Simulasi Protokol 6LoWPAN Mengunakan Cooja Simulator

Johan Ambarita[1], Felix N. Sitorus[2], Michael I. P. Siagian[3], Noel A. Sinaga[4], Ewis M. Simbolon[5], Albert Sagala[6]
[123456]Program Studi Teknik Elektro, Fakultas Teknik Informatika dan Elektro
Institut Teknologi Del, Laguboti, Toba Samosir, Sumatera Utara, Indonesia
[1]Ambaritajohan10@gmail.com, [2]felix.n.sitorus@gmail.com, [3]mips0110@gmail.com, [4]noelsinaga1627@gmail.com,
[5]ewismsimbolon03@gmail.com, [6]albert@del.ac.id

***Abstrak* — Pada perkembangan *internet of thing* (IOT) yang cepat pada zaman sekarang yang menyebabkan banyaknya perangkat – perangkat yang berbasis IOT yang beredar di pasaran. Hal tersebut mengakibatkan performa dari perangkat tersebut harus diperhatikan seperti masalah ketersediaan alamat pada internet protocol (IP) dan masalah keamanan data yang akan diproses menjadi salah satu faktor yang harus diperhatikan. Masalah perutean juga menjadi masalah yang dihadapi IOT, perutean yang dilakukan pada perangkat yang umum dipakai mengharuskan perangkat membutuhkan daya yang besar untuk memproses dan memtransmisikan data tersebut. Dalam penelitian ini akan menggunakan 6LoWPAN (*IPv6 over Low power Wireless Personal Area Networks* yang mengunakan protokol IPv6 untuk dapat mengatasi permasalahan ketersediaan alamat IP dan keamanan yang dialami selama ini. Ptotokol 6LoWPAN yang dikombinasikan dengan protokol perutean RPL (*Routing Protocol for Low power and lossy networks*) yang cocok digunakan pada perangkat yang memiliki daya yang terbatas. Pada penelitian ini dilakukan simulasi penggunaan protokol 6LoWPAN pada jaringan sensor nirkabel atau *wireless* sensor network (WSN) menggunakan Cooja simulator. Setelah itu dilakukan analisis menggunakan wireshark dan fitur collect view pada cooja dengan meninjau parameter Expected Transmission Count (ETX), *Throughput*, konsumsi energi, waktu konvergensi jaringan, rasio pengiriman paket dan jumlah paket kontrol RPL. Simulasi dilakukan dengan 49 sensor node dan 1 sink node pada dimensi lingkungan berjarak 40.000 m^2.**

Kata kunci : Internet of Things, Wireless sensor network, Quality of service, 6LoWPAN, RPL protokol

I. PENDAHULUAN

Teknologi *Wireless* Sensor Network (WSN) melibatkan banyak perangkat komunikasi, sehingga teknologi WSN cenderung sangat beragam, dimana teknologi ini umumnya melibatkan perangkat tertanam (*embedded device*) atau mikrokontroler, hingga perangkat sensor. Perkembangan dari teknologi WSN menghasilkan paradigma baru yang saat ini dikenal sebagai Internet of Things (IoT). Perangkat tertanam memiliki keterbatasan energi, sehingga perangkat pada WSN mentransmisikan sinyal dengan daya rendah dan komputasi yang rendah juga. Untuk memfasilitasi konektifitas WSN dan Internet maka Internet Engineering Task Force (IETF) membentuk protokol IPv6 over Low Power *Wireless* Personal Area Network (6LoWPAN). Pengembangan Ipv6 tidak terlepas dari kelebihan yang dimilikinya[1].

Berikut kelebihan – kelebihan yang dimiliki Ipv6[2]:

- Memiliki ketersediaan alamat IP sebanyak 3.4 x 10^{38} alamat. Hal ini mengatasi keterbatasan dari ketersediaan alamat yang dimiliki IPv4.
- IPv6 menyediakan fitur konfigurasi otomatis. Konfigurasi juga mudah untuk dilakukan terutama untuk pemasangan berskala besar.
- Dikarenakan ruang alamat yang luas, pengalamatan langsung dimungkinkan yang mengakibatkan pengalamatan dapat dilakukan secara efektif.
- Memiliki tingkat keamanan yang lebih baik dikarenakan dilengkapi oleh protokol IPSEC.
- IPv6 menyediakan kemampuan interoperabilitas dan mobilitas yang sudah tertanam dalam perangkat jaringan.

Dengan kelebihan – kelebihan yang dimiliki IPv6 , keamanan dan ketersediaan alamat IP yang menjadi perhatian utama di IOT akan terselesaikan.

Untuk mengatasi permasalahan dalam penggunaan daya yang kecil, protokol yang digunakan Routing Protocol for Low-Power and Lossy Network (RPL) yang cocok digunakan pada perangkat yang memiliki daya penyimpanan energi yang rendah. Beberapa contoh perangkat yang dapat menggunakan protokol routing RPL adalah MicaZ, Zolertia, Sky, dan ESB. Perangkat-perangkat ini dapat disimulasikan melalui simulator Cooja. Penggunaan simulator dapat berperan penting dalam hal proses prototyping dan pengujian skala besar dibandingkan dengan yang dilakukan di dunia nyata terlebih hasil dari simulasi dari cooja simulator dapat diupload ke hardware sensor node. Oleh karena itu, pada penelitian ini dilakukan suatu simulasi pada simulator cooja ketimbang melakukan pengamatan langsung pada hardware. Penelitian penggunaan energi sangat kompleks jika dilakukan pada dunia nyata, karena memerlukan banyak pertimbangan parameter lingkungan penelitian. Namun dengan menggunakan simulator, penulis dapat membatasi parameter - parameter tersebut sehingga lingkungan penelitian menjadi terkendali serta bisa mendapatkan output yang dikehendaki[3].

Cooja simulator merupakan aplikasi berbasis Java yang menyediakan tiga fitur utama yaitu, *graphical user interface*, *simulation*, dan, *extensible framework*. Penggunaan simulator jaringan diperlukan, karena mereproduksi lingkungan propagasi nirkabel adalah hal yang nyaris mustahil. Adapun penggunaan sensor atau perangkat nirkabel pada dunia nyata dinilai terlalu mahal dan kurang praktis. Pemilihan simulator ini juga tidak terlepas dari kepopuleran simulator ini yang memyebabkan banyaknya bahan pembelajaran bagi penulis.

Penyusunan paper ini adalah sebagai berikut. Pada bab 2, akan dijelaskan tentang teori dan pemahaman yang dibutuhkan untuk memahami penelitian ini beserta penjelasan tentang software yang digunakan. Pada bab 3, berisi tentang metode penelitian yang akan digunakan untuk memperoleh data beserta deskripsi tentang lingkungan dan konfigurasi dari simulasi yang dilakukan. Penjelasan hasil dan analisa pada penelitian ini akan dijelaskan pada bab 4. Bab 5, berisi kesimpulan penelitian yang telah dilakukan.

II. TINJAUAN PUSTAKA

A. *Sensor Jaringan Nirkabel*

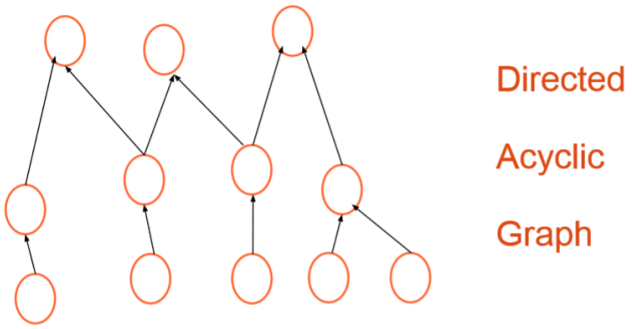
WSN (*Wireless* Sensor Network) atau jaringan sensor nirkabel merupakan suatu jaringan nirkabel yang terdiri dari beberapa sensor (sensor node) yang diletakkan ditempat - tempat yang berbeda untuk memonitoring kondisi suatu lingkungan. *Wireless Sensor Network* (WSN) adalah cabang baru dalam jaringan komputer yang terdiri dari beberapa sensor node yang saling berkomunikasi dan bekerja sama untuk mengumpulkan data - data dari lingkungan sekitar, misalnya suhu, tekanan udara, kelembapan udara dan beberapa parameter lingkungan lainnya. Untuk keperluan ini suatu node diperlengkapi dengan peralatan sensor yang digunakan untuk mendeteksi lingkungan sekitar dan perlatan komunikasi yang digunakan untuk berkomunikasi dengan sensor node yang lain dan sink. Selain itu sensor node juga di lengkapi dengan peralatan pemrosesan data, penyimpanan data sementra atau memory, peralatan komunikasi dan *power supply* atau baterai. Dalam aplikasi WSN, sensor node harus mempunyai dimensi yang sangat kecil, sehingga sensor node mempunyai keterbatasan baik dalam prosesor, memori atau *wireless*. Oleh karena itu banyak para peneliti tertarik dalam pengoptimalan kebutuhan energi dalam pemrosesan data ataupun proses komunikasi ke sink pada WSN.

WSN dapat diterapkan diberbagai bidang, umumnya digunakan untuk melakukan aktivitas *monitoring* dan *tracking*. Dalam bidang antisipasi dan pencegahan bencana, sensor dapat digunakan untuk mendeteksi kemungkinan bencana. Sensor diletakkan di berbagai daerah, ketika kemungkinan adanya bencana terdeteksi maka sensor akan mengirimkan data ke stasiun pusat. Selanjutnya di stasiun pusat terjadi pengolahan data, memberikan peringatan dini akan adanya bencana kepada para penduduk. Pemberitahuan dapat melalui berbagai media, melalui internet ataupun sms. Selain itu, informasi dari sensor - sensor dalam jaringan digabungkan dengan *Geographic Information System* dimungkinkan untuk mengetahui dimana titik aman yang terlindung dari bencana. Para penduduk selanjutnya bisa mengambil informasi tersebut dan mengeceknya pada GPS untuk melihat peta lokasi dari daerah yang aman bencana[4].

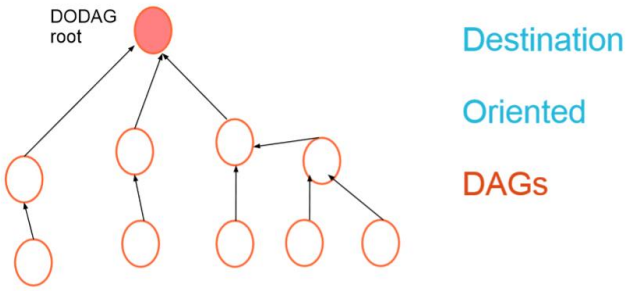
B. Routing Protocol for Low-Power and Lossy Network *(RPL)*

Low power dan Lossy Networks (LLN) terdiri dari jumlah node yang terbatas memiliki energi, memori, dan kapasitas pemrosesan yang terbatas. RPL dirancang sedemikian rupa sehingga fungsi beberapa *RPL Instance* (sekumpulan satu atau lebih DODAG yang berbagi RPLInstanceID) dapat berjalan pada waktu bersamaan dan penerusan paket dipisahkan dari optimisasi routing untuk mendukung jaringan sensor nirkabel. [5]

Konstruksi topologi adalah salah satu tujuan utama RPL, karena LLN biasanya tidak memiliki topologi yang telah ditentukan. RPL menetapkan satu atau lebih *root* agar berfungsi sebagai sink dan kemudian membentuk rute dari atau menuju sink. Rute yang dihasilkan membentuk *Directed Acyclic Graph* (DAG) sebagai topologi, yang selanjutnya dipartisi menjadi beberapa *Destination Oriented Directed Acyclic Graphs* (DODAG), hanya akan ada satu DODAG per sink. [5]



Gambar 1 Topologi DAG



Gambar 2 Topologi DODAG

Dalam menentukan rute, RPL mengunakan 2 protokol yaitu[6]:

- Protokol vektor jarak
  Protokol ini memanipulasi vektor dalam bentuk array yang berisikan informasi jarak ke node lain dan arah dari transmisi tersebut dalam jaringan. Node sink akan mendapatkan informasi dari node yang lainnya tentang bentuk dari topologi jaringan secara berkala.
  Protokol ini bekerja dengan menghitung jarak antara node dan menyimpan arah dari transmisi yang diterima antar node yang terjadi sebelum mentrasmisikan data ke node yang lain.

- Protokol *Source Routing*
  Mengizinkan penerima dari paket untuk secara sebagian atau secara keseluruhan menentukan rute dari paket yang diterima pada jaringan. Untuk dapat menentukan rutenya, protokol ini juga dapat menentukan semua kemungkinan dari perutean yang mungkin terjadi dan menentukan perutean yang paling menguntungkan.
  1) RPL *Identification*
     RPL menggunakan empat variabel untuk mengidentifikasi topologi :
     - RPLInstanceID
       RPLInstanceID mengidentifikasi satu set atau lebih *Destination Oriented DAGs* (DODAGs). Sebuah jaringan mungkin memiliki beberapa RPLInstanceIDs, masing – masingnya mendefinisikan satu set DODAG yang independen, yang dapat dioptimalkan untuk *Objective Function* (OF) dan/atau aplikasi yang berbeda. Himpunan DODAG yang diidentifikasi oleh RPLInstanceID disebut *RPL Instance*. Semua DODAG dalam *RPL Instance* yang sama

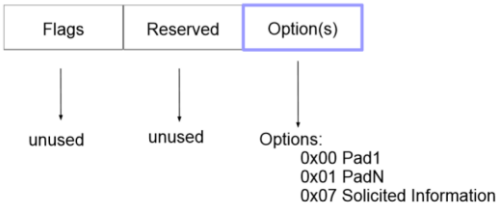
menggunakan *Objective Function* (OF) yang sama.

- DODAGID
  Ruang lingkup DODAGID adalah *RPL Instance*. Kombinasi RPLInstanceID dan DODAGID secara unik mengidentifikasi DODAG tunggal dalam jaringan. *RPL Instance* dapat memiliki beberapa DODAG, yang masing-masing memiliki DODAGID unik.

- Nomor versi dari DODAG
  DODAG kadang-kadang direkonstruksi dari *root* dari DODAG, dengan menambah nomor versi dari DODAG. Kombinasi RPLInstanceID, DODAGID, dan nomor versi DODAG secara unik mengidentifikasi versi DODAG.

- *Rank*
  Lingkup *Rank* adalah versi DODAG. Peringkat menetapkan urutan parsial atas versi DODAG, mendefinisikan posisi simpul individu sehubungan dengan *root* DODAG

2) Pesan Kontrol
   Ada empat jenis pesan kontrol:
   - *DODAG Information Solicitation* (DIS)
     Digunakan untuk mencari DIO dari simpul RPL.



Gambar 3 Struktur Data DIS

   - *DODAG Information Object* (DIO)
     DIO adalah pembawa informasi mengenai *RPL Instance* beserta konfigurasinya. Node pada RPL mengirimkan DIO mengunakan *Trickle Timer*. Prinsip kerja *Trickle Timer* adalah sebuah node mentransmisikan data kecuali jika mendengar beberapa transmisi lain yang datanya menunjukkan transmisinya sendiri berlebihan.

   - *Destination Advertisement Object* (DAO)
     Digunakan untuk menyebarkan informasi mengenai tujuan ke node ke atas menuju root. Dalam mode menyimpan, pesan DAO telah mengetahui dengan jelas posisi/lokasi node yang akan menerima pesan, pesan tersebut akan diterima oleh "anak" yang dikirim oleh "orangtua". Dalam mode tidak menyimpan, pesan DAO akan langsung menuju ke root DODAG. Pesan DAO dapat secara opsional, atas permintaan atau kesalahan eksplisit, diakui oleh tujuannya dengan pesan *Destination Advertisement Object Acknowledgement* (DAO-ACK) kembali ke pengirim DAO.

   - *Destination Advertisement Object Acknowledgement* (DAO-ACK)
     DAO-ACK digunakan dalam komunikasi unicast yang dilakukan oleh DAO "orangtua" atau DODAG *root* untuk menanggapi pesan unicast. Pesan CC digunakan untuk memeriksa penghitung pesan yang aman dan mengeluarkan respons tantangan.

3) Fungsi obyek
Fungsi obyek berfungsi memandu dan mengarahkan node RPL dalam membangun dan mengoptimalkan rute dalam *instance* RPL. Fungsi obyek juga mendefinisikan bagaimana setiap node harus menerjemahkan metrik dan kendala yang ditentukan dalam membentuk rute. Metrik perutean adalah nilai kuantitatif yang digunakan untuk mengukur jalur yang dilalui. Metrik dapat berupa metrik tautan atau metrik simpul. Metrik tautan digunakan untuk mengukur kualitas tautan yang ada di antara simpul, sedangkan metrik simpul adalah nilai kuantitatif dari properti simpul. Metrik ini biasanya aditif.
RPL mendukung dua fungsi obyektif berdasarkan jumlah hop metrik dan ETX. Fungsi tujuan nol (OF0) menggunakan metrik, jumlah hop, peringkat minimum dengan fungsi tujuan histeresis menggunakan penghitungan transmisi yang diharapkan (ETX).

4) Rank Calculation
Beberapa metrik digunakan untuk menetapkan peringkat dan lebih suka memilih "orangtua" berdasarkan pada peringkat. Setiap node bergerak antara menjadi simpul dan *"orangtua"* tergantung pada peringkat yang dimilikinya. *Rank* adalah bilangan bulat 16bit yang menunjukkan peringkat dari node dan mempengaruhi pesan kontrol DIO. Ini adalah representasi skalar dari lokasi node dalam DAG. *Rank* digunakan untuk menghindari *loop* serta untuk mendeteksi *loop*. *Rank* tidak metrik yang berisi jalan yang dilalui dan monoton meningkat ketika node menjauh dari node *root*. Ketika node menjauh dari *root* , maka peringkatnya meningkat.[7]

$$\mathrm{DAG}Rank = \mathrm{floor}\left(\frac{Rank}{\mathrm{MinHop}Rank\mathrm{Increase}}\right)$$

Persamaan 1. Perhitungan DAG*Rank*

MinHop*Rank*Increase menentukan jumlah maksimum hop. Node memeriksa peringkat dari *"orangtua"* dengan node tetangga. Node apapun yang memiliki peringkat terendah menjadi *"orangtua"* dari node tersebut. Jika keduanya sama maka tidak ada perubahan yang dibuat. *Rank* digunakan untuk menghindari *loop* dan metrik perutean digunakan untuk menemukan jalur terpendek antara node. Ketika ada beberapa *root*, node dengan *rank* terkecil dipilih sebagai *"orangtua"*.[7]

Fungsi tujuan nol menggunakan *hop count* (jumlah lompat yang dilakukan) sebagai *routing metric* untuk menentukan peringkat dari node. Setiap node diberi peringkat berdasarkan perhitungan dibuat dengan hop.

$$\mathrm{R(N)} = \mathrm{R(P)} + Rank_{\mathrm{Increase}}$$

Persamaan 2. Perhitungan DAG*Rank*

Keterangan:
R (N) = *Rank* node
R (P) = *Rank* dari Node *"orangtua"*.

Untuk mencari $Rank_{\mathrm{Increase}}$ mengunakan persamaan ,

$$Rank_{\mathrm{Increase}} = (\mathrm{Rf} * \mathrm{Sp} * \mathrm{Sr}) * \mathrm{MinHop}Rank\mathrm{Increase}$$

Persamaan 3. Perhitungan $Rank_{\mathrm{Increase}}$

Keterangan :
Rf = faktor *rank*
Sp = langkah dari *rank*
Sr =bentangan *rank*

Di *minimum rank with hysteresis objective function* (MRHOF) , ETX digunakan sebagai *routing metric* dan digunakan juga untuk perhitungan jalur metrik dan penentuan peringkat tersebut. Ada sedikit pendekatan yang berbeda dari OF0 yang digunakan di sini dalam perkiraan peringkat untuk node dan *"orangtua"* yang dipilih. Node dengan peringkat yang lebih rendah tidak segera terpilih sebagai "orangtua", jangan sampai menciptakan *churn* dalam jaringan. Sedangkan ambang batas ditetapkan dan jika peringkat kurang dari ambang batas yang ditetapkan maka orangtua antar dua node akan bertukar. Jika tidak, simpul terus memiliki orangtua sendiri terlepas dari *"orangtua"* yang tersedia dengan pangkat lebih rendah.[7]

C. 6LowPAN

6LoWPAN (*IPv6 over Low Power Wireless Personal Area Network)* , ide dasarnya adalah menggunakan IP untuk mengkoneksikan perangkat-perangkat kecil berdaya rendah yang berada pada cakupan jaringan nirkabel. WPAN *(Wireless Personal Area Network*) sendiri adalah jaringan yang digunakan untuk menghubungkan perangkat - perangkat pada area yang spesifik dimana jaringan tersebut bersifat nirkabel. Ipv6 digunakan untuk meningkatkan interoperabilitas dari protokol ini sehingga dapat menyesuaikan diri dengan revolusi jaringan yang heterogeneous dan IoT.[1]

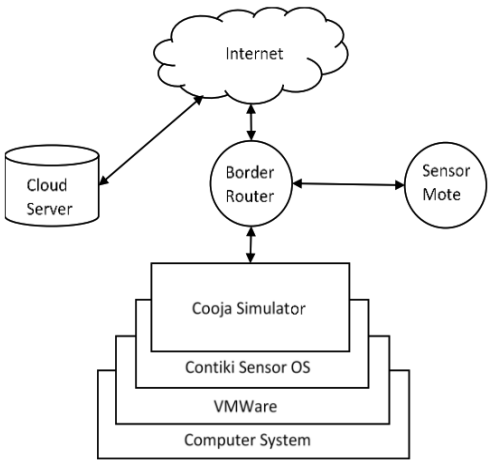
Tabel 1. Frame paket

IEEE 802.15.4 Low-Rate Wireless PAN	
Ipv6 over Low power Wireless Personal Area Network	
Internet Protocol Version 6	
Data	Internet Control Message v6

D. Cooja Simulator

Cooja adalah simulator jaringan yang dirancang untuk mensimulasikan jaringan sensor dengan sistem operasi *contiki*. Ini adalah simulator berbasis Java tetapi memungkinkan node sensor dapat ditulis dalam bahasa *C*.[7] Sistem operasi contiki bersifat *open source* , berkerja pada lingkungan yang membutuhkan daya rendah, biaya rendah untuk mikrokontroller, mempunyai *toolbox* untuk membangun sistem jaringan yang kompleks, tidak perlu membuat code untuk node pada jaringan karena sudah ada di cooja.
Pemilihan simulator cooja tidak terlepas dari kelebihan – kelebihan yang dimilikinya[7] :

- Contiki bekerja pada lingkungan daya yang rendah dan untuk bekerja pada lingkungan tersebut , contiki menyediakan standar internet. Yang mendukung Ipv4 dan Ipv6 , dan juga mendukung standar nirkabel untuk daya rendah seperti 6LowPAN, CoaP, RPL.
- Pengembangan contiki lebih cepat dan lebih mudah untuk dikembangkan dikarenakan program pada contiki mengunakan bahasa C
- Contiki dapat dijalankan pada kebanyakan perangkat nirkabel berdaya rendah yang banyak dijual dipasaran.
- Contiki dikembangkan oleh banyak developer from atmel, cisco, dan lain – lain.
- Contiki bersifat perangkat lunak *open source* , oleh karena itu contiki dapat digunakan secara bebas untuk tujuan komersil ataupun untuk penelitian.
- Contiki juga memiliki forum komunitas yang aktif untuk mengembangkan contiki atau sekedar diskusi tentang penggunaan *software* tersebut.



Gambar 4 Simulator Cooja dan Struktur Sensor

E. Wireshark

Wireshark adalah salah satu penganalisa jaringan *open source* yang paling bagus dalam menganalisa paket dan tersedia di publik. Ini digunakan untuk menangkap lalu lintas jaringan dan untuk memeriksa dengan cermat apa yang terjadi di jaringan. Wireshark memiliki GUI yang mudah digunakan dan dapat dikonfigurasi dengan banyak fitur. Wireshark dapat mendekode lebih dari 400 protokol dan mendukung lebih dari 750 protokol. Wireshark juga dapat berjalan di lebih dari 20 platform seperti unix, windows dan mac OS. Wireshark memiliki kemampuan untuk menangkap

jaringan sekaligus membaca file yang ditangkap. Wireshark punya kapasitas tampilan filter yang banyak. Dengan semua fitur ini, wireshark berdiri tegak di antara rekan-rekannya seperti WinDump, EtherPeek, Tepdump, Snoop, Snort, Dsniff, Ettercap, Packetlyzer, MacSniffer dan sebagainya. File pcap berisi rincian lalu lintas yang ditangkap dan file pcap ditangkap oleh simulator Cooja. Analisis file pcap ini menyediakan sejumlah besar informasi tentang jaringan, tautan, simpul dan paket-paket.[1]

III. METODE PENELITIAN

Untuk membuat simulasi ini seperti yang diharapkan, untuk itu dibuat pengaturan – pengaturan pada simulator cooja agar data yang didapat lebih mudah untuk dianalisa.

Tabel 2. Konfigurasi dari simulator cooja

Parameter	Nilai
Fungsi Obyek	OF0
Jumlah Node	51
Topologi	Random
Rasio Transmisi	100%
Rasio Menerima	100%
Jarak Transmisi	100 meter
Jarak Mnerima	120 meter
Waktu simulasi	6 menit
Luas daerah topologi	40.000 m^2
Lama Simulasi	10 menit
Wireless Channel	UDGM : *Distance Loss*

A. *Expected Transmission Count (ETX)*

Perhitungan transmisi yang diharapkan adalah jumlah yang menentukan jumlah transmisi paket sebuah simpul berharap dari node ke tujuan dengan sukses. ETX adalah nilai diskrit yang dihitung sebagai berikut ini:

$$ETX = \frac{1}{Df * Dr}$$

Persamaan 4. Perhitungan ETX

Keterangan :
Df = probabilitas suatu paket diterima oleh tetangga
Dr = probabilitas ack paket berhasil diterima

ETX adalah salah satu langkah untuk menentukan keandalan transmisi dalam jaringan. Semakin rendah ETX, semakin baik keandalan tautan dan kualitas tautan. ETX juga tidak boleh melebihi batas yang ditentukan.[5]

B. *Throughput*

RPL menganggap *Throughput* sebagai salah satu metrik perutean. *Throughput* dihitung dari awal hingga akhir traversal paket melalui tautan. Tanpa masuk ke detail kompleks definisi, kita dapat secara abstrak mendefinisikan *Throughput* sebagai produk dari jumlah paket, ukuran paket dan bilangan interger 8 untuk mengubah byte menjadi bit, (total besar data yang dikirim) dibagi dengan total waktu simulasi dalam hitungan detik. *Throughput* tergantung pada beban kerja lalu lintas jaringan.[5]

$$\text{Throughput} = \frac{\text{Total dari besar data yang dikirim}}{\text{Total durasi dari simulasi}}$$

Persamaan 5. Perhitungan Throughput

C. *Konsumsi Energi*

Masalah kritis utama dalam jaringan daya rendah dan *lossy* adalah konsumsi energi. Mereka bertenaga rendah dan konservasi energi dan penggunaan energi yang efisien sangat penting untuk itu. Ketika kita mengatakan energi tidak hanya energi yang dikonsumsi diperhitungkan tetapi juga energi sisa yang tetap dalam sumber energi. Nilai daya dan energi sangat tergantung pada biaya pengiriman dan penerimaan paket dalam jaringan. Kami menggunakan parameter tunggal di Contiki Cooja untuk mengukur konsumsi energi dan energi yang tersisa dalam sumber energi bertenaga baterai. Untuk baterai kita dapat menghitung waktu hidup rata-rata, berdasarkan penggunaan konsumsi energinya.

$$\text{Ee} = \frac{\text{Eb}}{\text{Eo}\frac{\text{T} - \text{t}}{\text{T}}}$$

Persamaan 6. Perhitungan Konsumsi Energi

Keterangan:
Ee = energi Tersisa
Eb = energi dalam baterai
Eo = energi awal
T = total waktu hidup
t = total waktu hidup yang telah berlalu.

Energi yang tersisa mudah dihitung dari berapa energi yang digunakan. Tetapi mungkin ada sumber energi yang lebih besar yang dapat menyimpan lebih banyak energi daripada sumber energi yang sangat kecil.[5].

1. *Duty cycling*

LLN memiliki sumber daya energi yang terbatas dan menjaga radio tetap menyala akan menghabiskan energi. Oleh karena itu, radio dimatikan sebanyak mungkin dan dihidupkan hanya saat diperlukan, untuk menghemat energi. Metode ini disebut *Duty cycling*. Metode ini dapat sangat mengurangi konsumsi energi. Ada berbagai jenis mendengarkan , salah satunya adalah *idle listening*, metode mendengarkan saluran selama itu tetap kosong dan sampai paket itu ditransmisikan. Ini adalah metode *cycling* yang mahal. Ada teknik lain seperti mendengarkan sampel dan penjadwalan. Dalam mendengarkan sampel, saluran secara berkala diperiksa untuk transmisi, dalam penjadwalan hanya pada waktu yang ditentukan. Plugin Powertrace digunakan dalam Contiki untuk melacak konsumsi energi. Ini mengukur energi CPU, energi *Low Power Mode* (LPM), energi *Radio Transmit* dan energi *Radio Listen*. Energi CPU adalah energi total yang digunakan oleh CPU untuk memproses data dan energ LPM adalah energi yang digunakan oleh sebuah simpul ketika berada dalam mode hemat daya. Baik energi pemancar radio dan energi radio dalam mendengarkan adalah energi yang digunakan oleh node untuk mengirim dan menerima paket. Energi total

yang digunakan adalah jumlah dari keempat jenis ini. Parameter-parameter ini diukur secara internal oleh Contiki pada saat dijalankan dan ditampilkan secara bersamaan[5].

D. *Network Convergence Time*

Pembentukan topologi adalah fungsi routing yang penting dari RPL sebelum memulai pengiriman data. Jaringan perlu diatur terlebih dahulu sebelum mulai mentransmisikan data, oleh karena itu waktu pengaturan jaringan sangat penting dalam suatu jaringan. Struktur DODAG dibentuk oleh RPL, dengan mengirim pesan kontrol dari root. Sampai akhir pembentukan topologi atau konstruksi DODAG, banyak pesan kontrol akan dikirim melalui jaringan. Waktu konvergensi adalah durasi total antara pesan kontrol pertama dan pesan kontrol terakhir. Waktu konvergensi yang lebih pendek menjadikan stabilitas jaringan lebih banyak[5].

$$\text{Convergence Time} = \text{TDIO} - \text{FDIO}$$

Keterangan :
TDIO = Waktu terakhir sampai pesan DIO diterima DAG
FDIO = waktu pertama pesan DIO diterima DAG

E. *Packet delivery Ratio (PDR)*

Rasio pengiriman paket suatu jaringan adalah rasio antara jumlah total paket yang diterima oleh suatu simpul dan jumlah total paket yang dikirim ke simpul itu. Nilai PDR dapat diperoleh dengan membagi jumlah paket yang diterima dan jumlah paket yang dikirim. Nilai ini memprediksi keandalan jaringan. Semakin banyak nilai PDR, semakin tinggi keandalan jaringan. Pada saat yang sama, metrik keandalan lainnya ETX berbanding terbalik dengan PDR. Jika nilai PDR tinggi maka nilai ETX secara otomatis akan sangat rendah. Jaringan tidak dapat memiliki waktu konvergensi absolut jika node mobile/lokasi berubah-ubah sepanjang waktu, tetapi hanya dapat menentukan waktu konvergensi awal[5].

$$\text{Rasio Paket terkirim} = \frac{\text{Total paket diterima}}{\text{Total Paket dikirim}} * 100$$

Persamaan 7. Perhitungan Rasio Paket Terkirim

F. *Control Traffic Overhead*

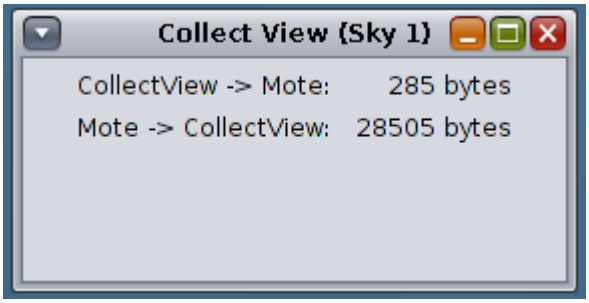
Pesan kontrol seperti DIO, DIS, DAO dihasilkan dalam RPL untuk mengatur jaringan dan untuk mengetahui topologi dari jaringan secara *update*. Pesan kontrol ini mutlak diperlukan untuk pembuatan DODAG. *Control traffic overhead* adalah jumlah total semua jenis pesan kontrol dalam jaringan. Efisiensi protokol perutean bergantung pada pengontrolan jumlah pesan-pesan ini dengan mengingat sumber energi yang terbatas di IoT. Pada saat yang sama pengurangan pesan kontrol menjadi menantang jika jaringan berada pada kondisi yang memungkinkan topologi dapat berubah – ubah. RPL memanfaatkan algoritma trickle untuk mengurangi *control traffic overhead*[5].

$$\text{Control Traffic Overhead} = \sum_{k=1}^{m} \text{DIO(k)} + \sum_{k=1}^{n} \text{DIS(k)} + \sum_{k=1}^{o} \text{DAO(k)}$$

Persamaan 8. Perhitungan jumlah pesan kontrol

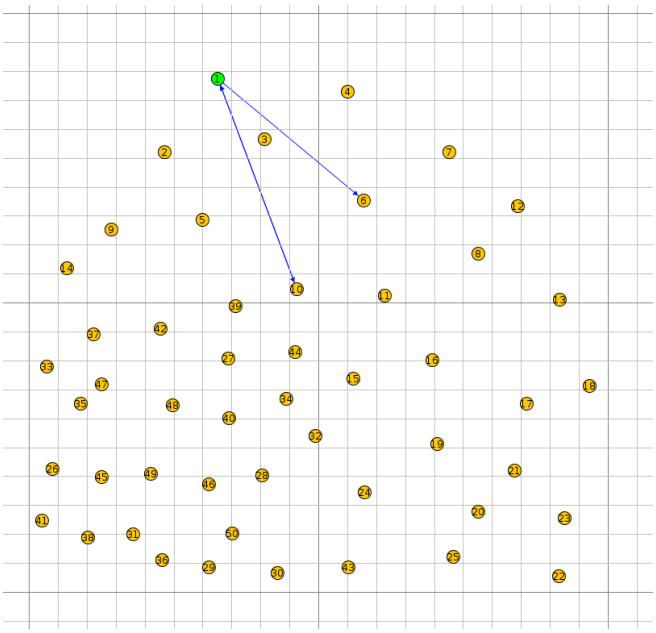
IV. HASIL DAN ANALISA

Cooja simulator memiliki GUI user-friendly dan mudah untuk dikonfigurasi.



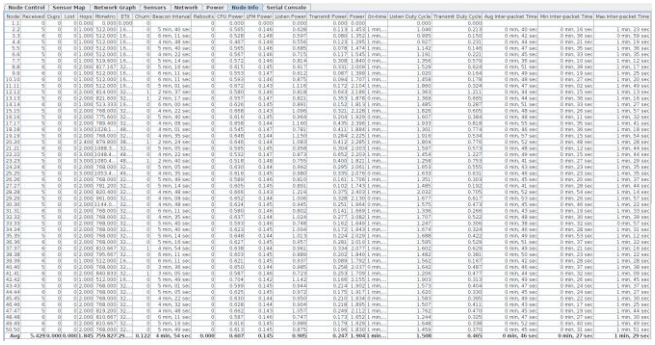
Gambar 5 Collect View pada Simulator Cooja

Topologi yang digunakan pada penelitian ini mengunakan topologi yang setiap node disusun secara acak dengan ketentuan tidak melewati luas daerah yang sudah ditentukan sebesar 400 m^2 .



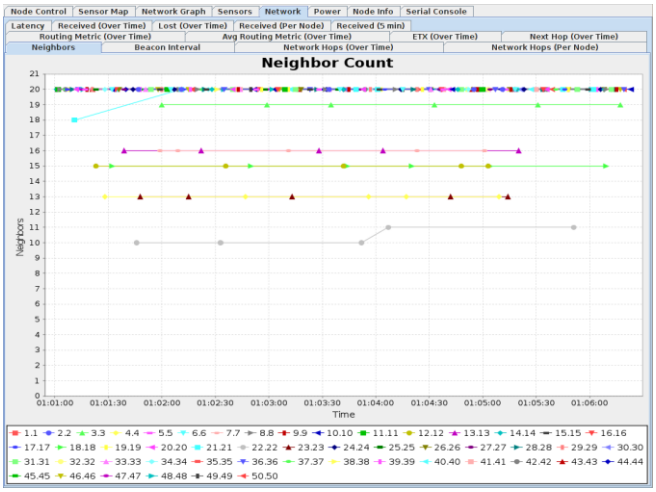
Gambar 6 Desian topologi pada simulator cooja

Pada simulasi ini didapat informasi – informasi berupa ETX, penggunaan energy , paket kontrol pada RPL, rasio pengiriman paket dan lain – lain untuk menganalisa 6LowPAN.



Gambar 7 Informasi Node *pada simulator cooja*

Pada simulasi ini juga dapat diketahui jumlah dari tetangga setiap sensor , hal ini mempengaruhi jumlah paket yang hilang dan penggunaan energi pada node.

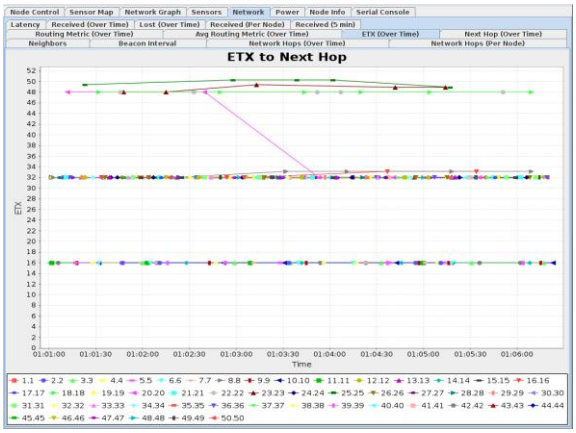


Gambar 8 jumlah tetangga node

Pada gambar dapat dilihat bahwa sensor node yang memiliki tetangga 20 cukup banyak , dikarenakan jarak transmisi dan jarak menerima dari node besar dan area topologi yang kecil untuk ukuran 51 node. Hal ini juga mempengaruhi pengunaan energi pada node dikarenakan dibutuhkan konsumsi baterai yang lebih besar untuk mentransmisikan paket dengan jangkauan yang lebih jauh.

A. *Pengukuran ETX*

Collect view pada simulasi Cooja menyediakan data secara online dan diperbarui dari nilai-nilai ETX. Setelah awal simulasi, *Collect view* dapat dibuka. Informasi simpul dari *Collect view* akan memberikan nilai ETX secara online. Setelah simulasi pada akhir waktu yang ditetapkan, nilai ETX harus dikumpulkan. Kita bisa mengambil nilai-nilai, secara individual untuk setiap node atau mengambil rata-rata nilai ETX dari seluruh jaringan. Nilai ETX akan bervariasi sesuai dengan durasi simulasi RPL. Semakin rendah nilai ETX lebih baik keandalan protokol.



Gambar 9 Nilai ETX pada Collect View

B. *Pengukuran Throughput*

Throughput dapat dihitung dengan menganalisis file pcap. File dari pcap yang diambil dari cooja dapat diproses oleh aplikasi wireshark. Nilai *Throughput* yang lebih besar , menandakan bahwa jaringan yang disimulasikan memiliki transmisi data yang lebih baik.

Statistics

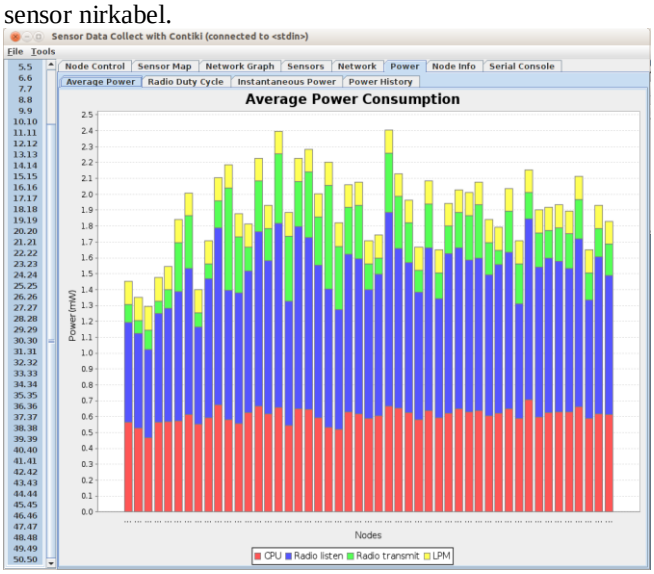
Measurement	Captured	Displayed	Marked
Packets	6092	6092 (100.0%)	2 (0.0%)
Time span, s	378.034	378.034	0.034
Average pps	16.1	16.1	58.8
Average packet size, B	90.5	90.5	97.5
Bytes	548346	548346 (100.0%)	194 (0.0%)
Average bytes/s	1450	1450	5705
Average bits/s	11 k	11 k	45 k

Gambar 10 Informasi besar dari simulasi

$$Throughput = \frac{548346}{378{,}04} = 1450{,}49\ bytes/second$$

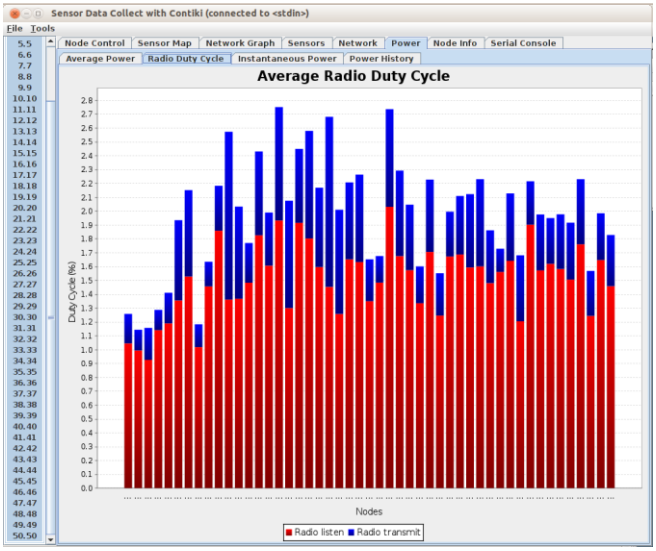
C. *Pengukuran Konsumsi Energi*

Tampilan dari energi yang ditampilkan oleh simulator cooja terdiri dari energi CPU, energi LPM, energi untuk mentransmisikan, energi untuk menerima paket. Pada gambar dimungkinkan untuk melihat 4 energi konsumsi secara individual dan kolektif menggunakan tampilan GUI ini. Semakin rendah konsumsi energi dari sebuah node dan jaringan, semakin baik cocok untuk pengaplikasian jaringan sensor nirkabel.



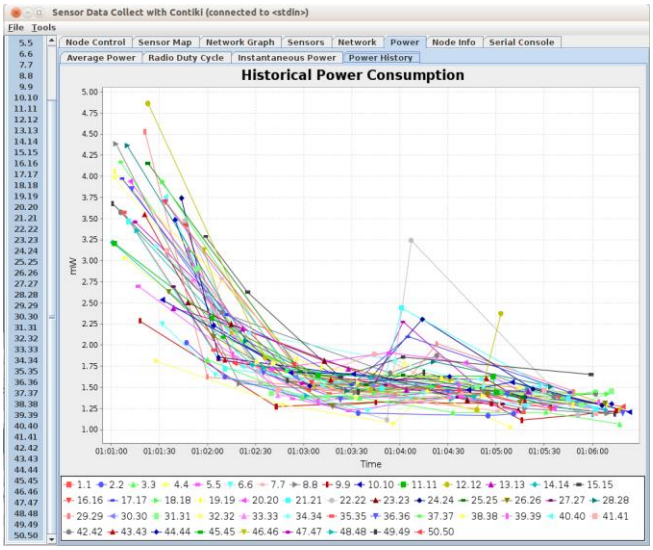
Gambar 11 Nilai Rata-Rata konsumsi energi pada Collect View

Dari gambar dibawah dapat disimpulkan bahwa node lebih banyak mendengarkan daripada mentransmisikan paket ke node lain. Hal ini dikarenakan topologi dari simulasi memiliki area jaringan yang kecil dan kemampuan untuk menstransmisikan paket yang berada pada jarak yang jauh , yang menyebabkan node yang berada pada jarak yang jauh pada node yang sedang menstransmisikan paket tersebut dapat menerima paket dan mengolah paket tersebut.



Gambar 12 Nilai rata-rata aktivitas radio pada Collect View

Pada gambar dibawah dapat disimpulkan , pada awal simulasi dibutuhkan energi yang besar dikarenakan node menstramisikan paket dan menerima paket secara bersamaan.

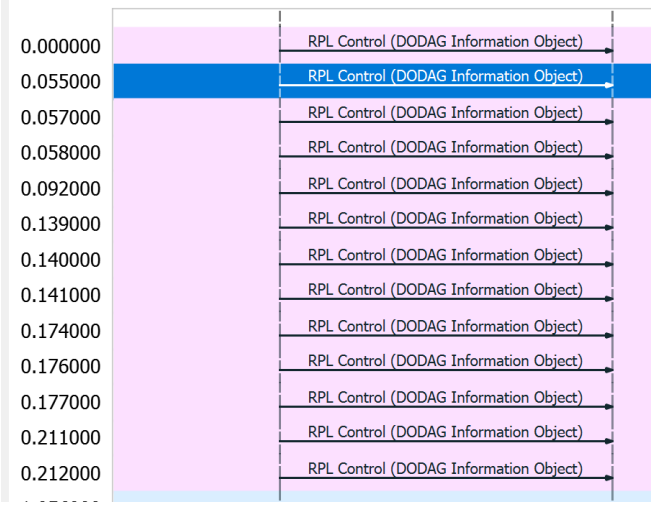


Gambar 13 Nilai sejarah konsumsi energi pada Collect View

Pada gambar diatas dapat kita lihat bahwa node 38 mengkomsumsi daya lebih rendah. Hal ini menunjukkan bahwa node ini lebih sedikit mengirimkan data atau pun menerima akibat dari pemilihan rute oleh metode RPL. Sebaliknya pada node 22, banyak daya yang digunakan oleh karena node tersebut berada pada jarak yang paling jauh, sehingga node tersebut lebih aktif daripada yang lain.

D. Pengukuran Waktu Konvergensi

File pcap yang ditangkap di Cooja dievaluasi oleh penganalisa jaringan wireshark. Untuk mendapatkan waktu konvergensi , dibutuhkan data dari DIO pertama dan terakhir pada node untuk berkomunikasi. Output mote memberikan rincian waktu di mana pesan dikirim, nomor ID mote dan rincian transmisi. Sangat mudah untuk menemukan DIO terakhir yang bergabung dengan DAG dalam file output mote itu. Waktu konvergensi perlu minimum untuk jaringan untuk memberikan stabilitas yang lebih baik.



Gambar 14 Rentang Waktu Pengiriman Pesan DIO

$$Waktu\ Konvergensi = 0{,}212 - 0 = 0{,}212\ sekon$$

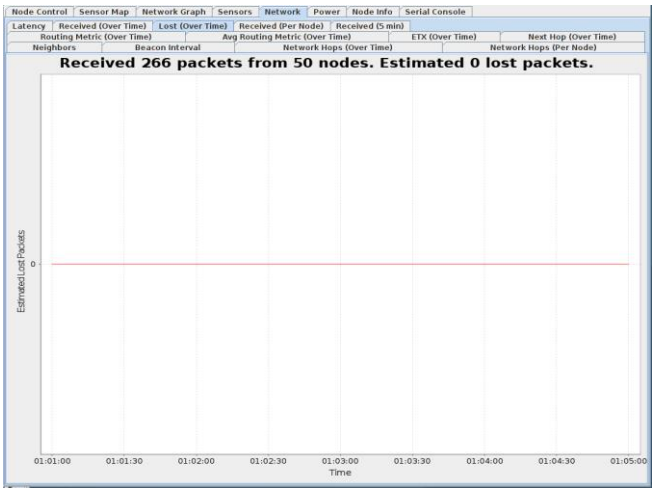
E. Pengukuran PDR

Packet Delivery Ratio (PDR) dapat dihitung dengan menggunakan file pcap dalam jaringan analyzer wireshark. Paket pengiriman total dan paket yang diterima dapat dilihat dari *collect view* pada simulasi cooja. Rasio pengiriman paket adalah salah satu faktor utama dalam menentukan menerima jaringan. Sebuah jaringan dengan cakupan transmisi yang baik akan menyediakan lebih dari 90% dari PDR. Nilai jaringan mempengaruhi nilai PDR.

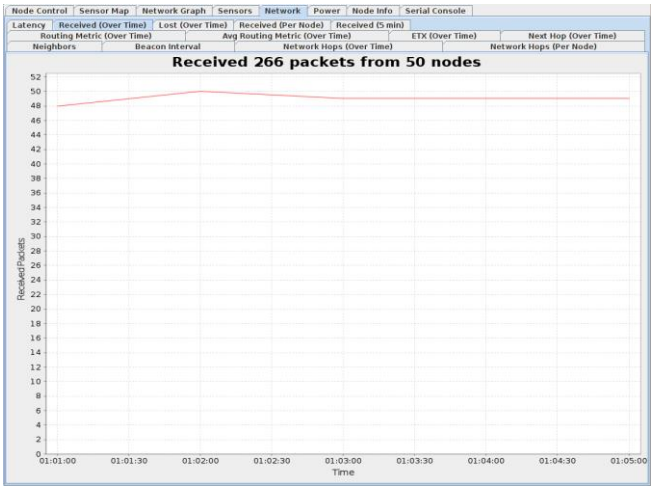
$$\text{Rasio Paket terkirim} = \frac{\text{Total paket diterima}}{\text{Total Paket dikirim}} * 100$$

$$\text{Rasio Paket terkirim} = \frac{266}{266} * 100$$

$$\text{Rasio Paket terkirim} = 100\ \%$$

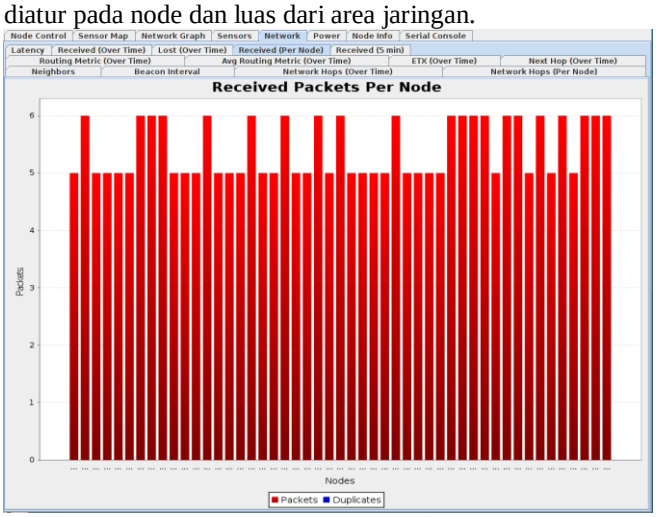


Gambar 15 Jumlah Pesan Hilang pada Simulasi



Gambar 16 Jumlah Pesan Diterima pada Simulasi

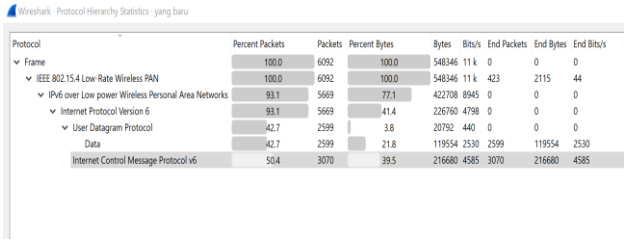
Pada gambar dapat disimpulkan bahwa semua node menerima paket lebih dari 5 paket. Hal ini tidak terlepas dari konfigurasi dari jarak transmisi dan jarak menerima yang diatur pada node dan luas dari area jaringan.



Gambar 17 Jumlah Pesan Diterima setiap node

F. Pengukuran *Traffic Control Overhead*

Untuk mendapatkan *traffic control overhead* , digunakan aplikasi wireshark untuk mendapatkan lewat file pcap. Untuk mendapatkan jumlah dari kontrol pesan yang digunakan protokol rpl dapat dilihat dengan fitur protokol hierrakki pada wireshark.Paket kontrol tersebut adalah DIO, DIS dan DAO. Jumlah dari semua itu merupakan total *traffic control overhead* jaringan. *Traffic control overhead* harus dikurangi sebanyak mungkin, karena lebih banyak lalu lintas akan menguras baterai dan sangat mempengaruhi perangkat bertenaga rendah di jaringan sensor nirkabel.



Gambar 18 protokol hierrakki

Dari gambar tersebut dapat diketahui , paket yang control pada rpl sebanyak 3070 , hampir setengah dari jumlah paket pada dikirimkan. Hal ini disebabkan posisi dari node yang saling berdekatan dengan kemampuan node yang mampu mentransmisikan dan menerima paket dari tetangga yang cukup jauh. Dengan semakin banyak terdeteksinya tetangga node , maka akan semakin banyak terjadinya proses penentuan rute yang membutuhkan pesan kontrol. Hal ini mengakibatkan penggunaan energi yang dimiliki node akan cepat habis dikarenakan proses transmisi dan menerima pesan kontrol dari node lain.

V. KESIMPULAN

Pada simulasi ini dapat disimpulkan bahwa

- Konfigurasi dari simulator cooja sangat berpengaruh ke hasil yang akan didapat seperti topologi yang digunakan, jarak transmisi dan menerima pada node, posisi dari node, dan lain – lain.
- ETX

  Nilai ETX akan bervariasi sesuai dengan durasi simulasi RPL. Semakin rendah nilai ETX lebih baik keandalan protokol.
- *Throughput*

  Pada simulasi ini , kecepatan transfer yang didapat 1450,49 bit/sekon.
- Konsumsi energi

  Konsumsi energi pada simulasi ini , lebih banyak dihabiskan untuk mendengarkan paket yang dikirim dan energi untuk memproses paket tersebut.
- Waktu konvergensi

  Pada simulasi ini, waktu konvergensi yang didapat 0,212 sekon . Waktu konvergensi cukup singkat dikarenakan topologi dari antar node yang berdekatan.

- Rasio paket yang diterima

  Pada simulasi ini, rasio paket yang diterima 100% dikarenakan topologi dan jarak antar node yang tidak memungkinkan terjadi adanya kehilangan paket antar node.

- Jumlah paket kontrol

  Pada simulasi ini , jumlah paket kontrol hampir setengah dari paket yang ditransmisikan oleh node. Dikarenakan topologi dan jarak transmisi dan menerima dari node yang jauh.

REFERENCES

[1] W. Darmawan, S. R. Akbar dan M. Data, "Analisis Performa Protokol 6LoWPAN pada Jaringan Sensor Nirkabel Menggunakan Cooja Simulator," *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, pp. 2963-2971, 2018.

[2] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struick, J. Vasseur dan R. Alexander, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Network," Internet Engineering Task Force, 2012.

[3] U. A. Arrozaqi, T. B. Santoso dan P. Kristalina, "SIMULASI ROUTING PROTOKOL PADA JARINGAN SENSOR NIRKABEL DENGAN MENGGUNAKAN METODE CLUSTER BASED".

[4] I. N. R. Hendrawan, "Analisis Kinerja Protokol Routing RPL pada Simulator Cooja," *JURNAL SISTEM DAN INFORMATIKA*, vol. 12, no. 2, pp. 9-18, 2018.

[5] J. A. Charles dan P. Kalavathi, "QoS Measurement of RPL using Cooja Simulator and Wireshark Network Analyser," *International Journal of Computer Sciences and Engineering*, vol. 6, no. 4, pp. 283-291, 2018.

[6] T. Mehmood, "COOJA Network Simulator: Exploring the Infinite Possible Ways to Compute the Performance Metrics of IOT Based Smart Devices to Understand the Working of IOT Based Compression & Routing Protocols".

[7] L. B. Saad, C. Chauvenet dan B. Tourancheau, "IPv6 Routing Protocol for Low Power and Lossy Sensor Networks Simulation Studies," *Sensors & Transducers Journal*, vol. 14, no. 2, pp. 79-92, 2012.

[8] L. B. Saad, C. Chauvenet dan B. Tourancheau, "Simulation of the RPL Routing Protocol for IPv6 Sensor Networks : two cases studies," 2011.

[9] J. Schonwalder, *Internet of Things: 802.15.4, 6LoWPAN, RPL, COAP*, 2010.

[10] "Ipv6now," IPv6 Now Pty Ltd, [Online]. Available: http://ipv6now.com.au/whyipv6.php. [Diakses 30 12 2018].

[11] M. Richardson dan I. Robles, "RPL- Routing over Low Power and Lossy Networks," IETF.

[12] "http://www.contiki-os.org/," [Online]. Available: http://www.contiki-os.org/. [Diakses 13 1 2019].

[13] C. Thomson, I. Romdhani, M. Qasem dan A. Y. Al-Dubai, Cooja Simulator Manual, Edinburgh Napier University, 2016.

[14] Z. Shelby dan C. Bormann, 6LoWpan : The Wireless Embedded Internet, Wiley, 2009.