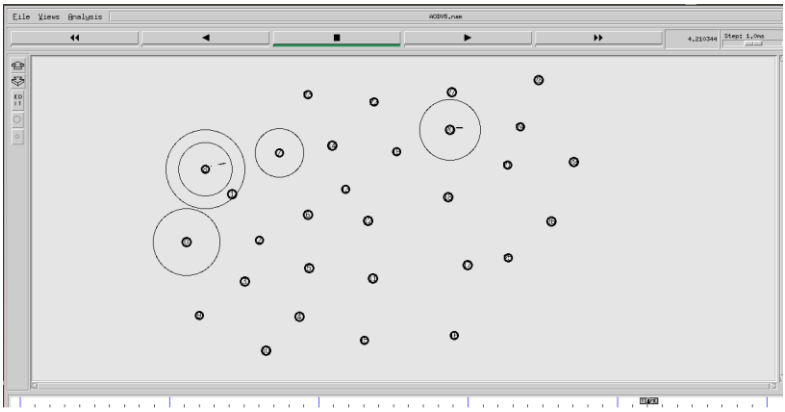
Sementara itu, pengujian dengan waktu simulasi 15 detik juga dilakukan pada protokol *routing waypoint* dengan penggunaan protokol transport UDP. Hasil dari penggunaan protokol transport UDP pada protokol *routing waypoint* dapat dilihat pada tabel dibawah ini.

Tabel 4. 10 Performansi Pengujian II (15 detik) *Waypoint* UDP

No	Tujuan	Jarak (Hop)	Delay	*Throughput*	PDR
1	*Node*-9	1	0.7231	35.732	90.5793
2	*Node*-16	4	1.1351	44.742	75.3789
3	*Node*-22	4	1.1351	44.742	75.3789
4	*Node*-28	7	1.9759	76.758	55.3561
5	*Node*-29	6	1.8476	68.092	58.0643

4.3.2 Hasil Pengukuran Protokol AODV

Pengukuran pada parameter protokol AODV juga dilakukan pada bagian ini, koordinat distribusi *node* pengujian yang digunakan juga masih mengambil dari Tabel 3.2, namun protokol *routing*-nya diganti dengan protokol *routing* AODV dan *node* tujuan hanya *node-9, node-16, node-22, node-28* dan *node-29* dipilih karena hanya sebagai pembanding yang mewakili *node-node* dengan jarak tertentu dimana jarak yang dimaksud adalah jarak terdekat, menengah dan terjauh dari *node* asal. Contoh hasil visual dari simulasi dengan protokol ini dapat dilihat pada gambar 4.5 berikut :



Gambar 4. 5 Contoh *Routing* Protokol AODV

a. Hasil Pengukuran I Protokol AODV

Pengukuran I protokol AODV dilakukan dengan skenario pengujian I. Berikut ini adalah hasil pengukuran parameter *delay*, *throughput* dan PDR pada protokol *routing* AODV menggunakan TCP dengan skenario pengujian I. Seluruh pengukuran dilakukan dengan *node* tujuan seperti yang sudah dijelaskan pada awal sub bab 4.3. Hasil dari penngukurannya dapat dilihat pada tabel dibawah ini.

Tabel 4. 11 Performansi Pengujian I (5 detik) AODV TCP

No	Tujuan	Koordinat	Delay	*Throughput*	PDR
1	*Node*-9	(331,118)	0.1540	120.469	98.1548
2	*Node*-16	(967,67)	0.2121	71.599	89.1692
3	*Node*-22	(767,651)	0.1087	119.020	84.3259
4	*Node*-28	(1178,705)	0.1967	59.082	86.7665
5	*Node*-29	(1266,500)	0.1699	122.385	89.1203

Sedangkan pengukuran terhadap protokol *routing* AODV dengan menggunakan protokol transport UDP juga dilakukan. Hasil dari penggunaan UDP tersebut dapat dilihat pada tabel dibawah ini dan nantinya juga digunakan sebagai komparasi.

Tabel 4. 12 Performansi Pengujian I (5 detik) AODV UDP

No	Tujuan	Koordinat	Delay	*Throughput*	PDR
1	*Node*-9	(331,118)	0.3819	103.707	94.9466
2	*Node*-16	(967,67)	1.982	27.647	76.4163
3	*Node*-22	(767,651)	0.6172	15.476	58.9823
4	*Node*-28	(1178,705)	0.6906	19.004	56.3337
5	*Node*-29	(1266,500)	1.624	21.222	57.4175

b. Hasil Pengukuran II Protokol AODV

Berikut ini adalah hasil pengukuran parameter *delay, throughput* dan PDR pada protokol *routing* AODV menggunakan TCP dengan skenario pengujian II yaitu waktu simulasi 15 detik. Seluruh pengukuran dilakukan dengan *node* 0 sebagai *node* awal (*source*).

Tabel 4. 13 Performansi Pengujian II (15 detik) AODV TCP

No	Tujuan	Koordinat	Delay	*Throughput*	PDR
1	*Node*-9	(331,118)	0.1584	120.681	99.9308
2	*Node*-16	(967,67)	0.1958	100.218	86.8269
3	*Node*-22	(767,651)	0.1450	118.068	84.7283
4	*Node*-28	(1178,705)	0.1843	53.085	87.6415
5	*Node*-29	(1266,500)	0.1704	127.412	88.0588

Sementara hasil dari protokol *routing* AODV menggunakan UDP dapat dilihat pada tabel dibawah ini.

Tabel 4. 14 Performansi Pengujian II (15 detik) AODV UDP

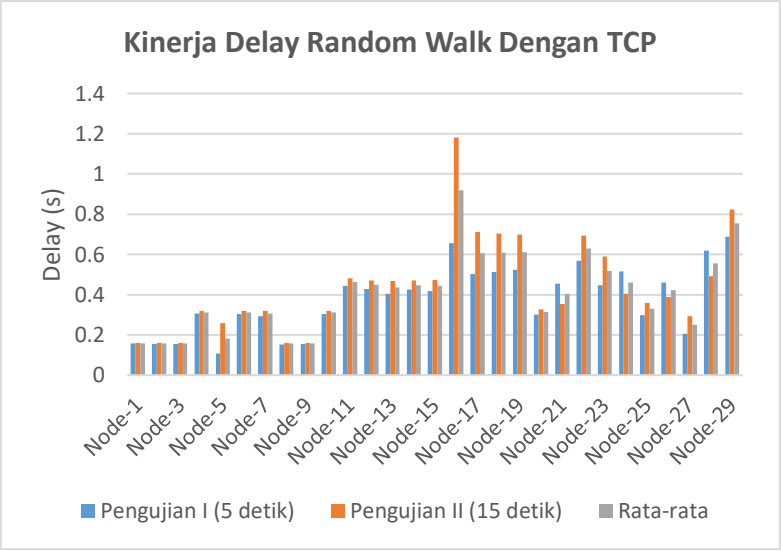
No	Tujuan	Koordinat	Delay	*Throughput*	PDR
1	*Node*-9	(331,118)	0.5893	103.764	95.472
2	*Node*-16	(967,67)	3.364	26.252	76.841
3	*Node*-22	(767,651)	3.304	27.177	60,523
4	*Node*-28	(1178,705)	0.936	35.975	65.732
5	*Node*-29	(1266,500)	1.363	23.588	67.651

4.4 Pembahasan Hasil Pengujian

Pada bagian ini akan dilakukan pembahasan hasil pengujian kinerja protokol *random walk*, protokol pembanding dan komparasi kinerja antar protokol tersebut. Ada beberapa hal yang perlu dibahas terkait dengan pengaruh lokasi *node* tujuan dan lama waktu simulasi, yang akan mempengaruhi besarnya parameter *delay*, *throughput* dan PDR.

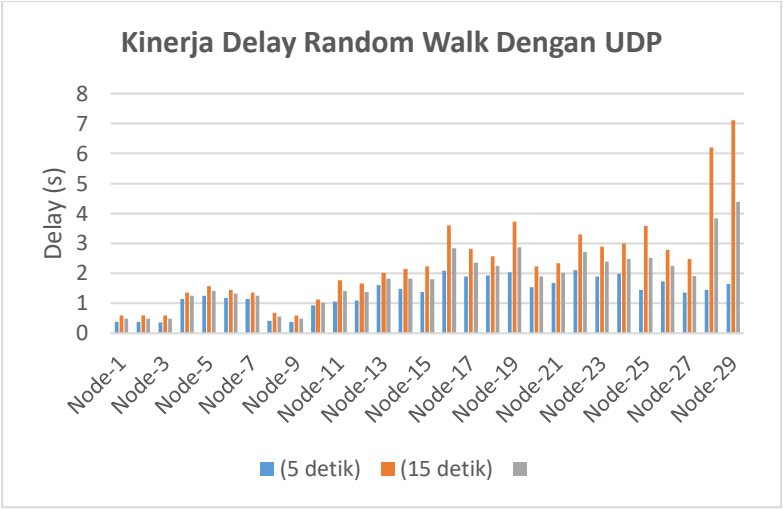
4.4.1 Kinerja Protokol *Random walk*

Kinerja dari protokol *random walk* terdiri dari 3 parameter yang telah dilakukan pengukuran. Yang pertama adalah pada parameter *delay*, kemudian parameter *throughput* dan yang terkahir parameter *packet delivery ratio* (PDR).



Gambar 4. 6 Kinerja *Delay Random Walk* dengan TCP

Dari hasil pengukuran protokol *random walk* dengan TCP maka diperoleh hasil yaitu dengan posisi *node* tujuan semakin jauh dalam hal ini berupa jumlah *hop* lebih banyak maka nilai *delay* yang dihasilkan cenderung lebih besar. Begitupun dengan penggunaan UDP pada protokol *random walk*, dimana dapat dilihat dari histogram dibawah ini.

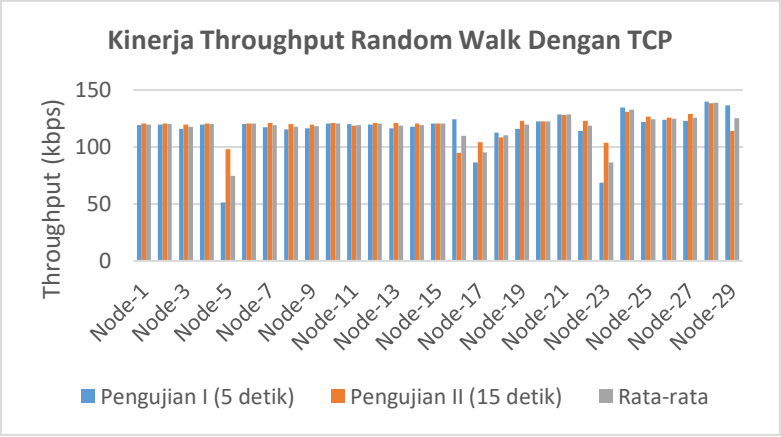


Gambar 4. 7 Kinerja *Delay Random Walk* dengan UDP

Ketika waktu simulasi ditingkatkan 3 kali lebih lama dari 5 detik menjadi 15 detik nilai *delay* menjadi sedikit lebih besar sebesar 6%, kecuali pada *node* lokasi tujuan yang jauh. Hal ini bisa diprediksi karena ketika terjadi kepadatan di lokasi yang jauh, rute pengiriman data belum tentu berubah karena adanya kemungkinan saat siklus berikutnya kondisi jalur yang dilalui sudah kembali normal *(*sudah *recovery)*. Sehingga pengaruh lama simulasi nyaris tidak berpengaruh kecuali pada *node-node* tertentu. Lalu bila dibandingkan dapat dilihat bahwa penggunaan UDP yang bersifat *connection-less* membuat *delay* yang dihasilkan lebih besar dari penggunaan TCP secara signifikan pada tiap *node* tujuan yang diujikan.

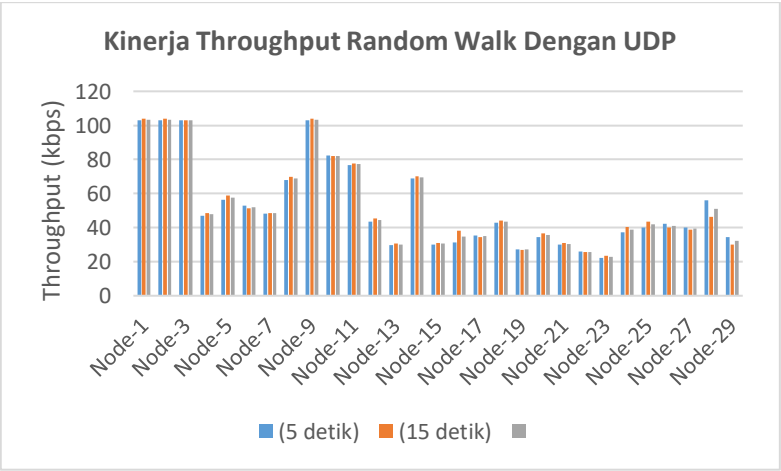
Pembahasan kinerja berikutnya adalah mengenai kinerja *throughput* pada protokol *routing random walk* dengan menggunakan TCP. Kenaikan dan penurunan dari nilai *throughput* dapat dilihat pada Gambar 4.8. Dan Sebagaimana yang terlihat pada Gambar 4.8, dimana gambar tersebut merupakan hasil pengukuran dari *throughput* yang dilakukan dengan menggunakan TCP dalam 2 skenario pengujian yang berbeda terlihat bahwa jarak *node source* terhadap *node destination* tidak berpengaruh

pada hasil pengujian *throughput.* Namun dengan meningkatkan waktu simulasi dari 5 detik menjadi 15 detik, nilai *throughput* mengalami sedikit peningkatan yaitu kurang lebih 3%.



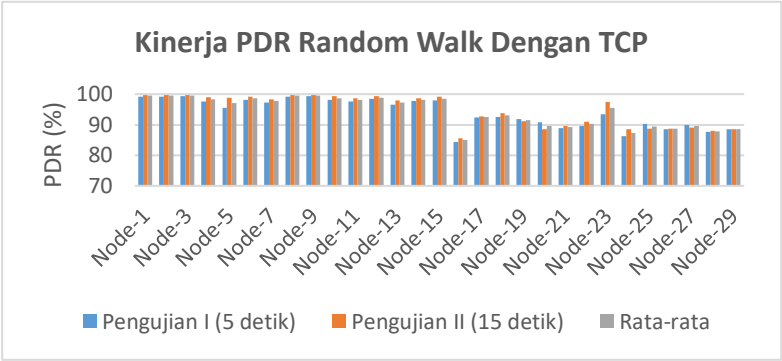
Gambar 4. 8 Kinerja *Throughput Random Walk* dengan TCP

Sementara kinerja *throughput* dengan menggunakan UDP dapat dilihat pada Gambar 4.9 dibawah ini.



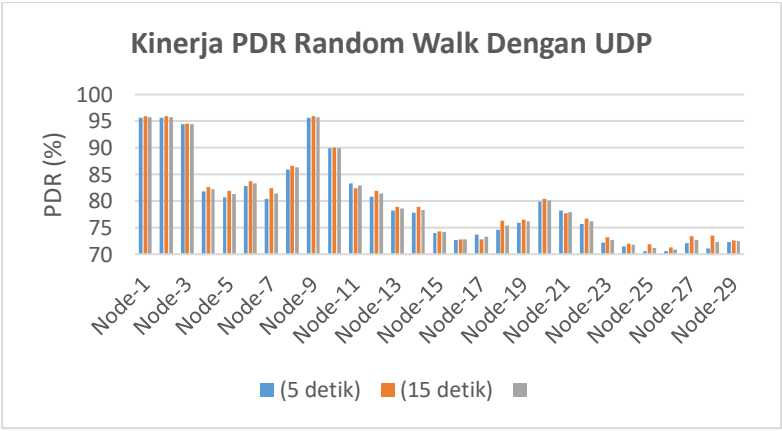
Gambar 4. 9 Kinerja *Throughput Random Walk* dengan UDP

Pembahasan kinerja selanjutnya yaitu parameter *packet delivery ratio* (PDR). Yang terjadi pada PDR antara skenario pengujian I (5 detik) dan pengujian II (15 detik) menggunakan TCP tidak mengalami perubahan yang signifikan (rata-rata 94.38%). Namun nilai PDR sebagian besar mengalami penurunan pada *node* tujuan dengan jumlah *hop* yang lebih banyak (jarak menjauh dari *node* asal).



Gambar 4. 10 Kinerja PDR *Random Walk* dengan TCP

Sementara itu dengan penggunaan UDP dapat dilihat pada gambar dibawah ini.



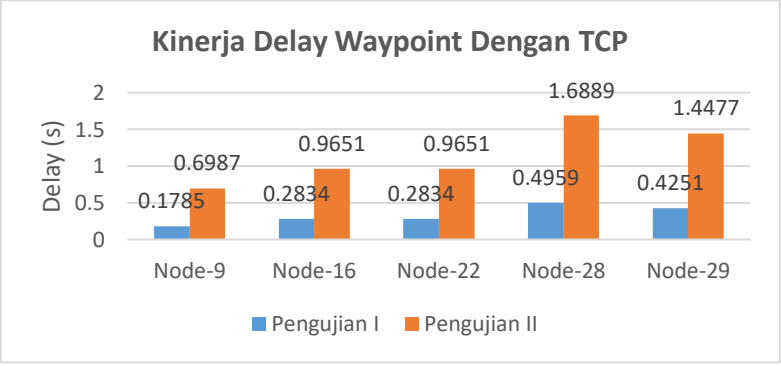
Gambar 4. 11 Kinerja PDR *Random Walk* dengan UDP

Dengan penggunaan UDP masih sama seperti dengan TCP dimana baik skenario pengujian I maupun pengujian II pada tiap *node* yang diuji tidak mengalami perubahan siginifikan walaupun dilakukan penambahan waktu simulasi dari 5 detik menjadi 15 detik. Sementara bila dibandingkan dengan TCP, penggunaan UDP membuat PDR mengalami penurunan yag cukup signifikan.

4.4.2 Kinerja Protokol Pembanding

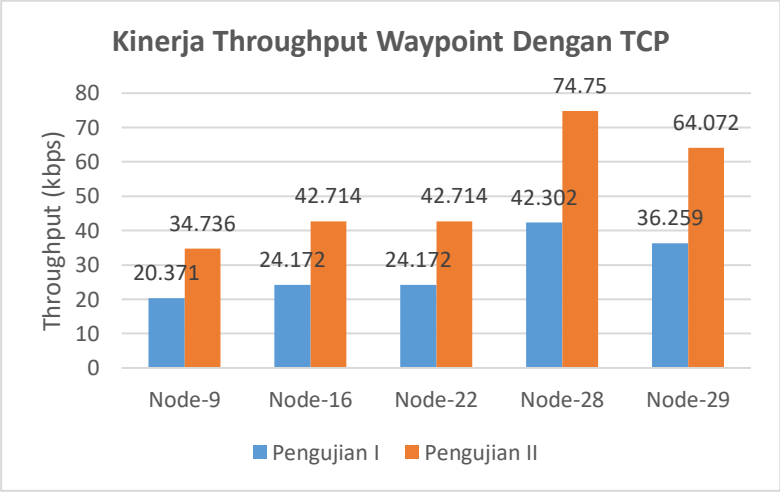
a. Kinerja Protokol *Waypoint*

Dari pengukuran yang sudah dilakukan pada protokol *waypoint*, maka dapat dilakukan pengambilan nilai parameter pengukuran antara pengujian I dan pengujian II sehingga dapat menghasilkan perbandingan antar pengujian dengan masing-masing parameter yang sudah diukur. Untuk pengambilan perbandingan yang pertama adalah perbandingan nilai *delay* dari protokol *waypoint* dan nantinya dapat dilihat bagaimana hasil pengujian antara pengujian I dengan pengujian II yang telah dilakukan. Dari *delay* antara 2 pengujian tersebut dapat dilihat bahwa semakin lama waktu simulasi maka semakin tinggi pula nilai *delay* yang dihasilkan. Sebagai contoh pada *node-28* dimana semakin lama waktu simulasi akhirnya berdampak pada kenaikan nilai *delay* yang signifikan dari 0.4959 detik pada waktu simulasi 5 detik menjadi 1.6889 detik pada waktu simulasi 15 detik. Nilai *delay* tersebut juga dipengaruhi oleh lokasi *node* yang dituju terutama pada *node* tujuan dengan *hop* yang lebih banyak. Perbandingan tersebut dapat dilihat pada Gambar 4.9 berikut ini.



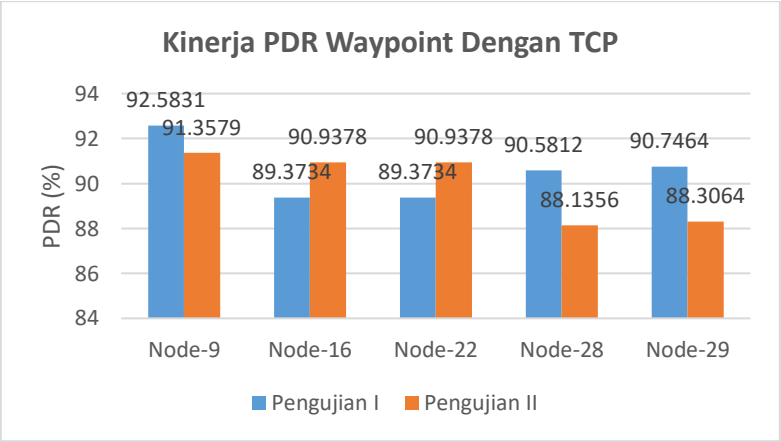
Gambar 4. 12 Kinerja *Delay Waypoint* dengan TCP

Pengambilan perbandingan yang kedua adalah perbandingan nilai *throughput* antara pengujian 1 dan pengujian II. Hasil perbandingan tersebut digunakan untuk melihat bagaimana naik atau turunnya nilai *throughput* pada *node* yang sama dengan skenario waktu pengujian yang berbeda. Dari pengukuran *throughput* tersebut dapat dilihat bahwa dengan bertambahnya waktu simulasi, bertambah pula nilai *throughput* yang dihasilkan. Nilai *throughput* juga berbanding lurus dengan banyaknya *hop* yang dilalui saat pengiriman data terutama pada *node* dengan lokasi yang jauh. Sebagai contoh antara *node* terdekat (*node-9*) dan *node* terjauh (*node-28*) memiliki nilai *throughput* sebesar 20.371 kbps dan 42.302 kbps pada waktu simulasi 5 detik. Sementara pada waktu simulasi 15 detik memiliki nilai sebesar 34.736 kbps dan 74.75 kbps. Hasil perbandingan dapat dilihat pada Gambar 4.10.



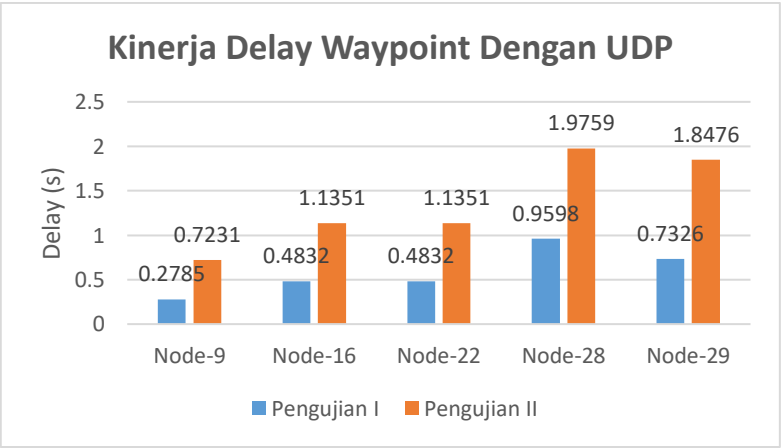
Gambar 4. 13 Kinerja *Throughput Waypoint* dengan TCP

Dari pengukuran tersebut juga dilakukan pengambilan perbandingan nilai PDR antara pengujian I dan pengujian II. Nilai PDR relatif sama pada setiap pengujian, baik pada pengujian I maupun pengujian II. Sehingga nilai PDR tidak terlalu dipengaruhi oleh lamanya waktu simulasi maupun banyaknya jumlah *hop* yang dilalui. Perbandingan nilai PDR dari kedua pengujian dapat dilihat pada Gambar 4.7 berikut.



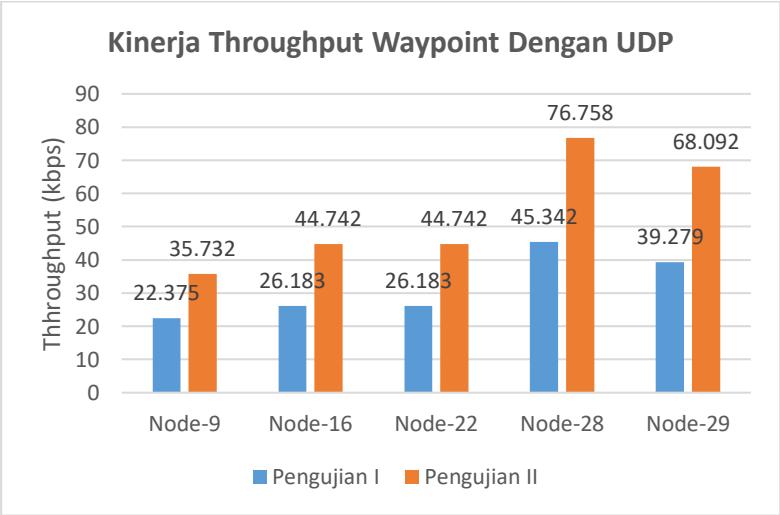
Gambar 4. 14 Kinerja PDR *Waypoint* dengan TCP

Selain pengukuran dengan penggunaan TCP, pengukuran juga dilakukan dengan penggunaan UDP. Masih sama dengan hasil pada penggunaan TCP, dengan bertambahnya waktu simulasi maka bertambah pula nilai *delay* yang dihasilkan. Hasil dapat dilihat pada Gambar 4.12. Selain itu nilai *delay* juga mengalami peningkatan bila dibandingkan dengan TCP.



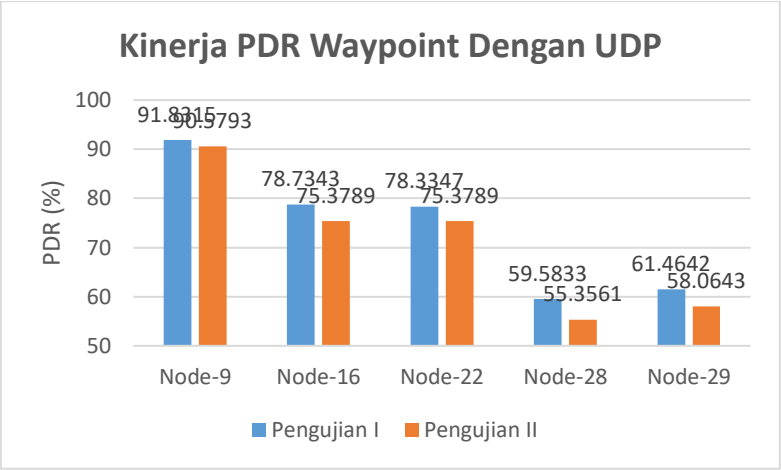
Gambar 4. 15 Kinerja *Delay Waypoint* dengan UDP

Dari pengukuran dengan penggunaan UDP, juga dilakukan pengukuran pada parameter *throughhput.* Hasil yang dihasilkan juga masih sama seperti penggunaan TCP dimana dengan bertambahnya waktu simulasi, nilai *throughput* juga mengalami penigkatan. Sebagai contoh pada *node-28* dimana *node* tersebut merupakan *node* dengan lokasi terjauh dari *node* asal yang memiliki nilai *throughput* sebesar 45.342 kbps pada waktu simulasi 5 detik dan meningkat menjadi 76.758 kbps pada saat waktu simulasi 15 detik. Hasilnya dapat dilihat pada Gambar 4.13. Dan bila dibandingkan dengan TCP, maka nilai tersebut juga mengalami perbedaan dimana TCP memiliki nilai yang lebih rendah.



Gambar 4. 16 Kinerja *Throughput Waypoint* dengan UDP

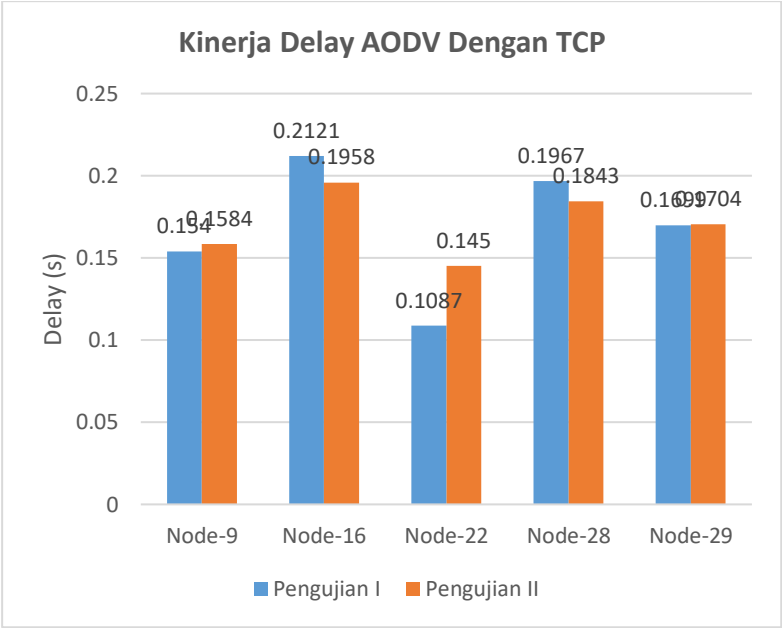
Dari pengukuran tersebut juga dilakukan pengambilan perbandingan nilai PDR antara pengujian I dan pengujian II. Nilai PDR relatif mengalami penurunan pada setiap *node* pengujian, baik pada pengujian I maupun pengujian II. Sehingga nilai PDR sangat dipengaruhi oleh lamanya waktu simulasi. Banyaknya jumlah *hop* yang dilalui juga sangat mempengaruhi hasil dari PDR. Sebagai contoh pada *node* dengan jarak terdekat dan terjauh yaitu *node-9* dan *node-28*. Perbandingan nilai PDR dari kedua pengujian dapat dilihat pada Gambar berikut.



Gambar 4. 17 Kinerja PDR *Waypoint* dengan UDP

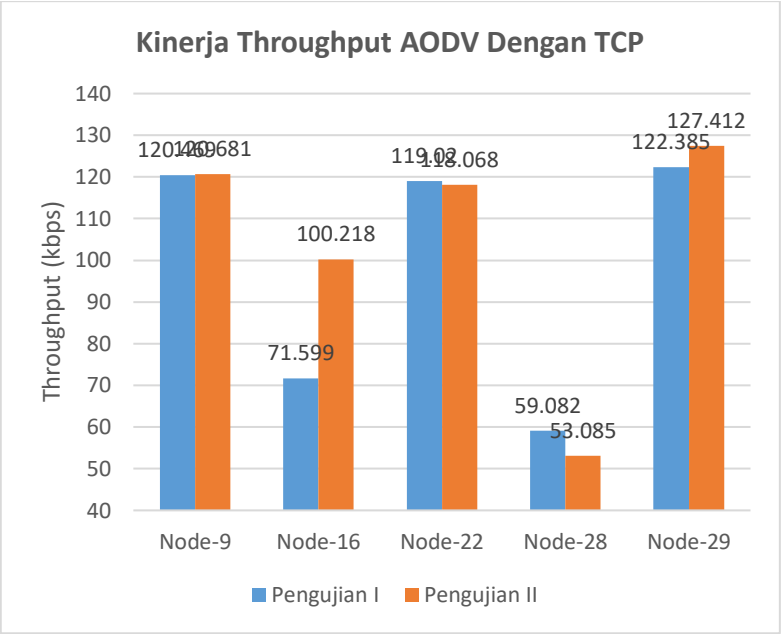
b. Kinerja Protokol AODV

Kinerja dari protokol *routing* AODVjuga terdiri dari 3 parameter pengukuran sama seperti pada pengukuran pada protokol *random walk* dan protokol *waypoint.* Dari pengukuran yang sudah dilakukan maka terdapat 2 skenario yang berbeda waktu simulasinya. Pengukuran ini juga dilakukan dengan penggunaan protokol transport TCP dan UDP. Oleh sebab itu dilakukan pengambilan perbandingan dari tabel hasil pengukuran untuk mengetahui bagaimana kinerjanya dalam 2 waktu simulasi berbeda. Hal pertama yang dilakukan adalah pengambilan perbandingan nilai *delay* dari protokol AODV dengan 2 skenario pengujian tersebut menggunakan protokol transport TCP. Dari perbandingan tersebut kinerja *delay* protokol AODV relatif sama pada 2 waktu simulasi yang berbeda dan tidak terjadi perubahan yang signifikan baik pada *node* terdekat (*node-9*) maupun *node* terjauh (*node-28*). Sebagai contoh yaitu pada *node-9* nilai *delay* pada waktu simulasi 5 detik sebesar 0.154 detik sedangkan pada waktu simulasi 15 detik sebesar 0.1584 detik. Hal ini membuktikan bahwa lama waktu simulasi tidak terlalu berpengaruh pada nilai *delay* protokol *routing* AODV. Hasil perbandingan antara 2 skenario pengujian dapat dilihat pada Gambar 4.12 berikut ini.



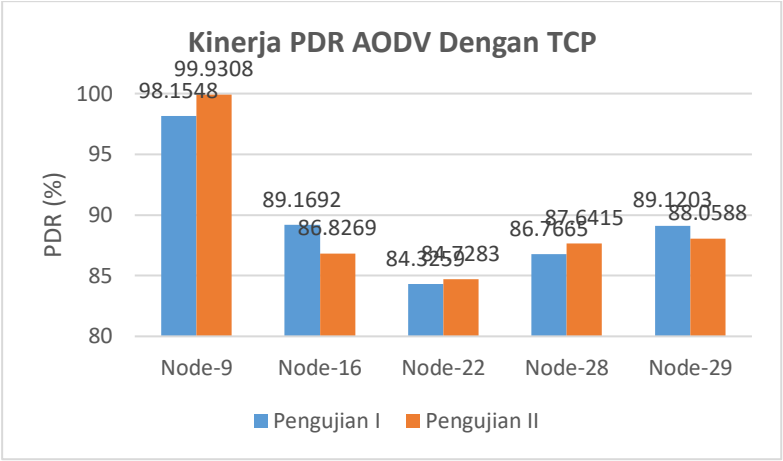
Gambar 4. 18 Kinerja *Delay* AODV dengan TCP

Dengan pengukuran yang sudah dilakukan, maka parameter pengukuran berikutnya yang diukur adalah nilai *throughput.* Dari perbandingan dapat dilihat bagaimana naik turunnya nilai *throughput* terhadap perbedaan waktu simulasi pada *node-node* tujuan yang sama. Nilai *throughput* tidak terlalu dipengaruhi oleh lama waktu simulasi. Letak *node* tujuan hanya berpengaruh pada *node* dengan lokasi yang tidak terlalu jauh maupun dekat (*node* dengan jarak menengah). Sebagai contoh pada *node-29* pada waktu simlasi 5 detik bernilai 122.385 kbps dan pada waktu simulasi 15 detik bernilai 127.412 kbps. Hal ini membuktikan lama waktu simulasi tidak terlalu berpengaruh pada nilai *throughput.* Namun lama waktu simulasi berpengaruh cukup signifikan pada *node* jarak menengah (*node*-16) dengan nilai sebesar 71.599 kbps pada waktu simulasi 5 detik dan 100.218 kbps pada waktu simulasi 15 detik. Pada protokol ini nilai *throughput* memiliki kestabilan nilai pada waktu simulasi yang berbeda. Hasil dapat dilihat pada gambar berikut.



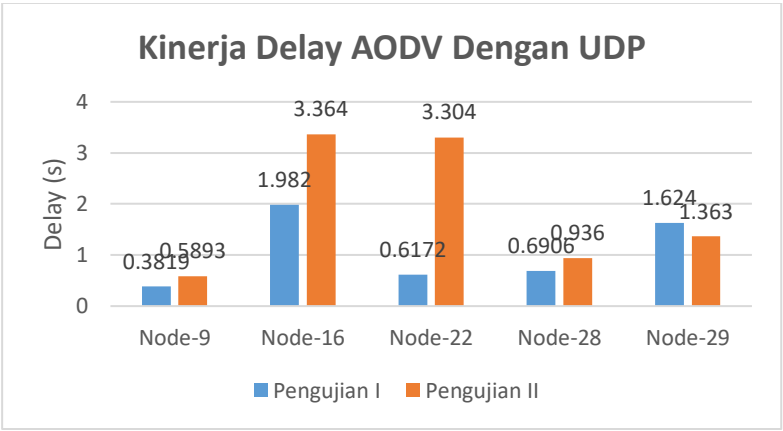
Gambar 4. 19 Kinerja *Throughput* AODV dengan TCP

Pada perhitungan nilai PDR protokol AODV, nilai PDR juga akan diambil perbandingannya sehingga dapat mengetahui bagaimana nilainya terhadap perbedaan waktu simulasi pada 2 skenario yang berbeda. Nilai PDR pada AODV tidak mengalami perubahan yang signifikan walaupun lama waktu simulasinya berbeda. Sehingga nilai PDR pada protokol AODV bisa dibilang lebih stabil daripada protokol *waypoint* dan *random walk* pada *node* tujuan yang sama. Namun nilai PDR protokol AODV mengalami penurunan pada *node* menengah dan kembali meningkat pada *node* tujuan yang jauh. Sebagai contoh dapat dilihat pada *node* dengan jarak terdekat dan jarak terjauh yaitu *node 9* dan *node 28*. Dari *node-9* didapatkan nilai PDR sebesar 98.15% pada pengujian 1 dan sebesar 99.93% pada pengujian II. Sementara pada *node-28* memiliki nilai PDR sebesar 86.76% pada pengujian I dan sebesar 87.64% pada pengujian II. Dari hal ini protokol AODV lebih memiliki tingkat PDR yang baik pada *node* tujuan dengan jarak yang jauh.

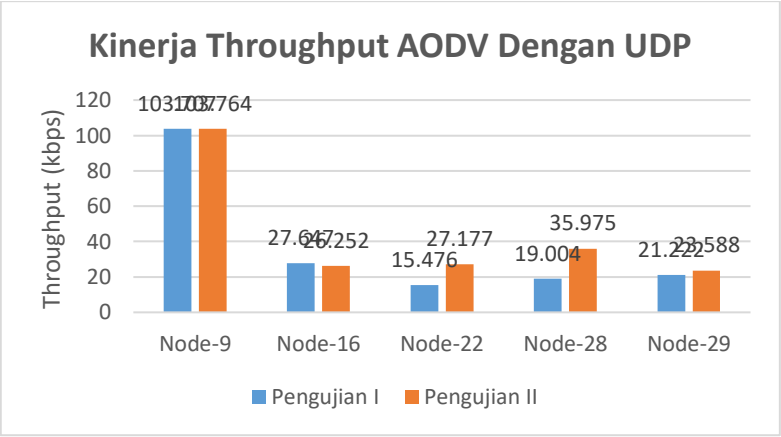


Gambar 4. 20 Kinerja PDR AODV dengan TCP

Selanjutnya adalah pengambilan perbandingan nilai *delay* dari protokol AODV dengan 2 skenario pengujian menggunakan protokol transport UDP. Dari hasil pada gambar dapat dilihat bahwa penggunaan UDP membuat *delay* pada AODV menigkat terutama pada *node* dengan jarak menengah. Namun *delay* kembali turun pada *node* dengan jarak yang lebih jauh.

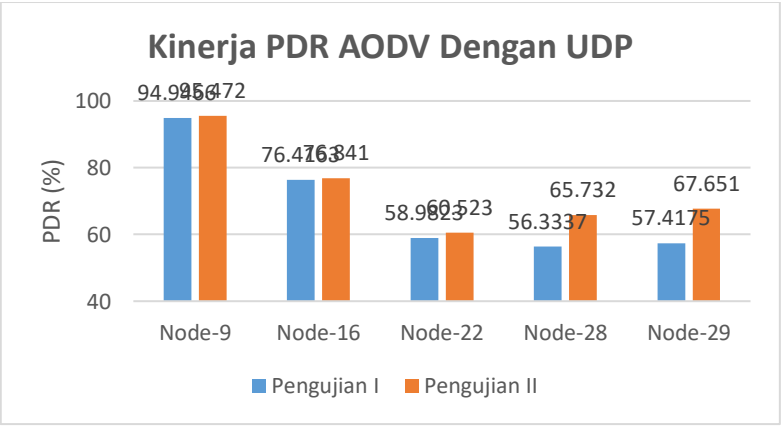


Gambar 4. 21 Kinerja *Delay* AODV dengan UDP



Gambar 4. 22 Kinerja *Throughput* AODV dengan UDP

Sementara pada perbandingan nilai *throughput* antara pengujian I dengan pengujian II menggunakan UDP pada protokol AODV dapat dilihat pada gambar. Dari hasil tersebut diketahui bahwa lama waktu simulasi tidak terlalu berpengaruh pada *node-node* yang diuji. Namun terlihat bahwa jarak antara *node* asal dengan tujuan memiliki pengaruh yang cukup besar dimana semakin jauh jarak maka semakin kecil nilai *throughput* yang dihasilkan.



Gambar 4. 23 Kinerja PDR AODV dengan UDP

Pada kinerja PDR AODV dengan UDP didapatkan hasil bahwa lama waktu simulasi tidak terlalu berpengaruh pada nilai PDR yang dihasilkan dan relatif sama pada tiap *node* yang diuji. Namun jarak antara *node* asal dengan tujuan memiliki pengaruh yang cukup signifikan dengan semakin jauh jarak maka semakin kecil nilai PDR yang dihasilkan.

4.4.3 Perbandingan Kinerja Protokol

Untuk mengetahui seberapa baik kinerja protokol *random walk* terhadap protokol lain yang dalam hal ini dipilih protokol *waypoint* dan AODV. Perbandingan antar protokol ini hanya dilakukan pada beberapa *node* tujuan yaitu *node-9, node-16, node-22, node-28* dan *node-29*, sebagaimana hasilnya telah ditampilkan pada tabel-tabel sebelumnya. Hasil perbandingan kinerja ketiga protokol akan dibandingkan terutama terkait dengan perbandingan parameter *delay*, *throughput* dan PDR.

a. Perbandingan Pada *Node* yang Diuji Dengan TCP

Pada parameter *delay*, didapatkan perbandingan antar protokol *routing* yang digunakan. Dari parameter *delay* tersebut dapat terlihat bagaimana nilai *delay* pada tiap *node* yang diujikan dengan 2 skenario pengujian yang memiliki lama waktu simulasi berbeda. Perbandingan *delay* antar protokol pada tiap *node* yang diuji dapat dilihat pada Tabel 4.15 berikut ini.

Tabel 4. 15 Perbandingan *Delay* Antar Protokol *Routing* dengan TCP

No	Tujuan	Delay (s)					
		Pengujian I			Pengujian II		
		RW	WP	AODV	RW	WP	AODV
1	*Node*-9	0.1551	0.1785	0.154	0.1607	0.6987	0.1584
2	*Node*-16	0.657	0.2834	0.2121	1.1807	0.9651	0.1958
3	*Node*-22	0.5677	0.2834	0.1087	0.6942	0.9651	0.145
4	*Node*-28	0.6193	0.4959	0.1967	0.4923	1.6889	0.1843
5	*Node*-29	0.6868	0.4251	0.1699	0.8239	1.4477	0.1704
Rata-rata		**0.53718**	**0.33326**	**0.16828**	**0.67036**	**1.1531**	**0.17078**

Adapun besarnya nilai *delay* dari hasil pengujian yang telah diujikan pada sejumlah *node* tujuan yang berbeda-beda, sebagaimana yang terlihat dalam Tabel 4.8, maka tampak bahwa dengan semakin jauh lokasi *node* tujuan maka semakin besar pula *delay* yang dihasilkan. Hanya saja secara umum, penggunaan protokol AODV memiliki delay yang relatif lebih baik dibanding protokol *random walk* maupun protokol *waypoint*. Sehingga dalam hal ini, penggunaan protokol *waypoint* akan berdampak pada *delay* yang relatif lebih besar kecuali pada *node* dengan jumlah *hop* dan waktu simulasi yang lebih sedikit.

Selanjutnya pada parameter *throughput*, dapat dilihat juga bagaimana nilai *throughput* antar protokol pada tiap *node* yang diujikan dengan 2 waktu simulasi yang berbeda yaitu 5 detik dan 15 detik. Hasil perbandingan tersebut akan digunakan sebagai acuan penentuan kinerja parameter *throughput* antar protokol. Perbandingan nilai *throughput* dapat dilihat pada Tabel 4.16 berikut.

Tabel 4. 16 Perbandingan *Throughput* Antar Protokol dengan TCP

No	**Tujuan**	**Throughput (kbps)**					
		Pengujian I			**Pengujian II**		
		RW	**WP**	**AODV**	**RW**	**WP**	**AODV**
1	*Node*-9	116.343	20.371	120.469	119.689	34.736	120.681
2	*Node*-16	124.312	24.172	71.599	94.823	42.714	100.218
3	*Node*-22	113.848	24.172	119.02	123.015	42.714	118.068
4	*Node*-28	139.555	42.302	59.082	138.46	74.75	53.085
5	*Node*-29	136.374	36.259	122.385	114.195	64.072	127.412
Rata-rata		**126.0864**	**29.4552**	**98.511**	**118.0364**	**51.7972**	**103.8928**

Pada perbandingan parameter *throughput*, sebagaimana yang terlihat pada Tabel 4.9 maka protokol *random walk* relatif paling baik pada semua *node* pengujian, sementara *waypoint* memiliki kinerja *throughput* yang paling buruk. Sedangkan protokol AODV memiliki nilai yang kurang stabil pada tiap *node* yang diujikan, hal ini menunjukkan bahwa protokol AODV hanya dapat memiliki nilai *throughput* yang baik pada *node-node* tertentu saja.

Kemudian pada perbandingan parameter *packet delivery ratio* (PDR), kinerjanya juga diuji pada *node-node* yang dipilih saja. Dari parameter *packet delivery ratio* (PDR) tersebut dapat dilihat bagaimana perbandingan kinerja antar protokol pada masing-masing *node* yang diuji. Untuk mengetahui perbandingan nilai *packet delivery ratio* (PDR) dari ketiga protokol tersebut.

Dari nilai perbandingan yang didapatkan pada masing-masing *node* yang diuji untuk parameter PDR, masing-masing protokol memiliki kinerja yang relatif sama, baik dalam *node* terdekat mauapun *node* terjauh. Hal ini diperlihatkan pada Tabel 4.10. Sehingga pemilihan penggunaan salah satu dari ketiga protokol tersebut tidak terlalu mempengaruhi nilai PDR yang dihasilkan pada masing-masing *node* yang diuji. Begitupun dengan perbandingan antara skenario pengujian I dengan skenario pengujian II. Itu artinya lama waktu simulasi juga tidak terlalu berpengaruh pada nilai PDR dengan ketiga protokol tersebut.

Tabel 4. 17 Perbandingan PDR Antar Protokol *Routing* dengan TCP

No	Tujuan	Packet Delivery Ratio (%)					
		Pengujian I			Pengujian II		
		RW	WP	AODV	RW	WP	AODV
1	*Node*-9	99.2644	92.5831	98.1548	99.7571	91.3579	99.9308
2	*Node*-16	84.3779	89.3734	89.1692	85.6255	90.9378	86.8269
3	*Node*-22	89.6005	89.3734	84.3259	90.9031	90.9378	84.7283
4	*Node*-28	87.5923	90.5812	86.7665	88.0007	88.1356	87.6415
5	*Node*-29	88.4781	90.7464	89.1203	88.5581	88.3064	88.0588
Rata-rata		**89.8626**	**90.5315**	**89.5073**	**90.5689**	**89.9351**	**89.4372**

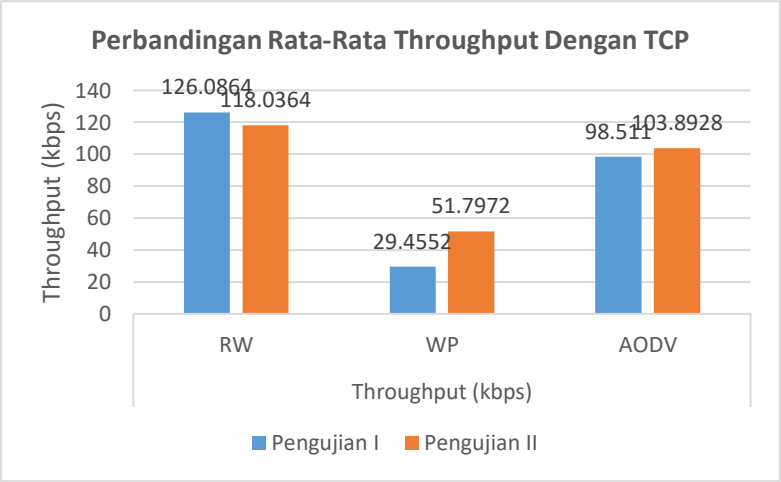
Dengan nilai PDR yang hampir sama pada setiap protokol, maka penentuan penggunaan protokol hanya akan berpengaruh besar pada nilai *delay* dan *throughput.*

b. Perbandingan Rata-Rata 2 Skenario Pengujian Dengan TCP

Pada bagian ini dilakukan perbandingan terhadapap rata-rata nilai parameter dari *node-node* yang telah diuji. Perbandingan rata-rata diambil dari rata-rata nilai parameter pada kelima node yang diuji dengan skenario

pengujian masing-masing. Perbandingan yang pertama adalah mengenai rata-rata nilai parameter *throughput.* Kemudian disusul dengan perbandingan pada rata-rata nilai parameter *delay*. Dan yang terkahir adalah perbandingan dari rata-rata nilai parameter PDR.

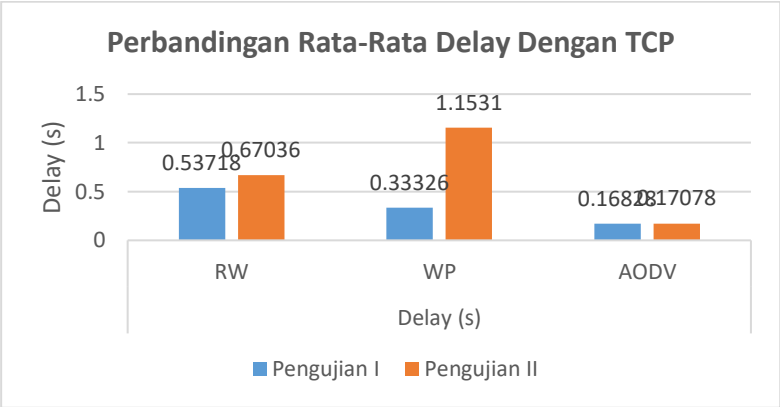
Dari keseluruhan pengujian parameter *random walk* baik delay, *throughput* ataupun PDR, maka secara rata-rata terbukti bahwa protokol ini lebih unggul dalam hal *throughput* dibanding protokol AODV maupun *waypoint*. Hal tersebut telah terbukti dari rata-rata nilai *throughput* yang dihasilkan dari tiap protokol yang telah diuji. Dan nilai rata-rata *throughput* dari protokol *random walk* juga lebih unggul dalam semua skenario pengujian, baik pada pengujian I dengan waktu simulasi 5 detik maupun pada pengujian II dengan waktu simulasi 15 detik. Nilai rata-rata *throughput* untuk tiap protokol yang diuji diambil dari Tabel 4.9 dan dapat dilihat pada gambar 4.15 berikut ini.



Gambar 4. 24 Perbandingan Performansi *Throughput* TCP

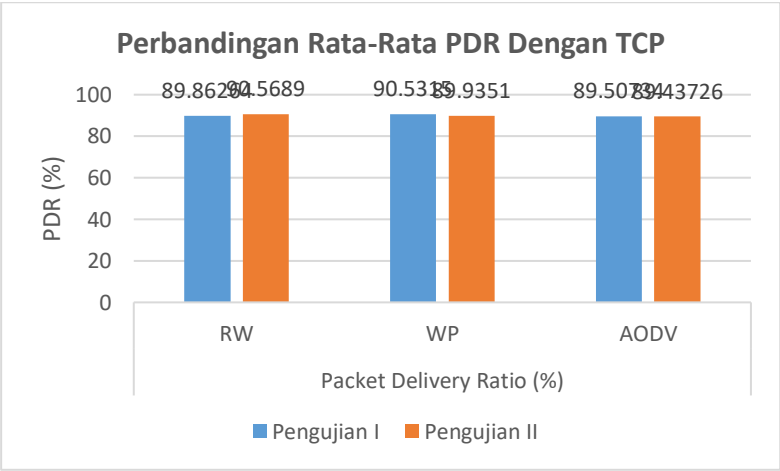
Sementara AODV yang didesain sebagai protokol *adhoc* yang bersifat reaktif, sudah sewajarnya memiliki *delay* yang lebih kecil. Hal ini dimungkinkan karena pada saat pencarian rute, ia akan mem-*broadcast* pemberitahuan ke seluruh *node* sensor tetangga dan akan memilih *node* tetangga yang memberi respon tercepat. Meski tidak terpaut besar, *delay*

random walk masih lebih baik dibandingkan *delay* protokol *waypoint*. Hal ini dapat dilihat pada Gambar 4.16.



Gambar 4. 25 Perbandingan Performansi *Delay* TCP

Terkait dengan parameter PDR, dari 2 skenario pengujian dengan nilai rata-rata PDR pada *node* telah diuji maka penggunaan ketiga protokol tidak memberikan perbedaan hasil yang signifikan (relatif sama). Hal ini dapat dilihat pada Gambar 4.17.



Gambar 4. 26 Perbandingan Performansi PDR TCP

Dari keseluruhan pembahasan diatas, nilai parameter yang diuji dari 2 skenario pengujian pada protokol *random walk* memiliki nilai rata-rata *delay* sebesar 0.60377 detik, nilai rata-rata *throughput* sebesar 122.0614 kbps dan nilai rata-rata PDR sebesar 90.21577%. Terkait dengan kinerja protokol *random walk* dibandingkan dengan protokol *waypoint* dan AODV, maka dengan target nilai PDR yang relatif sama, dapat disimpulkan bahwa protokol *random walk* memiliki nilai *throughput* yang paling baik dibandingkan kedua protokol yang lain. Sedangkan parameter delay-nya berada pada urutan kedua setelah protokol AODV.

c. Perbandingan Pada *Node* yang Diuji Dengan UDP

Pada parameter *delay*, didapatkan perbandingan antar protokol *routing* yang digunakan. Dari parameter *delay* tersebut dapat terlihat bagaimana nilai *delay* pada tiap *node* yang diujikan dengan 2 skenario pengujian yang memiliki lama waktu simulasi berbeda. Perbandingan *delay* antar protokol pada tiap *node* yang diuji dapat dilihat pada Tabel berikut ini.

Tabel 4. 18 Perbandingan *Delay* Antar Protokol *Routing* dengan UDP

No	Tujuan	Delay (s)					
		Pengujian I			Pengujian II		
		RW	WP	AODV	RW	WP	AODV
1	*Node*-9	0.38	0.2785	0.3819	0.599	0.7231	0.5893
2	*Node*-16	2.092	0.4832	1.982	3.595	1.1351	3.364
3	*Node*-22	2.104	0.4832	0.6172	3.308	1.1351	3.304
4	*Node*-28	1.45	0.9598	0.6906	6.207	1.9759	0.936
5	*Node*-29	1.641	0.7326	1.624	7.116	1.8476	1.363
Rata-rata		**1.5334**	**0.5874**	**1.05914**	**4.165**	**1.36336**	**1.91126**

Adapun besarnya nilai *delay* dari hasil pengujian yang telah diujikan pada sejumlah *node* tujuan yang berbeda-beda, sebagaimana yang terlihat dalam Tabel , maka tampak bahwa dengan semakin jauh lokasi *node* tujuan maka semakin besar pula *delay* yang dihasilkan. Hanya saja secara umum, penggunaan protokol *waypoint* memiliki delay yang relatif lebih baik dibanding protokol *random walk* maupun protokol AODV. Hal ini

dikarenakan penggunaan UDP pada jalur *routing* yang sudah ditentukan (stati) tidak menghasilkan perbedaan dengan penggunaan TCP denga protokol *routing* statis. Sehingga dalam hal ini, penggunaan protokol dinamis akan berdampak pada *delay* yang relatif lebih besar kecuali pada *node* dengan jumlah *hop* dan waktu simulasi yang lebih sedikit.

Selanjutnya pada parameter *throughput*, dapat dilihat juga bagaimana nilai *throughput* antar protokol pada tiap *node* yang diujikan dengan 2 waktu simulasi yang berbeda yaitu 5 detik dan 15 detik. Hasil perbandingan tersebut akan digunakan sebagai acuan penentuan kinerja parameter *throughput* antar protokol. Perbandingan nilai *throughput* dapat dilihat pada Tabel berikut.

Tabel 4. 19 Perbandingan *Throughput* Antar Protokol dengan UDP

No	**Tujuan**	**Throughput (kbps)**					
		Pengujian I			**Pengujian II**		
		RW	**WP**	**AODV**	**RW**	**WP**	**AODV**
1	*Node*-9	103.12	22.375	103.707	103.783	35.732	103.764
2	*Node*-16	31.316	26.183	27.647	38.021	44.742	26.252
3	*Node*-22	25.856	26.183	15.476	25.669	44.742	27.177
4	*Node*-28	56.004	45.342	19.004	46.181	76.758	35.975
5	*Node*-29	34.554	39.279	21.222	30.059	68.092	23.588
Rata-rata		**50.17**	**31.872**	**37.4112**	**48.7426**	**54.013**	**43.3512**

Pada perbandingan parameter *throughput*, sebagaimana yang terlihat pada Tabel maka protokol *random walk* relatif paling baik pada semua *node* pengujian, sementara *waypoint* memiliki kinerja *throughput* yang paling buruk. Sedangkan protokol *waypoint* memiliki nilai yang kurang stabil pada tiap *node* yang diujikan, hal ini ditunjukkan pada pengujian I mmemiliki nilai yang relatif kecil sedangkan pada pengujian II memiliki nilai yang relatif lebih besar.

Kemudian pada perbandingan parameter *packet delivery ratio* (PDR), kinerjanya juga diuji pada *node-node* yang dipilih saja. Dari parameter

packet delivery ratio (PDR) tersebut dapat dilihat bagaimana perbandingan kinerja antar protokol pada masing-masing *node* yang diuji. Untuk mengetahui perbandingan nilai *packet delivery ratio* (PDR) dari ketiga protokol tersebut dapat dilihat pada Tabel berikut.

Tabel 4. 20 Perbandingan PDR Antar Protokol *Routing* dengan UDP

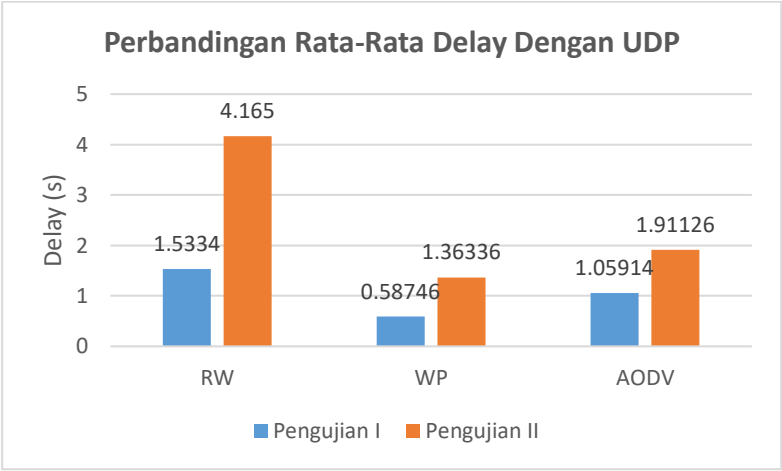
No	Tujuan	Packet Delivery Ratio (%)					
		Pengujian I			Pengujian II		
		RW	WP	AODV	RW	WP	AODV
1	*Node*-9	95.642	91.8315	94.9466	95.911	90.5793	95.472
2	*Node*-16	72.702	78.7343	76.4163	72.834	75.3789	76.841
3	*Node*-22	75.706	78.3347	58.9823	76.733	75.3789	60.523
4	*Node*-28	71.032	59.5833	56.3337	73.533	55.3561	65.732
5	*Node*-29	72.258	61.4642	57.4175	72.625	58.0643	67.651
Rata-rata		**77.468**	**73.9896**	**68.8192**	**78.327**	**70.9515**	**73.2438**

Dari nilai perbandingan yang didapatkan pada masing-masing *node* yang diuji untuk parameter PDR, masing-masing protokol memiliki kinerja yang berbeda, baik dalam *node* terdekat maupun *node* terjauh. Hanya saja secara keseluruhan protokol *routing random walk* memiliki nilai PDR yang sedikit lebih baik. Sehingga pemilihan penggunaan salah satu dari ketiga protokol tersebut akan mempengaruhi nilai PDR yang dihasilkan pada masing-masing *node* yang diuji. Sementara dengan perbandingan antara skenario pengujian I dengan skenario pengujian II, lama waktu simulasi tidak terlalu berpengaruh pada nilai PDR dengan ketiga protokol tersebut.

d. Perbandingan Rata-Rata 2 Skenario Pengujian Dengan UDP

Pada bagian ini dilakukan perbandingan terhadapap rata-rata nilai parameter dari *node-node* yang telah diuji. Perbandingan rata-rata diambil dari rata-rata nilai parameter pada kelima node yang diuji dengan skenario pengujian masing-masing. Perbandingan yang pertama adalah mengenai rata-rata nilai parameter *delay*. Kemudian disusul dengan perbandingan

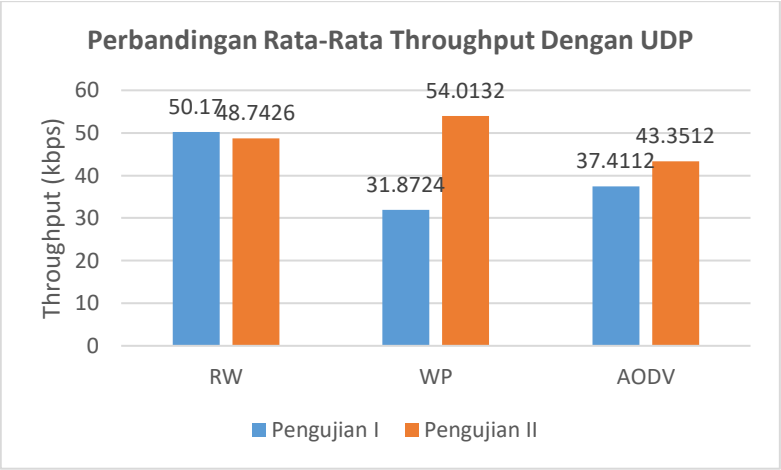
pada rata-rata nilai parameter *throughput*. Dan yang terkahir adalah perbandingan dari rata-rata nilai parameter PDR.



Gambar 4. 27 Perbandingan Performansi *Delay* UDP

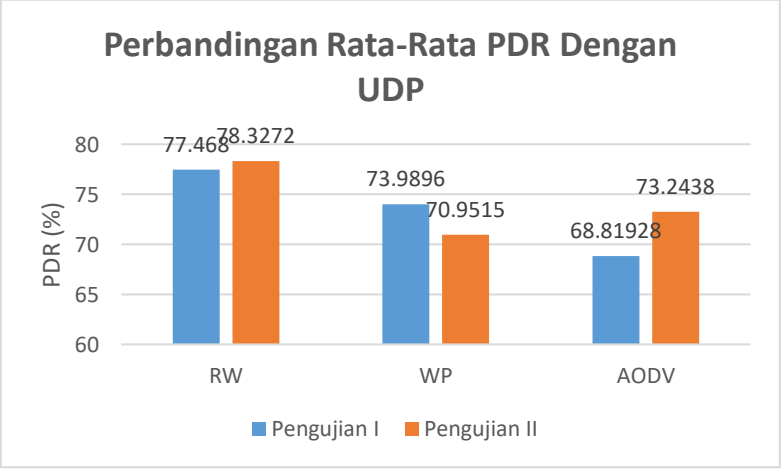
Dengan penggunaan UDP, *Waypoint* yang didesain sebagai protokol statis, sudah sewajarnya memiliki *delay* yang tidak berubah bila dibandingkan dengan TCP. Hal ini dimungkinkan karena pada *waypoint* tidak terjadi pencarian rute sehingga membuat seolah *delay* lebih kecil pada protokol transport *connection-less*. Sementara *Random Walk* dan AODV mengalami penigkatan nilai *delay* bila dibandingkan dengan TCP. Hal ini dipengaruhi oleh sifat dinamis kedua protokol tersebut sehingga dalam pengiriman data paket selalu melakukan perubahan jalur-jalur pengiriman yang membuat nilai *delay* semakin besar.

Secara rata-rata terbukti bahwa protokol *random walk* lebih unggul dalam hal *throughput* dibanding protokol AODV maupun *waypoint*. Hal tersebut telah terbukti dari rata-rata nilai *throughput* yang dihasilkan dari tiap protokol yang telah diuji. Dan nilai rata-rata *throughput* dari protokol *random walk* juga lebih unggul dalam semua skenario pengujian, baik pada pengujian I dengan waktu simulasi 5 detik maupun pada pengujian II dengan waktu simulasi 15 detik. Hal ini dapat dilihat pada Gambar.



Gambar 4. 28 Perbandingan Performansi *Throughput* UDP

Terkait dengan parameter PDR, dari 2 skenario pengujian dengan nilai rata-rata PDR pada *node* telah diuji maka penggunaan *random walk* memiliki nilai PDR yang lebih besar terpaut 3% hingga 5% dengan protokol *routing* lain. Hal ini dapat dilihat pada Gambar

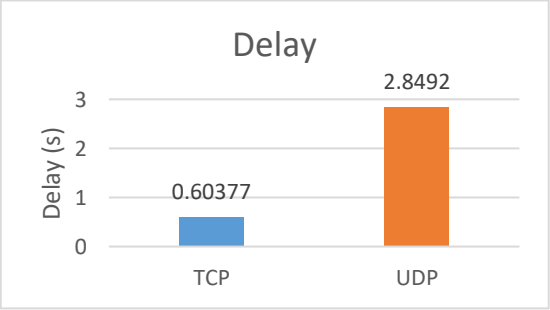


Gambar 4. 29 Perbandingan Performansi PDR UDP

Dari keseluruhan pembahasan dengan UDP diatas, nilai parameter yang diuji dari 2 skenario pengujian pada protokol *random walk* memiliki nilai rata-rata *delay* sebesar 2.8492 detik, nilai rata-rata *throughput* sebesar 49.4563 kbps dan nilai rata-rata PDR sebesar 77.8976%. Terkait dengan kinerja protokol *random walk* dibandingkan dengan protokol *waypoint* dan AODV, maka dapat disimpulkan bahwa protokol *random walk* memiliki nilai *throughput* dan PDR yang paling baik dibandingkan kedua protokol yang lain. Sedangkan parameter delay-nya berada pada urutan terakhir dari ketiga protokol *routing* tersebut.

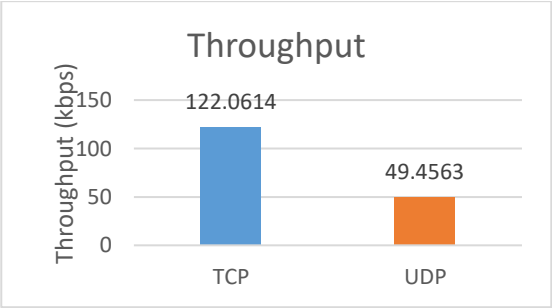
e. Perbandingan Protokol *Random Walk* Dengan TCP dan UDP

Dari penggunaan protokol transport TCP dan UDP pada protokol *routing Random Walk*, *Waypoint* dan AODV, maka hasilnya dapat dibandingkan. Perbandingan tersebut merupakan nilai rata-rata dari ketiga protokol *routing* tersebut dengan parameter *delay*, *throughput* dan *packet delivery ratio*. Hasil perbandingan dari ketiga protokol *routing* dengan TCP dan UDP dapat dilihat pada gambar dibawah ini.



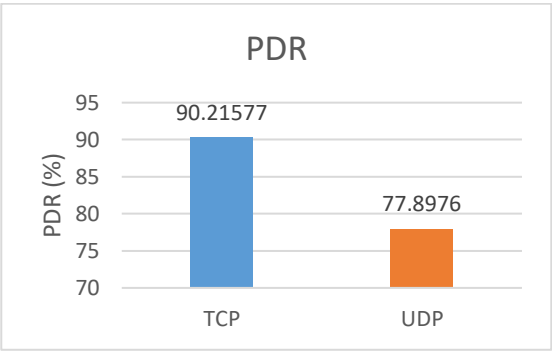
Gambar 4. 30 Perbandingan *Delay* TCP dan UDP

Dari perbandingan *delay random walk* dengan TCP dan UDP, dapat terlihat bahwa TCP memiliki nilai yang lebih baik dari pada UDP dengan nilai *delay* sebesar 0.60377 detik. Sementara pada penggunaan UDP memiliki nilai *delay* sebesar 2.8492 detik. Dari hal tersebut dapat diketahui bahwa penggunaan TCP pada protokol *routing random walk* memiliki kinerja *delay* yang jauh lebih baik daripada penggunaan UDP.



Gambar 4. 31 Perbandingan *Throughput* TCP dan UDP

Dari perbandingan *throughput random walk* dengan TCP dan UDP, dapat terlihat bahwa TCP memiliki nilai yang lebih baik dari pada UDP dengan nilai *throughput* sebesar 122.0614 kbps. Sementara pada penggunaan UDP memiliki nilai *throughput* sebesar 49.4563 kbps.



Gambar 4. 32 Perbandingan PDR TCP dan UDP

Begitupun dengan hasil perbandingan PDR dimana penggunaan TCP memiliki nilai yang lebih baik. Dengan nilai PDR TCP sebesar 90.21577 % dan UDP sebesar 77.8976 %. Maka dari itu dapat disimpulkan bahwa penggunaan TCP pada protokol *routing random walk* menghasilkan kinerja yang lebih baik daripada penggunaan UDP baik pada nilai *delay*, *throughput* maupun PDR.

BAB 5
PENUTUP

5.1 Kesimpulan

Setelah dilakukan pengkajian, pengujian dan pembahasan hasil penelitian yang diperoleh terkait dengan judul tugas akhir "Implementasi *Random walk Routing* Algorithm Pada Simulasi Jaringan Sensor Nirkabel", maka dapat diambil kesimpulan sebagai berikut:

1. Pemilihan jenis protocol *routing* akan mempengaruhi tingkat kinerja proses transmisi data pada jaringan (*node* sensor), terutama pada parameter delay, *throughput* dan packet delivery ratio (PDR).
2. Jumlah hop lintasan yang banyak akan menghasilkan delay yang tinggi. Delay tinggi bisa berpotensi pada nilai PDR yang rendah.
3. Protokol *random walk* memiliki jalur *routing* yang bersifat dinamis sehingga berpotensi menghasilkan *throughput* yang lebih baik disbanding waypoint yang bersifat statis.
4. Protokol AODV yang bersifat reaktif akan memperbarui tabel *routing*-nya setiap saat (bersifat adhoc), sehingga berpotensi memiliki delay yang paling kecil.
5. Dari seluruh pengujian yang telah dilakukan dengan protokol transport TCP, protokol *routing random walk* menghasilkan parameter delay, *throughput* dan PDR masing-masing sebesar 0.60377 detik, 122.0614 kbps dan 90.21577%. Protokol ini paling unggul dalam hal *throughput*, walaupun masih kalah unggul dalam hal delay dibanding AODV. Namun masih lebih baik di semua parameter dibanding protocol waypoint.
6. Dari keseluruhan pengujian dengan UDP, nilai parameter yang diuji dari 2 skenario pengujian pada protokol *random walk* memiliki nilai rata-rata delay sebesar 2.8492 detik, nilai rata-rata throughput sebesar 49.4563 kbps dan nilai rata-rata PDR sebesar 77.8976%.
7. Dari penggunaan TCP dan UDP pada protokol *random walk* dapat diketahui bahwa TCP memiliki kinerja yang lebih baik daripada UDP. Hal ini berlaku pada parameter pengukuran *delay*, *throughput* dan juga PDR.

5.2 Saran

Penggunaan algoritma *random walk* ternyata masih memiliki kelemahan, sehingga perlu ada beberapa perbaikan untuk penelitian berikutnya :

1. Meski masih unggul dalam hal *throughput*, algoritma *routing random walk* masih perlu dikembangkan untuk mengatasi berbagai kelemahan terutama dalam hal delay transmisi.
2. Jika kelemahan delay tidak bisa diatasi, algoritma *random walk* masih bisa dikembangkan dengan menambahkan metode lain yang dapat meningkatkan keamanan jaringan khususnya kerahasiaan sumber data (*node* asal).

DAFTAR PUSTAKA

[1] M. Kharratzadeh, "*Random walk Routing* in *Wireless* Sensor Networks", Semantic Scholar, 2010.

[2] I. Mabrouki, "*Random walk* Based *Routing* Protocol for *Wireless* Sensor Networks", in Proceeding ValueTools '07 Proceedings of the 2nd international conference on Performance evaluation methodologies and tools Article No. 71, 2007.

[3] K.D. Hanabaratti, R. Jogdand, "Design of an Efficient *Random walk Routing* Protocol for *Wireless* Sensor Networks", Research Gate, 2011.

[4] A.N Mian, M. Fatima, R. Khan, R. Prakash, "An Empirical Evaluation of Lightweight *Random walk* Based *Routing* Protocol in Duty Cycle Aware *Wireless* Sensor Network", The Scientific World Journal Vol. 2014, Article ID 946249, 2014.

[5] S.A. Amiri, KT. Foerster, R. Jacob, S. Schmid, "Charting the Algorithmic Complexity of Waypoint *Routing*", in ACM SIGCOMM Computer Communication Review Volume 48 Issue 1, January 2018.

[6] ER. M. Kaur, "*Routing* Protocols of *Wireless* Sensor Networks : A Review" in IRACST – International Journal of Computer Networks and *Wireless* Communications (IJCNWC), ISSN: 2250-3501 Vol.3, No1, February 2013.

[7] W. Dargie, C. Poellabauer, "FUNDAMENTALS OF *WIRELESS* SENSOR NETWORKS", Wiley Series on *Wireless* Communications and Mobile Computing, John Wliey & Sons Ltd. , 2010.

[8] C. Fischione, "An Introduction to *Wireless* Sensor Networks", Royal Institute Technology, September 2014.

[9] Fei Hu, "*Routing* Protocols for Sensor Networks", https://feihu.eng.ua.edu/WSN_*routing*.ppt , 2018.

[10] M. Veeraraghavan, "Three Planes in Networks", Elec. & Comp. Engg. Dept/CATT Polytechnic University Presentation, 2016.

[11] R.D. Poetra, "Analisis Perbandingan Performansi Antara Protokol *Routing* B.A.T.M.A.N dan AODV pada Skenario Bencana Alam", Tugas Akhir, DTE-FTE-ITS, Januari 2018.

[12] M.K. Sukma, I.D. Irawati, Hafidudin, "Analisa Perbandingan Kinerja *Routing* Protokol Pada *Wireless* Sensor Network (WSN) Dengan Metode Gradient Based Approach Dan Geographic Based

Approach", Telkom University e-Proceeding of Engineering : Vol.2 , No.1, pp. 169, April 2015.

[13] A. Ali, Y. Ming, S. Chakraborty, S. Iram, "A Comprehensive Survey on Real-Time Applications of WSN", School of Electronics and Information Engineering, Hebei University of Technology, Tianjin, China, November 2017.

[14] A. Cunha, A. Koubaa, R. Severino, M. Alves, "Open-ZB: an open-source implementation of the IEEE 802.15.4/ZigBee protocol stack on TinyOS", IEEE 4th International Conference on Mobile Adhoc and Sensor Systems, Pisa, Italy, October 2007.

[15] D. Smant, "The Random Walk Myth", https://sites.google.com/site/davesmant/monetary-economics/random-walk-myth, 2001.

[16] NS2 Manual, "NS2 Tutorial Handbook", 2014.

[17] Tracegraph Manual, "Tracegraph - a graphing software to plot the trace *file*s from NS2", https://www.nsnam.com/2012/09/tracegraph-graphing-software-to-plot.html, 2012.

LAMPIRAN

A. Listing Program NS2 (tcl *file*)

```
# This script is created by NSG2 beta1
# <http://wushoupong.googlepages.com/nsg>

#===================================
#     Simulation parameters setup
#===================================
set val(chan)   Channel/WirelessChannel    ;# channel type
set val(prop)   Propagation/TwoRayGround   ;# radio-propagation model
set val(netif)  Phy/WirelessPhy            ;# network interface type
set val(mac)    Mac/802_11                 ;# MAC type
set val(ifq)    Queue/DropTail/PriQueue    ;# interface queue type
set val(ll)     LL                         ;# link layer type
set val(ant)    Antenna/OmniAntenna        ;# antenna model
set val(ifqlen) 50                         ;# max packet in ifq
set val(nn)     30                         ;# number of mobilenodes
set val(rp)     RW/AODV/Waypoint           ;# routing protocol
set val(x)      1300                       ;# X dimension of topography
set val(y)      800                        ;# Y dimension of topography
set val(stop)   5.0/15.0                   ;# time of simulation end

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

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

#Open the NS trace file
```

```
set trace file [open RW/AODV/Waypoint.tr w]
$ns trace-all $trace file

#Open the NAM trace file
set nam file [open RW/AODV/Waypoint.nam w]
$ns namtrace-all $nam file
$ns namtrace-all-wireless $nam file $val(x) $val(y)
set chan [new $val(chan)];#Create wireless channel

#=================================
#     Mobile node parameter setup
#=================================
$ns node-config -adhocRouting  $val(rp) \
                -llType        $val(ll) \
                -macType       $val(mac) \
                -ifqType       $val(ifq) \
                -ifqLen        $val(ifqlen) \
                -antType       $val(ant) \
                -propType      $val(prop) \
                -phyType       $val(netif) \
                -channel       $chan \
                -topoInstance  $topo \
                -agentTrace    ON \
                -routerTrace   ON \
                -macTrace      ON \
                -movementTrace ON

#=================================
#       Nodes Definition
#=================================
#Create 30 nodes
set n0 [$ns node]
$n0 set X_ 300
$n0 set Y_ 300
$n0 set Z_ 0.0
$ns initial_node_pos $n0 20
```

```
set n1 [$ns node]
$n1 set X_ 414
$n1 set Y_ 420
$n1 set Z_ 0.0
$ns initial_node_pos $n1 20
set n2 [$ns node]
$n2 set X_ 481
$n2 set Y_ 305
$n2 set Z_ 0.0
$ns initial_node_pos $n2 20
set n3 [$ns node]
$n3 set X_ 445
$n3 set Y_ 203
$n3 set Z_ 0.0
$ns initial_node_pos $n3 20
set n4 [$ns node]
$n4 set X_ 582
$n4 set Y_ 114
$n4 set Z_ 0.0
$ns initial_node_pos $n4 20
set n5 [$ns node]
$n5 set X_ 606
$n5 set Y_ 236
$n5 set Z_ 0.0
$ns initial_node_pos $n5 20
set n6 [$ns node]
$n6 set X_ 603
$n6 set Y_ 369
$n6 set Z_ 0.0
$ns initial_node_pos $n6 20
set n7 [$ns node]
$n7 set X_ 531
$n7 set Y_ 523
$n7 set Z_ 0.0
$ns initial_node_pos $n7 20
set n8 [$ns node]
```

```
$n8 set X_ 346
$n8 set Y_ 483
$n8 set Z_ 0.0
$ns initial_node_pos $n8 20
set n9 [$ns node]
$n9 set X_ 331
$n9 set Y_ 118
$n9 set Z_ 0.0
$ns initial_node_pos $n9 20
set n10 [$ns node]
$n10 set X_ 498
$n10 set Y_ 30
$n10 set Z_ 0.0
$ns initial_node_pos $n10 20
set n11 [$ns node]
$n11 set X_ 765
$n11 set Y_ 210
$n11 set Z_ 0.0
$ns initial_node_pos $n11 20
set n12 [$ns node]
$n12 set X_ 753
$n12 set Y_ 353
$n12 set Z_ 0.0
$ns initial_node_pos $n12 20
set n13 [$ns node]
$n13 set X_ 696
$n13 set Y_ 433
$n13 set Z_ 0.0
$ns initial_node_pos $n13 20
set n14 [$ns node]
$n14 set X_ 663
$n14 set Y_ 542
$n14 set Z_ 0.0
$ns initial_node_pos $n14 20
set n15 [$ns node]
$n15 set X_ 743
```

```
$n15 set Y_ 55
$n15 set Z_ 0.0
$ns initial_node_pos $n15 20
set n16 [$ns node]
$n16 set X_ 967
$n16 set Y_ 67
$n16 set Z_ 0.0
$ns initial_node_pos $n16 20
set n17 [$ns node]
$n17 set X_ 1001
$n17 set Y_ 243
$n17 set Z_ 0.0
$ns initial_node_pos $n17 20
set n18 [$ns node]
$n18 set X_ 953
$n18 set Y_ 413
$n18 set Z_ 0.0
$ns initial_node_pos $n18 20
set n19 [$ns node]
$n19 set X_ 823
$n19 set Y_ 526
$n19 set Z_ 0.0
$ns initial_node_pos $n19 20
set n20 [$ns node]
$n20 set X_ 957
$n20 set Y_ 581
$n20 set Z_ 0.0
$ns initial_node_pos $n20 20
set n21 [$ns node]
$n21 set X_ 1100
$n21 set Y_ 493
$n21 set Z_ 0.0
$ns initial_node_pos $n21 20
set n22 [$ns node]
$n22 set X_ 767
$n22 set Y_ 651
```

```
$n22 set Z_ 0.0
$ns initial_node_pos $n22 20
set n23 [$ns node]
$n23 set X_ 602
$n23 set Y_ 668
$n23 set Z_ 0.0
$ns initial_node_pos $n23 20
set n24 [$ns node]
$n24 set X_ 1132
$n24 set Y_ 588
$n24 set Z_ 0.0
$ns initial_node_pos $n24 20
set n25 [$ns node]
$n25 set X_ 1102
$n25 set Y_ 262
$n25 set Z_ 0.0
$ns initial_node_pos $n25 20
set n26 [$ns node]
$n26 set X_ 1210
$n26 set Y_ 352
$n26 set Z_ 0.0
$ns initial_node_pos $n26 20
set n27 [$ns node]
$n27 set X_ 962
$n27 set Y_ 675
$n27 set Z_ 0.0
$ns initial_node_pos $n27 20
set n28 [$ns node]
$n28 set X_ 1178
$n28 set Y_ 705
$n28 set Z_ 0.0
$ns initial_node_pos $n28 20
set n29 [$ns node]
$n29 set X_ 1266
$n29 set Y_ 500
$n29 set Z_ 0.0
```

```
$ns initial_node_pos $n29 20

#=================================
#       Agents Definition
#=================================
#Setup a TCP connection
set tcp7 [new Agent/TCP]
$ns attach-agent $n0 $tcp7
set sink8 [new Agent/TCPSink]
$ns attach-agent $n29 $sink8
$ns connect $tcp7 $sink8
$tcp7 set packetSize_ 1500


#=================================
#       Applications Definition
#=================================
#Setup a FTP Application over TCP connection
set ftp3 [new Application/FTP]
$ftp3 attach-agent $tcp7
$ns at 0.0 "$ftp3 start"
$ns at 5.0 "$ftp3 stop"


#=================================
#       Termination
#=================================
#Define a 'finish' procedure
proc finish {} {
    global ns tracefile namfile
    $ns flush-trace
    close $tracefile
    close $namfile
    exec nam RW/AODV/Waypoint.nam &
    exit 0
}
```

```
for {set i 0} {$i < $val(nn) } { incr i } {
    $ns at $val(stop) "\$n$i reset"
}
$ns at $val(stop) "$ns nam-end-wireless $val(stop)"
$ns at $val(stop) "finish"
$ns at $val(stop) "puts \"done\" ; $ns halt"
$ns run
```

B. Tabel Pengujian *Random Walk* dengan TCP

Tabel Pengujian Delay Random Walk

No	Tujuan	Delay (s)		Rata-rata
		Pengujian I (5 detik)	Pengujian II (15 detik)	
1	*Node*-1	0.1565	0.1611	0.1588
2	*Node*-2	0.1562	0.1611	0.15865
3	*Node*-3	0.1543	0.1613	0.1578
4	*Node*-4	0.3051	0.3193	0.3122
5	*Node*-5	0.1062	0.258	0.1821
6	*Node*-6	0.3039	0.319	0.31145
7	*Node*-7	0.2926	0.3186	0.3056
8	*Node*-8	0.1537	0.161	0.15735
9	*Node*-9	0.1551	0.1607	0.1579
10	*Node*-10	0.3049	0.3193	0.3121
11	*Node*-11	0.4436	0.4816	0.4626
12	*Node*-12	0.4282	0.471	0.4496
13	*Node*-13	0.4049	0.4689	0.4369
14	*Node*-14	0.4251	0.4693	0.4472
15	*Node*-15	0.4169	0.4724	0.44465
16	*Node*-16	0.657	1.1807	0.91885
17	*Node*-17	0.5018	0.7107	0.60625
18	*Node*-18	0.5136	0.7049	0.60925
19	*Node*-19	0.5226	0.6973	0.60995
20	*Node*-20	0.3003	0.3262	0.31325
21	*Node*-21	0.4555	0.3544	0.40495
22	*Node*-22	0.5677	0.6942	0.63095
23	*Node*-23	0.4457	0.5911	0.5184
24	*Node*-24	0.5169	0.404	0.46045
25	*Node*-25	0.2979	0.3601	0.329

26	*Node*-26	0.4597	0.3879	0.4238
27	*Node*-27	0.2061	0.2938	0.24995
28	*Node*-28	0.6193	0.4923	0.5558
29	*Node*-29	0.6868	0.8239	0.75535

Tabel Pengujian Throughput Random Walk

No	**Tujuan**	***Throughput* (kbps)**		**Rata-rata**
		Pengujian I (5 detik)	**Pengujian II (15 detik)**	
1	*Node*-1	119.015	120.633	119.824
2	*Node*-2	119.775	120.633	120.204
3	*Node*-3	116.072	119.447	117.7595
4	*Node*-4	119.749	120.63	120.1895
5	*Node*-5	51.449	98.16	74.8045
6	*Node*-6	120.08	120.724	120.402
7	*Node*-7	117.056	121.02	119.038
8	*Node*-8	115.304	120.047	117.6755
9	*Node*-9	116.343	119.689	118.016
10	*Node*-10	120.375	121.064	120.7195
11	*Node*-11	119.886	118.799	119.3425
12	*Node*-12	119.694	121.042	120.368
13	*Node*-13	116.432	121.062	118.747
14	*Node*-14	117.809	120.535	119.172
15	*Node*-15	120.444	120.59	120.517
16	*Node*-16	124.312	94.823	109.5675
17	*Node*-17	86.444	104.315	95.3795
18	*Node*-18	112.536	108.204	110.37
19	*Node*-19	115.942	123.094	119.518
20	*Node*-20	122.202	122.612	122.407

21	*Node*-21	128.56	128.178	128.369
22	*Node*-22	113.848	123.015	118.4315
23	*Node*-23	68.589	103.692	86.1405
24	*Node*-24	134.7	130.884	132.792
25	*Node*-25	122.04	126.496	124.268
26	*Node*-26	123.912	125.82	124.866
27	*Node*-27	122.702	128.833	125.7675
28	*Node*-28	139.555	138.46	139.0075
29	*Node*-29	136.374	114.195	125.2845

Tabel Pengujian Packet Delivery Ratio Random Walk

No	**Tujuan**	**Packet Delivery Ratio (%)**		**Rata-rata**
		Pengujian I (5 detik)	**Pengujian II (15 detik)**	
1	*Node*-1	99.2558	99.7556	99.5057
2	*Node*-2	99.2549	99.7556	99.50525
3	*Node*-3	99.2662	99.7567	99.51145
4	*Node*-4	97.5219	98.9943	98.2581
5	*Node*-5	95.5462	98.7904	97.1683
6	*Node*-6	98.0985	99.1899	98.6442
7	*Node*-7	97.2586	98.3721	97.81535
8	*Node*-8	99.2285	99.7567	99.4926
9	*Node*-9	99.2644	99.7571	99.51075
10	*Node*-10	98.0847	99.2688	98.67675
11	*Node*-11	97.6587	98.6556	98.15715
12	*Node*-12	98.4184	99.3807	98.89955
13	*Node*-13	96.5571	97.93	97.24355
14	*Node*-14	97.7335	98.6425	98.188
15	*Node*-15	97.9656	99.1134	98.5395

16	*Node*-16	84.3779	85.6255	85.0017
17	*Node*-17	92.327	92.7033	92.51515
18	*Node*-18	92.4652	93.7196	93.0924
19	*Node*-19	91.8227	91.1795	91.5011
20	*Node*-20	90.7226	88.4706	89.5966
21	*Node*-21	88.9502	89.5392	89.2447
22	*Node*-22	89.6005	90.9031	90.2518
23	*Node*-23	93.4626	97.4823	95.47245
24	*Node*-24	86.2059	88.5255	87.3657
25	*Node*-25	90.2006	88.7467	89.47365
26	*Node*-26	88.4593	88.7823	88.6208
27	*Node*-27	89.9602	89.11	89.5351
28	*Node*-28	87.5923	88.0007	87.7965
29	*Node*-29	88.4781	88.5581	88.5181

C. Tabel Pengujian *Random Walk* dengan UDP

Tabel Pengujian Delay Random Walk

No	Tujuan	Delay (s)		Rata-rata
		Pengujian I (5 detik)	Pengujian II (15 detik)	
1	*Node*-1	0.378	0.597	0.4875
2	*Node*-2	0.378	0.597	0.4875
3	*Node*-3	0.368	0.593	0.4805
4	*Node*-4	1.15	1.356	1.253
5	*Node*-5	1.245	1.577	1.411
6	*Node*-6	1.182	1.452	1.317
7	*Node*-7	1.138	1.351	1.2445
8	*Node*-8	0.423	0.689	0.556
9	*Node*-9	0.38	0.599	0.4895
10	*Node*-10	0.924	1.121	1.0225
11	*Node*-11	1.055	1.777	1.416
12	*Node*-12	1.084	1.658	1.371
13	*Node*-13	1.617	2.017	1.817
14	*Node*-14	1.492	2.137	1.8145
15	*Node*-15	1.372	2.233	1.8025
16	*Node*-16	2.092	3.595	2.8435
17	*Node*-17	1.901	2.82	2.3605
18	*Node*-18	1.933	2.567	2.25
19	*Node*-19	2.032	3.724	2.878
20	*Node*-20	1.543	2.235	1.889
21	*Node*-21	1.674	2.346	2.01

22	*Node*-22	2.104	3.308	2.706
23	*Node*-23	1.899	2.891	2.395
24	*Node*-24	1.985	2.992	2.4885
25	*Node*-25	1.455	3.588	2.5215
26	*Node*-26	1.731	2.782	2.2565
27	*Node*-27	1.353	2.48	1.9165
28	*Node*-28	1.45	6.207	3.8285
29	*Node*-29	1.641	7.116	4.3785

Tabel Pengujian Throughput Random Walk

No	**Tujuan**	***Throughput* (kbps)**		**Rata-rata**
		Pengujian I (5 detik)	**Pengujian II (15 detik)**	
1	*Node*-1	103.12	103.783	103.4515
2	*Node*-2	103.12	103.783	103.4515
3	*Node*-3	102.98	102.98	102.98
4	*Node*-4	47.073	48.46	47.7665
5	*Node*-5	56.234	58.902	57.568
6	*Node*-6	52.778	51.335	52.0565
7	*Node*-7	48.218	48.531	48.3745
8	*Node*-8	68.024	69.704	68.864
9	*Node*-9	103.12	103.783	103.4515
10	*Node*-10	82.356	81.927	82.1415
11	*Node*-11	76.778	77.779	77.2785
12	*Node*-12	43.35	45.451	44.4005
13	*Node*-13	29.715	30.646	30.1805

14	*Node*-14	68.778	70.091	69.4345
15	*Node*-15	29.991	31.113	30.552
16	*Node*-16	31.316	38.021	34.6685
17	*Node*-17	35.333	34.456	34.8945
18	*Node*-18	42.762	43.982	43.372
19	*Node*-19	27.285	26.933	27.109
20	*Node*-20	34.436	36.656	35.546
21	*Node*-21	29.935	30.871	30.403
22	*Node*-22	25.856	25.669	25.7625
23	*Node*-23	22.256	23.334	22.795
24	*Node*-24	37.269	40.321	38.795
25	*Node*-25	39.978	43.617	41.7975
26	*Node*-26	42.123	40.032	41.0775
27	*Node*-27	40.086	38.877	39.4815
28	*Node*-28	56.004	46.181	51.0925
29	*Node*-29	34.554	30.059	32.3065

Tabel Pengujian Packet Delivery Ratio Random Walk

No	**Tujuan**	**Packet Delivery Ratio (%)**		**Rata-rata**
		Pengujian I (5 detik)	**Pengujian II (15 detik)**	
1	*Node*-1	95.642	95.911	95.7765
2	*Node*-2	95.643	95.912	95.7775
3	*Node*-3	94.457	94.498	94.4775
4	*Node*-4	81.771	82.637	82.204
5	*Node*-5	80.732	81.881	81.3065

6	*Node*-6	82.823	83.735	83.279
7	*Node*-7	80.447	82.45	81.4485
8	*Node*-8	85.889	86.653	86.271
9	*Node*-9	95.642	95.911	95.7765
10	*Node*-10	89.902	90.023	89.9625
11	*Node*-11	83.324	82.435	82.8795
12	*Node*-12	80.818	81.912	81.365
13	*Node*-13	78.236	78.894	78.565
14	*Node*-14	77.792	78.872	78.332
15	*Node*-15	73.956	74.321	74.1385
16	*Node*-16	72.702	72.834	72.768
17	*Node*-17	73.717	72.771	73.244
18	*Node*-18	74.567	76.292	75.4295
19	*Node*-19	75.866	76.514	76.19
20	*Node*-20	79.892	80.391	80.1415
21	*Node*-21	78.223	77.652	77.9375
22	*Node*-22	75.706	76.733	76.2195
23	*Node*-23	72.134	73.142	72.638
24	*Node*-24	71.521	72.017	71.769
25	*Node*-25	70.54	71.844	71.192
26	*Node*-26	70.587	71.234	70.9105
27	*Node*-27	72.101	73.346	72.7235
28	*Node*-28	71.032	73.533	72.2825
29	*Node*-29	72.258	72.625	72.4415

BIODATA PENULIS

Muhammad Danang Retzafakhri merupakan pemuda kelahiran Magetan, 19 Maret 1996 yang sering dipanggil Danang. Penulis tinggal di kelurahan Mangundikaran, Kabupaten Nganjuk. Penulis memulai pendidikan formal saat pendidikan dasar di SD Aisyiyah 1 Nganjuk, pendidikan menengah di SMPN 1 Nganjuk dan SMAN 2 Nganjuk, selanjutnya pada tahun 2014 melanjutkan pendidikan tinggi di Departemen Teknik Elektro Institut Teknologi Sepuluh Nopember Surabaya. Penulis berusaha mewujudkan keinginan menjadi seorang Engineer dalam bidang Teknik Elektro khusunya Telekomunikasi. Selanjutnya penulis dapat dihubungi melalui: danangretza23@gmail.com

Halaman ini sengaja dikosongkan