ANALISA PERBANDINGAN PERFORMANSI PROTOKOL *ROUTING* OLSR, BATMAN, DAN BABEL PADA JARINGAN MESH

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
MOCH. ARIUS EKO CAHYONO
NIM: 115090600111023

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2016

PENGESAHAN

ANALISA PERBANDINGAN PERFORMANSI PROTOKOL ROUTING OLSR, BATMAN, DAN BABEL PADA JARINGAN MESH

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
MOCH. ARIUS EKO CAHYONO
NIM: 115090600111023

Skripsi ini telah diuji dan dinyatakan lulus pada
24 November 2016
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Adhitya Bhawiyugha, S.Kom, M.S	Ir. Heru Nurwarsito, M.Kom
NIP: 201405 890720 1 001	NIP: 196504021990021001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D

NIP: 19710518 200312 1 001

ORISINALITAS SKRIPSI

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

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

Malang, Agustus 2016

Moch. Arius Eko Cahyono
Nim: 115090600111023

Moch. Arius Eko Cahyono

NIM 115090600111023

KATA PENGANTAR

Puji syukur penulis ucapkan kehadirat Allah SWT yang telah melimpahkan segala Rahmat, Karunia, dan Hidayah-Nya sehingga penulis dapat menyelesaikan skripsi dengan judul "Analisa Perbandingan Performansi Protokol *Routing* OLSR, BATMAN dan Babel pada jaringan Mesh".

Skripsi ini diajukan sebagai salah satu syarat memperoleh gelar Sarjana Komputer di Program Studi Informatika Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya Malang. Atas terselesaikannya skripsi ini, penulis mengucapkan terima kasih kepada beberapa pihak diantaranya:

1. Kedua orang tua penulis yang selalu memberikan doa, motivasi, dukungan moril dan materil sebagai penyemangat dalam menyelesaikan skripsi ini.
2. Bapak Adhitya Bhawiyuga, S.Kom., M.S., M.Eng dan Bapak Ir. Heru Nurwarsito, M.Kom selaku dosen pembimbing I dan pembimbing II yang telah banyak memberikan ilmu, bantuan, bimbingan, dan motivasi dalam penyelesaian skripsi ini.
3. Bapak Tri Astoto Kurniawan, S.T., M.T., Ph.D selaku Ketua Program Studi Informatika Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya.
4. Segenap Bapak dan Ibu dosen yang telah mendidik dan mengajarkan ilmunya kepada penlis selama menempuh pendidikan di Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya Malang.
5. Teman-teman Informatika angkatan 2011 yang selalu mendukung dan berbagi ilmu dari awal perkuliahan sampai tahap akhir penyelesaian skripsi.
6. Segenap staf dan karyawan di Program Teknologi Informasi dan Ilmu Komputer yang telah banyak membantu penulis dalam pelaksanaan penyusunan skripsi ini.

7. Semua pihak yang telah membantu terselesaikannya skripsi ini yang tidak dapat penulis sebutkan satu per satu.

Penulis menyadari bahwa skripsi ini masih memiliki kekurangan dan jauh dari sempurna, karena keterbatasan materi dan pengetahuan yang dimiliki penulis. Maka, saran dan kritik yang membangun dari semua pihak sangat diharapkan demi penyempurnaan selanjutnya. Semoga skripsi ini dapat bermanfaat dan berguna bagi semua pihak, baik penulis maupun pembaca, dan semoga Allah SWT meridhoi dan dicatat sebagai amalan ibadah, Amin.

Malang, 25 Juni 2015

Moch. Arius Eko Cahyono
ariuscahyono@yahoo.co.id

ABSTRAK

Moch. Arius Eko Cahyono. 2016. Analisa Perbandingan Performansi Protokol *Routing* OLSR, BATMAN dan Babel Pada Jaringan Mesh. Program Studi Ilmu Komputer, Fakultas Ilmu Komputer Universitas Brawijaya.

Pembimbing: Adhitya Bhawiyuga, S.Kom., M.S. dan Ir. Heru Nurwarsito, M.Kom

WMN (Wireless Mesh Network) merupakan salah satu teknologi pada jaringan komputer yang beroperasi pada standar IEEE 802.11 dan terdiri dari router, client, sensor serta perangkat selular. Untuk mengatur lalulintas data pada WMN, diperlukan protokol *Routing*. Protokol *Routing* dibagi menjadi dua jenis, proaktif dan reaktif *Routing.* Proaktif routing lebih cocok untuk diterapkan dalam WMN. Dengan demikian, dilakukan perbandingan performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan *Mesh*. Penelitian ini dilkakukan dengan membandingkan performansi QOS(*Throughput, Jitter, Packet Loss*) antara protokol *Routing* OLSR, BATMAN dan Babel pada jaringan *Mesh*. Hasil penelitian berupa data nilai rata-rata performansi terbaik diantara protokol *Routing* OLSR, BATMAN dan Babel.

Kata Kunci: Analisis Performansi, *Mesh*, OLSR, BATMAN, Babel

ABSTRACT

Moch. Arius Eko Cahyono. 2016. *' Comparative Performance Analysis of Routing Protocols OLSR , BATMAN and Babel On Mesh Network '. Study Program of Computer Science , Faculty of Computer Science University of Brawijaya .*

***Advisor*: Adhitya Bhawiyuga, S.Kom., M.S. dan Ir. Heru Nurwarsito, M.Kom**

WMN (Wireless Mesh Network) is one of the technology on a computer network that operates on the IEEE 802.11 standard and consists of a router , client , sensors and mobile devices . To adjust the data traffic on the WMN , Routing protocols required . Routing protocols are divided into two types , both proactive and reactive routing . Proactive routing is more suitable to be applied in WMN . Thus , do OLSR Routing protocol performance comparison , BATMAN and Babylon to the Mesh network . This research was conducted by comparing the performance of QOS (Throughput , Jitter , Packet Loss) between OLSR Routing protocols , BATMAN and Babylon to the Mesh network . The results of research in the form of data the average value of the best performance among the OLSR Routing Protocol , BATMAN and Babel .

Keywords : Performance Analysis , Mesh , OLSR , BATMAN , Babel

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1 PENDAHULUAN

1.1 Latar belakang

WMN (*Wireless Mesh Network*) merupakan salah satu teknologi pada jaringan komputer yang beroperasi pada standar IEEE 802.11 dan terdiri dari router, *client*, sensor serta perangkat selular (Ratelinggi, 2015). WMN adalah jaringan *Wireless* menggunakan topologi Mesh dengan konsep hubungan dari setiap AP (*Acces point*) yang ada sehingga membentuk sebuah jaringan fisik. Jika salah satu AP mengalami kerusakan, maka peranannya akan diambil oleh AP lainnya. AP pengganti akan menyediakan koneksi pada jaringan ketika masih berada dalam jangkauan jaringan *Mesh*. Pengatur lalulintas data yang disebut Protokol *Routing* diperlukan dalam pengiriman paket data. Jenis-jenis Protokol *Routing*, yakni *Proactive Routing, Reactive Routing* dan *Hybrid* (DeCristofaro,dkk.,2014). *Proactive Routing* paling cocok diterapkan pada jaringan WMN. Didukung oleh kinerja *Proactive Routing* yang dinamis. Beberapa jenis protokol *Routing Proactive* yang umum digunakan adalah OLSR, BATMAN, dan Babel.

Protokol *Routing* diperlukan untuk menentukan jalur tercepat pada pengiriman paket data. Pada WMN, protokol *Routing* berperan untuk efisiensi komunikasi *multi-hop* antar jarak *node*(Friginal,dkk., 2012). Permasalahannya adalah Protokol *Routing* apa yang paling tepat diaplikasikan pada WMN.

Protokol *Routing Proactive* dapat mengoptimalkan proses pengiriman paket data. Protokol *Routing Proactive* pada WMN dapat melakukan pertukaran kontrol pesan secara berkala dengan *node* lain untuk memenuhi permintaan *update* rute ke semua jarak *node* yang terdapat dalam jaringan(Friginal,dkk., 2012). Terutama ketika terdapat salah satu node terputus dari jaringan mesh, protokol *Routing Proactive* secara otomatis mencari *path* alternatif lainnya.

Dengan adanya permasalahan tersebut, maka diperlukan perbandingan performansi antara protokol *Routing* OLSR, BATMAN dan Babel. Pada penelitian sebelumnya telah dilakukan pembahasan mengenai protokol *Routing* OLSR, BATMAN dan Babel. Protokol OLSR merupakan optimasi dari Protokol *link state* sehingga mewarisi stabilitas algoritma *link state* dengan keuntungan ketersediaan rute *Routing* yang dibutuhkan setiap saat(Clausen, dkk.,2003). Semakin besar dan padat jaringan, maka semakin besar pula optimasi yang dapat dicapai ole OLSR dibandingkan dengan algoritma *link state*(Clausen, dkk.,2003). BATMAN didesain untuk diterapkan pada jaringan *unreliable* seperti *wireless* dan selalu mengalami banyak *packet loss*(Margolang, 2014). Kehandalan BATMAN di dukung oleh kemampuan dalam penentuan jalur *Routing* dengan melakukan pengenalan *node* lain melelui *node* tetangga terdekatnya(Neumann,dkk., 2008). Babel mempunyai dua karakteristik khas untuk mengoptimalkan kinerja relay(Mbolhasan,dkk., 2009). Pertama, menggunakan *history-sensitive* pemilihan rute untuk meminimalkan dampak *route flaps*. Kedua, Babel mengesekusi *update* reaktif dan memaksa untuk meminta informasi ruting ketika mendetaksi kegagalan ruting dari salah satu *node* tetangga. Pengujian performansi ketiga protokol *Routing* akan

dilakukan dalam satu jaringan fisik WMN di gedung C FILKOM UB. Penelitian ini bertujuan untuk mengetahui performansi masing-masing protokol *Routing* pada WMN dan mendapatkan protokol *Routing* dengan performansi paling baik. Performansi yang akan dipertimbangkan adalah Qos(*Throughput, Jitter* dan *Packet Loss*) pada msing-masing protokol *Routing.* Kategori kwalitas *Throughput* dibagi menjadi 4, yakni : sangat bagus jika bernilai 76-100%, bagus jika bernilai 56-75%, sedang jika bernilai 26-50% dan jelek ketika < dari 25%(TIPHON., 1999). Sedangkan presentase perhitungan throughput dapat diketrahui dengan paket data diterima dibagi lama pengamatan(TIPHON., 1999). Kategori kwalitas *Jitter* dibagi dalam 4 kategori dengan pembagian nilai sangat bagus ketika nilai jitter 0 ms, bagus ketika bernilai 0-75 ms, sedang ketika bernilai 75-125 ms dan jelek ketika bernilai 125-225 ms(TIPHON., 1999). Sedangkan pada *Packet Loss* terdapat 4 kategori kwalitas servis. *Packet Loss* akan sangat baik jika bernilai 0%, baik jika bernilai 0≥3%,sedang jika bernilai 3≥15% dan jelek jika bernilai 15≥25%(TIPHON., 1999).

Berdasarkan uraian latar belakang yang telah dipaparkan, dalam penelitian ini penulis menggunakan protokol *Routing* OLSR, BATMAN dan Babel dalam menyelesaikan permasalahan pencarian performansi protokol *Routing* paling sesuai jika diterapkan pada jaringan WMN. Setelah ditemukan protokol *Routing* dengan performansi paling sesuai ketika diterapkan pada jaringan WMN, diharapkan dapat membantu dalam pengiriman paket data dengan cepat, tepat dan akurat. Dari uraian tersebut, penulis mengambil judul "Perbandingan Performansi Protokol Routing OLSR, BATMAN dan Babel pada Jaringan MESH".

1.2 Rumusan masalah

Berdasarkan latar belakang di atas, dapat dirumuskan beberapa masalah sebagai berikut:

1. Bagaimana performansi protokol *Routing* OLSR, BATMAN dan Babel jika diterapkan pada jaringan mesh?
2. Bagaimana memilih dan menentukan protokol *Routing* terbaik diantara OLSR, BATMAN dan Babel yang memiliki performansi paling bagus untuk diterapkan pada jaringan mesh?

1.3 Tujuan

Tujuan umum penelitian ini adalah analisa perbandingan performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan mesh. Penelitian bertempat di Gedung C FILKOM UB. Penelitian dilakukan dengan mengirim paket UDP antar *node* aktif pada jaringan mesh dengan 5 variasi besaran paket data. Hasil pengiriman paket UDP akan diambil nilai rata-rata untuk dilakukan perbandingan performansi dari masing-masing protokol *Routing* OLSR, B.A.T.M.A.N dan Babel pada jaringan mesh.

Tujuan khusus penelitian perbandingan performansi protokol *Routing* OLSR, B.A.T.M.A.N dan Babel pada jaringan mesh meliputi:

1. Mengetahui performansi protokol *Routing* OLSR, B.A.T.M.A.N dan Babel pada jaringan mesh.
2. Melakukan analisa performansi protokol *Routing* OLSR, B.A.T.M.A.N dan Babel mesh untuk mendapatkan protokol *Routing* yang paling sesuai diterapkan pada jaringan mesh.

1.4 Manfaat

Manfaat yang ingin diperoleh dari hasil penelitian ini adalah:

1. Memberikan pandangan solusi dalam memilah algoritma yang paling sesuai untuk diterapkan pada jaringan Mesh.
2. Memberikan wawasan pada bidang ilmu pengetahuan dalam penilaian kinerja protokol *Routing* di jaringan Mesh.

1.5 Batasan masalah

Agar pembahasan tidak melebar dari latar belakang dan terfokus dengan apa yang berkaitan dengan sistem, maka batasan dalam penelitian ini yaitu:

1. Pembahasan terbatas pada analisa tiga protokol *Routing* diantaranya OLSR, BATMAN dan Babel.
2. Terbatas pada jaringan Mesh.
3. Menggunakan Raspberry pi sebagai media pembangun jaringan Mesh.
4. Melakukan pengujian pengiriman paket data dengan skenario yang telah ditentukan.
5. Topologi Mesh dipasang di Gedung C FILKOM.

1.6 Sistematika pembahasan

Penulisan skematik pembahasan dan penyusunan laporan penelitian dapat diuraikan sebagai berikut:

BAB 1 PENDAHULUAN

Pada Bab I Pendahuluan dijelaskan mengenai latar belakang, rumusan masalah, tujuan, manfaat penelitian, batasan penelitian dan sistematika pembahasan dari " Analisa Perbandingan Performansi Protokol *Routing* OLSR, BATMAN dan Babel pada Jaringan Mesh ".

BAB 2 KAJIAN PUSTAKA DAN LANDASAN TEORI

Pada bab ini berisi beberapa dasar teori dan referensi penelitian yang pernah ada yang memiliki tujuan dan analisa yang hampir sama dengan "Analisa Perbandingan Performansi Protokol *Routing* OLSR, BATMAN dan Babel pada Jaringan Mesh".

BAB 3 METODE PENELITIAN

Pada bab ini membahas beberapa langkah kerja yang akan dilakukan dalam penelitian, diantaranya studi literatur, analisis kebutuhan, implementasi dan pengujian performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan Mesh.

BAB 4 LINGKUNGAN PENELITIAN

Bab ini membahas lingkungan penelitian yang digunakan dalam analisa performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan Mesh.

BAB 5 PENGUJIAN DAN ANALISIS

Pada bab pengujian dan analisis ditunjukkan hasil pengujian serta dilakukan analisa terhadap perbandingan performansi dari ketiga protokol *Routing* tersebut sehingga dapat ditarik kesimpulan protokol *Routing* dengan performansi paling paling bagus ketika diterapkan pada jaringan Mesh.

BAB 6 PENUTUP

Bab ini berisi kesimpulan dari pembahasan rumusan masalah yang ada berdasarkan pengujian dan analisis serta berisi saran-saran dan pengembangan yang dapat dilakukan dari "Analisa Perbandingan Performansi Protokol *Routing* OLSR, BATMAN dan Babel pada Jaringan Mesh . "

BAB 2 KAJIAN PUSTAKA DAN LANDASAN TEORI

Bab ini akan menjelaskan tinjauan pustaka yang berisi penelitian-penelitian sebelumnya dengan tujuan atau metode menyerupai penelitian penulis serta berisi tentang dasar teori yang mendukung dalam proses penelitian "Analisa Perbandingan Performansi Protokol *Routing* OLSR, BATMAN dan Babel pada Jaringan Mesh".

2.1 Tinjauan Pustaka

Pada penelitian dengan judul "Pengembangan *Wireless Mesh Network* Berbasis IEEE802.11", setiap *user* pada *Wireless Mesh Network* dapat beroperasi sebagai *host* maupun *router* untuk meneruskan(*forward*) paket menuju *node* lain yang belum termasuk dalam *coverage* dari *gateway*. Jaringan ini biasanya sangat dinamis dan memiliki sifat *self-configuring* dan *self-healing* . (zulham, dkk., 2008).

Topologi jaringan memerlukan protokol *Routing* yang bertanggung jawab untuk melakukan pengiriman data. Keberhasilan pengiriman data sangat dipengaruhi oleh adanya protokol *Routing*. Penelitian yang berjudul "*Mitigating the Impact of Ambient Noise on Wireless Mesh Networks Using Adaptive Link-Quality-based Packet Replication*" melakukan pengujian kualitas jaringan mesh dari efek *ambient noise* dengan menggunakan protokol *Routing Proaktif.* Protokol *Routing Proaktif* mempunyai kinerja yang sejalan dengan konsep WMN, yakni dapat menunjang pemeliharaan kualitas jaringan didalam cakupan Mesh. Kinerja *Routing proaktif* memiliki konsep pertukaran informasi dari masing-masing *node* secara berkala mencakup semua *node* pada jaringan tersebut(Friginal,dkk., 2012).

Berdasarkan beberapa penelitian di atas, penulis memperoleh ide untuk melakukan perbandingan performansi *Protokol Routing Proaktif* OLSR, BATMAN dan Babel. Perbandingan performansi yang dilakukan penulis, diharapkan dapat menentukan protokol *Routing* paling tepat untuk diterapkan pada WMN. Protokol *Routing* diterapkan untuk memelihara kualitas jaringan Mesh sehingga memungkinkan adanya jaminan kelancaran pengiriman paket data.

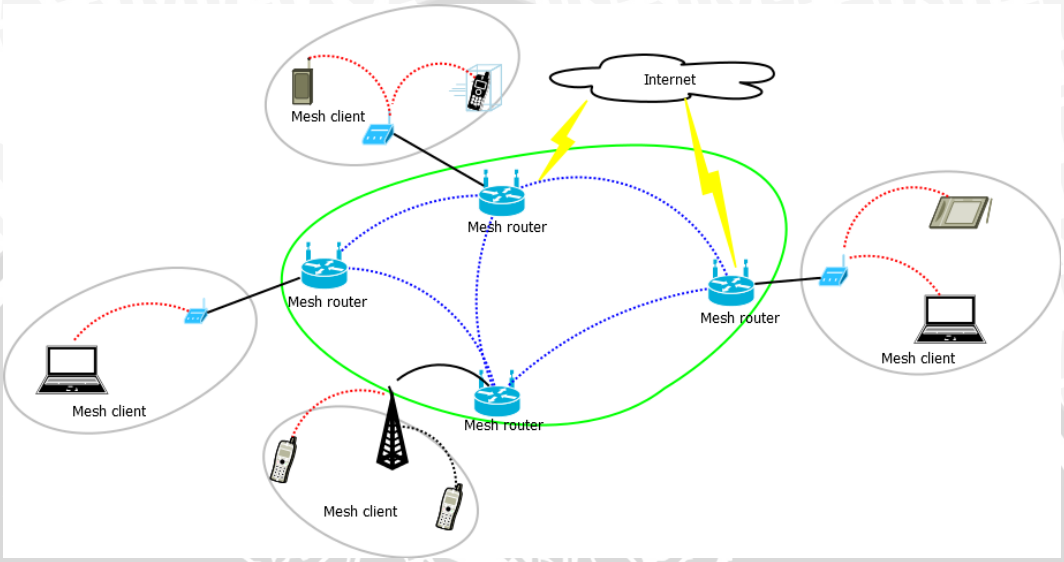
2.2 Dasar Teori

Berdasarkan beberapa informasi yang didapat dari beberapa kajian pustaka, maka dalam "Analisa Perbandingan Performansi Protokol *Routing* OLSR, BATMAN dan Babel pada Jaringan *Mesh*" terdapat beberapa dasar teori, antara lain:

2.2.1 *Wireless Mesh Network*(WMN)

WMN merupakan jaringan komunikasi *Wireless* yang terbentuk dari *node* radio dengan jumlah minimal dua atau lebih jalur komunikasi data pada setiap *node*(Febriadi, dkk.,2013). Setiap *node* WMN bertindak sebagai *host* dan *router* untuk meneruskan paket-paket informasi yang akan dikirim menuju *node* lain(Febriadi, dkk.,2013).

Kelebihan dari WMN yaitu kemampuan untuk melakukan *self-healing* dan *self-configuring. Self-healing* merupakan kemampuan untuk melakukan penyambungan *node* secara otomatis ketika terdapat *node* yang aktif. *Self-*configuring merupakan kemampuan untuk melakukan konfigurasi jaringan secara otomatis. Arsitektur gambar dari *WMN* dapat dilihat pada gambar 2.1.



Gambar 2.1 Arsitektur Jaringan *Wireless Mesh Network*

Sumber:(Ratelinggi, dkk.,2015)

2.2.2 Protokol *Routing*

Protokol *Routing* pada umumnya tidak melakukan metode tertentu di setiap *node*. Protokol *Routing* mendefinisikan format pesan(informasi) yang dikirim untuk perhitungan matrik rute. Protokol *Routing* pada dasarnya dibagi menjadi 2 kategori, yakni reaktif, proaktif (Ratelinggi, dkk.,2015). Protokol *Routing* reaktif hanya mencari *node* sesuai dengan arah tujuan pengiriman paket data. Protokol *Routing* proaktif akan mempertahankan rute ke semua *node* terlepas dari permintaan pada lapisan atasnya(Abolhasan, dkk.,2009). Jenis protokol *Routing* memiliki karakteristik yang berbeda antara satu dengan yang lainnya. Pada penelitian ini akan membahas 3 jenis Protokol *Routing* OLSR, BATMAN dan Babel. Adapun karakteristik Protokol *Routing* OLSR, BATMAN dan Babel adalah sebagai berikut.

2.2.2.1 OLSR

Optimized Link State Routing Protocol (OLSR) adalah salah satu Protokol *Routing* proaktif yang di desain untuk jaringan *ad-hoc*. Protokol OLSR merupakan optimasi dari Protokol *link state* sehingga mewarisi stabilitas algoritma *link state* dengan keuntungan ketersediaan rute *Routing* yang dibutuhkan setiap saat. *Overhead* yang disebabkan oleh *flooding*(pembanjiran) pesan kontrol, dapat diminimalisir dengan memilih *node* lain yang tersedia dalam jaringan bertujuan

memulihkan kembali pesan kontrol. Proses pemilihan *node* pada OLSR menggunakan MPRS. Teknik ini secara signifikan dapat mengurangi jumlah transmisi ulang yang diperlukan dengan melakukan *broadcast* pesan ke semua *node* dalam jaringan untuk membuat tabel *Routing* baru. Ketersediaan *path*(jalur *Routing*) terpendek OLSR, dibutuhkan *partial link state* dengan pembanjiran *broadcast* data. Set minimal informasi *link state* yang diperlukan mengharuskan semua *node* yang terpilih sebagai MPRS menyatakan *link*(hubungan) ke *node* penyeleksi MPR mereka. Hubungan yang terbentuk akan digunakan sebagai media pertukaran informasi antar *node* untuk mengurangi kemungkinan terjadinya redudansi data pada setiap *node*.

OLSR mengoptimalkan reaksi perubahan topologi dengan mengurangi interval waktu maksimum transmisi pesan kontrol secara berkala. Selain itu, OLSR terus mempertahankan jalur *Routing* antar *node* dalam jaringan meskipun ketersediaan *node* sumber atau tujuan yang dapat berubah sewaktu-waktu. OLSR sangat cocok untuk diaplikasikan pada jaringan bercakupan besar dan padat ditunjang oleh optimasi dari kinerja baik MPRS. Semakin besar dan padat jaringan, maka semakin besar pula optimasi yang dapat dicapai ole OLSR dibandingkan dengan algoritma *link state* (Clausen, dkk.,2003).

Node OLSR melakukan pengiriman pesan kontrol secara berkala bertujuan mempertahankan *path Routing*. Pesan kontrol yang dikirim berisikan nomor urut sesuai urutan pengiriman pesan. Sehingga, penerima pesan kontrol dapat dengan mudah mengidentifikasi informasi yang lebih baru untuk memperbaharui tabel informasi pada masing-masing *node*(Clausen, dkk.,2003).

Proses komunikasi OLSR bergantung pada informasi yang terdapat di tabel *Routing* masing-masing *node* jaringan. Setiap *node* meneruskan informasi terhadap *node* tetangganya sebagai "*multipoint relay*" (MPR). *Node* yang terpilih sebagai MPRS bertanggung jawab untuk meneruskan *control traffic* yakni dengan melakukan difusi ke seluruh jaringan(Thakare, dkk.,2002). MPRS menyediakan mekanisme yang efisien untuk membanjiri *control traffic* dengan mengurangi jumlah transmisi yang diperlukan. *Node* MPRS memiliki tanggung jawab khusus ketika mendeklarasikan informasi *link state* dalam jaringan. Pendeklarasian informasi *link-state* digunakan untuk mendapatkan informasi *link-state* tambahan. Informasi *link-state* tambahan yang tersedia dapat dimanfaatkan untuk redudansi informasi pada table *Routing* di setiap *node* MPRS.

Node yang terpilih sebagai MPR oleh beberapa *node* tetangga mengumumkan informasi secara berkala didalam isi pesan kontrol mereka. Metode ini dilakukan *node* untuk mengumumkan ke jaringan, bahwa ia memiliki kemampuan memilih *node* lain sebagai MPR. MPRS digunakan sebagai pembangun *path* dari *node* yang awal ke *node* tujuan manapun dalam jaringan. Selanjutnya, Protokol menggunakan MPRS untuk memfasilitasi efisiensi *flooding* pesan kontrol dalam jaringan.

MPRS bekerja dengan memilih *node* lain berjarak satu *hop* dengan terarah. Oleh karena itu, dalam melakukan pemilihan *path* melalui *node* MPRS secara tidak

langsung dapat menghindari permasalahan yang terkait dengan transfer paket data melalui hubungan banyak arah(Clausen, dkk.,2003). Seperti permasalahan tidak mendapatkan informasi dari hop lainnya, teknik ini digunakan pada *traffic unicast*.

2.2.2.2 BATMAN

Permasalahan yang sering ditemui pada protokol *Routing* klasik adalah tidak cocok diaplikasikan pada jaringan *ad-hoc nirkabel*. Ini disebabkan karena posisi susunan *client* jaringan *nirkabel* tidak terstruktur(dinamis), mengakibatkan perubahan susunan topologi jaringan *nirkabel* dan tidak dapat diandalkan pada media *inheren*(Lindner, Mark., et all, 2006). OLSR merupakan protokol saat ini yang didesain untuk skenario seperti itu. OLSR mengalami sejumlah perubahan dari spesifikasi aslinya ketika bekerja pada WMN skala besar.

Better Approach To Mobile Ad-Hoc Network (BATMAN) merupakan sebuah protokol *Routing* bersifat *proactive* berupa pengembangan dari protokol *Routing* OLSR (Margolang, 2014). Pendekatan yang dilakukan algoritma BATMAN adalah dengan membagi pengetahuan tentang *path end-to-end* terbaik antar *node* didalam jaringan *mesh* untuk semua *node* jaringan. Pemeliharaan informasi tentang *hop* berikutnya yang memiliki jalur terbaik terhadap semua *node* lain diketahui oleh setiap *node* aktif. Protokol *Routing* BATMAN didesain untuk diterapkan pada jaringan dengan *link* yang tidak dapat diandalkan.

Protokol BATMAN dapat digambarkan bahwasannya setiap *node* melakukan *flooding* dengan mengirimkan pesan *broadcast* (pesan originator atau OGM) untuk menginformasikan keberadaannya ke *node* tetangga. *Flooding* OGM dilakukan untuk menghindari beredarnya informasi *Routing* yang berbeda sehingga mencegah adanya *Routing loop*. *Node* tetangga kembali melakukan *broadcast* OGM untuk menginformasikan *node* tetangga-nya mengenai keberadaan inisiator asli pesan ini dan seterusnya. BATMAN kemudian memilih tetangga yang merespon pesan OGM dan mengkonfigurasi tabel *Routing* masing-masing. *Broadcast* pesan OGM yang dilakukan pada BATMAN akan menyebabkan *flooding* pesan originator. Besar paket OGM yang baku adalah 52 *byte* sudah termasuk IP dan UDP *overhead*. OGM mengandung alamat originator, alamat *node* transmisi paket, TTL dan nomor urut pesan.

OGM yang mengikuti *path* padat akan kehilangan paket atau mengalami keterlambatan pengiriman pesan pada jaringan *Mesh.* OGM yang memilih *path* lebih lancar akan mengirim paket lebih cepat dan lebih dapat diandalkan. Jumlah penerimaan OGM oleh *node* tetangga dapat diketahui dengan melihat nomor urut yang diberikan *node* awal pengirim OGM. *Node* tetangga masing-masing menerima OGM maksimal sekali dan hanya diterima dari *node* tetangga yang telah teridentifikasi sebagai *path* terbaik hop berikutnya dari inisiator asli OGM.

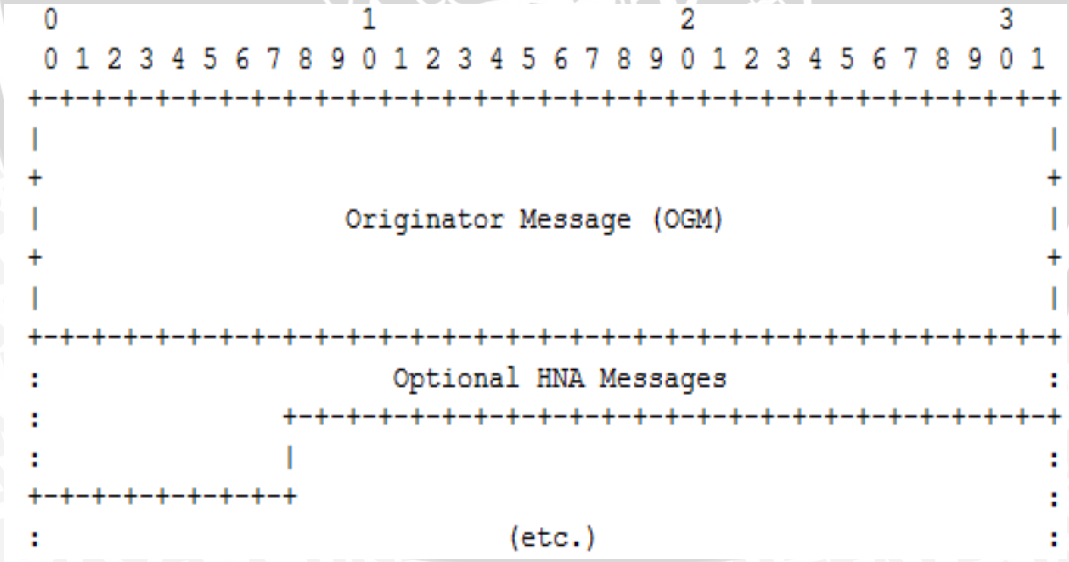
A. *Karakteristik BATMAN*

Protokol *Routing* BATMAN bekerja pada *layer* 3. Mekanisme *Routing* BATMAN adalah *IP Address* sebagai sarana komunikasi antar node. BATMAN hanya memperhatikan penentuan *best nexthop* yang mempengaruhi mekanisme *Routing* menjadi lebih efisien dan cepat. *Best nexthop* merupakan metode yang digunakan BATMAN untuk menentukan tetangga *hop* terbaik dengan mempertimbangkan respon dari pesan OGM yang telah di *broadcast.* Salah satu metode penentuan *best nexthop* pada BATMAN dilakukan dengan *broadcast* OGM. BATMAN melakukan *broadcast* OGM ke seluruh *node* dalam jaringan.

BATMAN didesain untuk diterapkan pada jaringan *unreliable* seperti *wireless* dan selalu mengalami banyak *packet loss* (Margolang, 2014). BATMAN hanya mengenali satu *single hop* tetangga tanpa mengetahui topologi MANET dari keseluruhan jaringan sebagai dasar penentuan *nexthop* setiap *node*. Sebagai contoh, dalam sebuah MANET terdapat *node* A-K. *Node* A mengetahui bahwa terdapat *node* K pada suatu tempat di dalam MANET. *Node* A dapat mengetahui keberadaan *node* K melalui *neighbour* terbaik. Akan tetapi, *node* A tidak mengetahui jarah *hop* atau *node* untuk menuju *node* K. Sehingga, BATMAN memiliki kelebihan *bandwith friendliness* yang sulit divisualisasikan. Sedangkan untuk mempermudah BATMAN memiliki *vis server* yang berfungsi sebagai pencari data jaringan dari setiap *node* yang digunakan sebagai pemvisualisasi MANET dalam bentuk grafik dari topologi jaringan yang ada(Neumann,dkk., 2008).

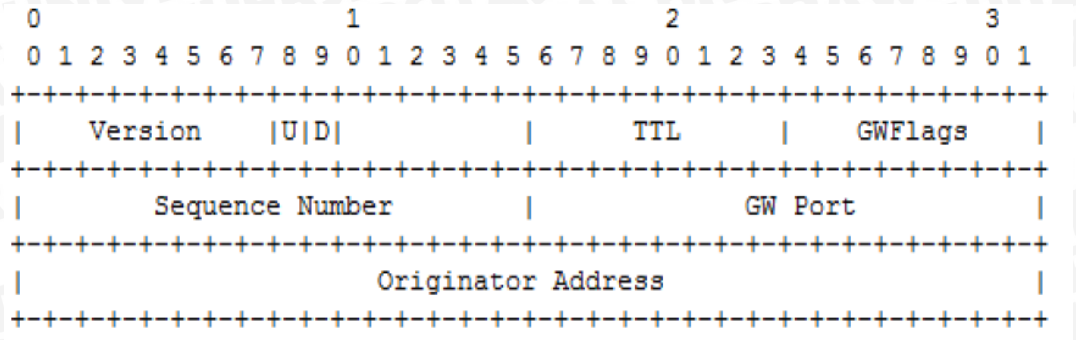
B. Format Paket BATMAN

Format paket BATMAN dapat diilustrasikan seperti gambar 2.2**.**



Gambar 2.2 Format Paket BATMAN
Sumber: (Neumann,dkk., 2008)

Paket BATMAN adalah paket UDP terdiri dari OGM dan *Optional Host Network Announcement* (HNA) *Message*. OGM memiliki besar paket tetap, yakni 12 oktet. Isi OGM diilustrasikan pada gambar 2.3.



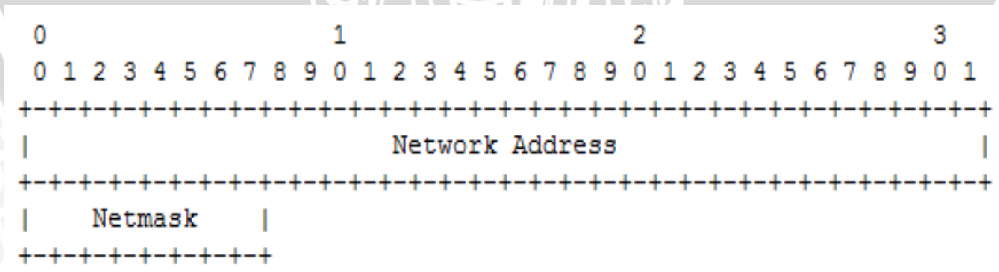
Gambar 2.3 Format OGM

Sumber: (Neumann,dkk., 2008)

OGM adalah paket yang dikirim sebagai memberitahu eksistensi *node* yang berada di dalam MANET. Isi dari OGM antara lain (Neumann,dkk., 2008):

a. *Version*: untuk membedakan paket yang berbeda versi BATMAN. Jika menerima paket BATMAN yang berbeda, maka paket tersebut akan diacuhkan.
b. *Is-direct-link flag*: digunakan untuk menunjukkan apakah sebuah *node* merupakan *node* tetangga atau bukan.
c. *Unidirectional flag*: digunakan untuk menunjukkan apakah *node* tetangga menggunakan hubungan *bidirectional* atau tidak.
d. *TTL (Time To Live)*: digunakan untuk membatasi *hop* pengiriman OGM.
e. *Gateway flags*: digunakan untuk menunjukkan jika *host/node* ini memberikan layanan sambungan ke *internet* (*gateway*).
f. *Squence number*: *originator* pada OGM akan menambahkan satu setiap *sequence* number dari OGM baru sebagai pengenal urutan pengiriman paket.
g. *Originator address*: alamat IPv4 dari *interface* BATMAN dimana OGM dihasilkan.

Paket HNA *message* dapat diilustrasikan pada gambar 2.4.



Gambar 2.4 *Format* Pesan HNA

Sumber: (Neumann,dkk., 2008)

Keterangan:

a) *Netmask*: jumlah *bit* yang merepresentasikan besar dari *network*.
b) *Network Address*: alamat *network* IPv4 yang digunakan.

C. Cara Kerja OGM

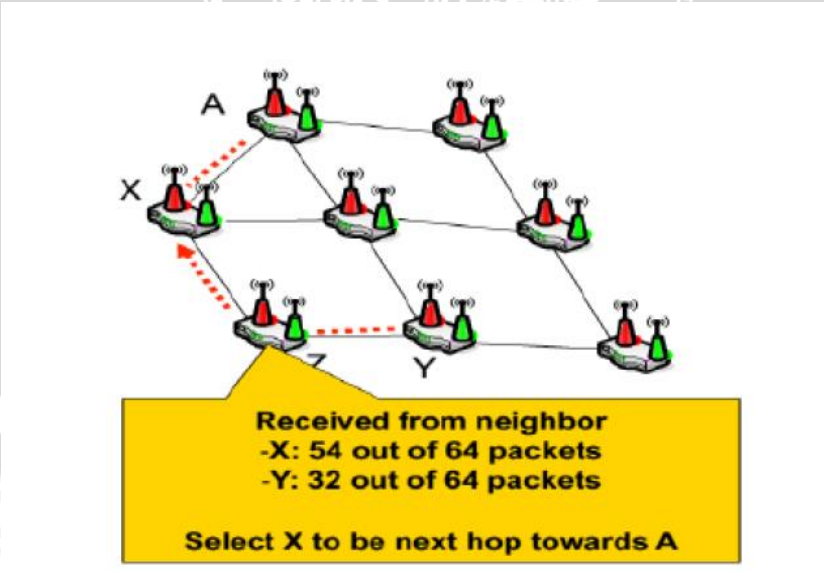
Beirkut merupakan cara penyebaran OGM (Neumann,dkk., 2008):

1. OGM di-*broadcast* secara periodik (dengan *interval* satu detik) oleh setiap *node* dengan besar paket masing-masing sekitar 52 *byte*.
2. Paket OGM dikirim ke *node* tetangga untuk memberitahukan eksistensi dari *node* pengirim.
3. *Node* melakukan *selective flooding* dengan hanya melakukan *broadcast* ulang paling banyak satu kali kepada *node* tetangga yang sudah diidentifikasikan memiliki jalur yang terbaik.

Pesan OGM yang diterima kemudian diproses dengan ketentuan:

a) Paket OGM yang di-*broadcast* pada umumnya hilang dikarenakan sambungan yang lemah ataupun terjadi tabrakan.
b) OGM yang melalui jalur yang baik tersebar lebih cepat dan lebih *reliable*.
c) Setiap *node* menghitung *node* tetangga mana yang memberikan *broadcast* paket yang paling banyak.
d) Berdasarkan proses perhitungan tersebut, *node* tetangga tersebut akan ditandai sebagai *node* dengan jalur yang baik (*good path*) untuk menuju sumber paket.
e) OGM juga melakukan pengecekan untuk *bidirectional link*.

Pemrosessan pesan OMG dapat divisualisasikan seperti pada gambar 2.5.



Gambar 2.5 Mekanisme Pemrosesan OGM

Sumber:(Kassler, A.J., 2012)

D. Mekanisme Routing BATMAN

BATMAN menjalankan *Routing daemon* untuk menjaga *Routing table*-nya terus *update*. *Routing daemon* berfungsi menjaga *path* pada OGM-OGM baru dan menjaga *list* seluruh *originator* pengirim OGM. BATMAN menjaga satu *entry*

dedicated Routing untuk setiap OGM dan HNA yang telah dikenal. *Routing entry* berfungsi menunjukkan *interface outgoing* dari BATMAN dan *IP Address* dari *nexthop direct link* tetangga menuju *originator* terkait. BATMAN harus memasukkan sebuah *path* menuju semua *node*, bahkan jika *node* tersebut merupakan tetangga ber-status *link-local bidirectional single hop* (Neumann,dkk., 2008).

E. Pemilihan dan Pembentukan Path BATMAN

Pemilihan *node* pada BATMAN sebagai langkah awal pembentukan *path* menuju *node* lain yakni ketika sebuah *node* mendapat OGM dari originator yang tidak dikenal atau mendapat OGM untuk *node* yang tidak dikenal diluar cakupan jaringan. Sehingga *node* baru dari luar jaringan akan dimasukkan ke dalam *Routing table* yang akan dilanjutkan dengan mekanisme pemilihan tetangga menggunakan *link-local bidirectional*. Mekanisme pemilihan tetangga menggunakan *link-local bidirectional* dimaksudkan untuk memilih jalur terbaik yang akan dijadikan sebagai *gateway* menuju tujuan. *update Routing table* akan dilakukan apabila terjadi perubahan susunan peringkat tetangga dengan *path* terbaik (Margolang, 2014).

F. Penghapusan Path BATMAN

Penghapusan *path* pada BATMAN dari *Routing table* terjadi ketika sebuah *node* tidak menerima OGM dan HNA dari *originator* pada rentang waktu melebihi *WIDOW_SIZE* dan interval *PURGE_TIMEOUT* (Neumann,dkk., 2008). Sedangkan penghapusan *path* dari *Routing table* akan dilakukan secara otomatis.

G. BATMAN-advanced

BATMAN-*advanced* atau BATMAN-adv merupakan implementasi protokol *Routing* BATMAN berupa modul *kernel*. BATMAN-adv bekerja di lapisan *data link layer* pada *network layer*. Protokol *Routing* BATMAN bekerja pada layer 3. Protokol *Routing* yang bekerja pada layer 3 saling bertukar informasi *Routing* dengan mengirimkan paket UDP. Selain itu, Protokol *Routing* pada layer 3 menetapkan keputusan *Routing* mereka dengan memanipulasi *kernel Routing table*. Aktifitas *Routing* pada BATMAN di lakukan oleh BATMAN-adv yang beroperasi sepenuhnya pada *layer* 2. Semua mekanisme dan penetapan jalur *Routing* dilakukan pada layer 2 dengan menggunakan MAC (*Media Access Control*) *Address* (Neumann,dkk., 2008).

Proses enkapsulasi dan *packet forwarding* pada BATMAN-adv dilakukan untuk seluruh paket. Paket data dapat diterima seluruh tujuan pengiriman dan menjembatani seluruh *node* untuk berpartisipasi dalam *switch virtual network*. Mekanisme *Routing* pada BATMAN-adv yakni semua *node* tidak mengetahui topologi jaringan melainkan dengan *link-local* tetangga yang mengakibatkan perubahan *network* tidak mempengaruhi *node*. Karakteristik BATMAN-adv adalah (Neumann,dkk., 2008):

a. Pada BATMAN-adv, berbagai aplikasi *network layer* dapat dijalankan, misalnya IPv4, IPv6, DHCP, IPX, dan lain-lain.

b. *Node* dapat berpastisipasi ke dalam MANET tanpa memiliki *IP Address* karena BATMAN-adv menggunakan MAC *Address*.
c. *Node* yang bukan merupakan bagian MANET dapat dengan mudah terintegrasi dengan MANET, contoh *notebook* tanpa BATMAN-adv dapat terhubung dengan BATMAN-adv di MANET.
d. Terdapat mekanisme *optimizing* dari aliran data yang melalui MANET (*interface alternating*, *low overhead*, *forward error correction*, dsb.)
e. Protokol berjalan dengan mengandalkan *broadcast/multicast* paket melalui *node* MANET dan *non*-MANET.

Konfigurasi dan *debugging* terhadap *kernel* modul pada BATMAN-adv digunakan batctl *tool*. Batctl dapat digunakan untuk melakukan aktifitas *ping*, *traceroute*, dan *tcpdump* pada lapisan dua(Margolang, 2014).

H. Interface Virtual bat0

Permasalahan yang timbul akibat kebanyakan aplikasi dapat diatasi dengan memanfaatkan pengalamatan logika di lapisan 3 (*network layer*) berupa IPv4 sebagai sarana komunikasi dengan aplikasi lain, sementara BATMAN-adv bekerja di lapisan 2 (*data-link layer*). BATMAN-adv membuat sebuah *interface virtual* yaitu bat0 di setiap *node*, dengan fungsi dasar *interface* bat0 untuk membantu BATMAN-adv dalam memenuhi kebutuhan pengalamatan logika pada lapisan 3. Bat0 memungkinkan BATMAN-adv untuk mendukung aplikasi yang bekerja dengan memanfaatkan pengalamatan lapisan 3.

2.2.2.3 Babel

Babel adalah Protokol *Routing Proactive* berdasarkan algoritma *Distance Vector* dan merupakan evolusi dari harapan jumlah transmisi(ETX) algoritma dengan pemilihan rute yang lebih cerdas dari pengguna metode *hop-count* sederhana (Mbolhasan,dkk., 2009). Metode *hop-count* cukup efektif untuk direplikasikan pada jaringan padat. Babel mempunyai dua karakteristik khas untuk mengoptimalkan kinerja relay (Mbolhasan,dkk., 2009). Pertama, menggunakan *history-sensitive* pemilihan rute untuk meminimalkan dampak *route flaps*. Kedua, Babel mengesekusi *update* reaktif dan memaksa untuk meminta informasi ruting ketika mendetaksi kegagalan ruting dari salah satu *node* tetangga.

Babel merupakan Protokol *Routing* yang dapat menggunakan IPv4 dan IPv6 secara bersamaan. Sifat *hybrid* yang dimiliki Babel sehingga paket *update* tunggalnya dapat membuat rute-rute untuk beberapa Protokol *network-layer* (IPv4 & IPv6) sekaligus. Dengan demikian, Babel dapat dikatakan secara khusus dan efisien cocok untuk me-*manage* jaringan dengan dua pengalamatan logika(IPv4 dan IPv6). Meski demikian, Babel memiliki dua keterbatasan yang tidak dapat digunakan pada beberapa situasi (Margolang, 2014):

a) Babel bergantung pada *update* tabel *Routing* secara berkala dari *path* awal yang lancar. Hal ini dikarenakan pada jaringan besar, Babel akan menghasilkan *path* lebih banyak dibandingkan Protokol lain yang

melakukan *update* table *Routing* ketika terjadi perubahan topologi jaringan.

b) Babel bergantung pada *hold time* meskipun terdapat salah satu *node* mati ditengah pengiriman paket data. Ketika Babel berada pada kondisi salah satu *prefix* yang sebelumnya terpisah dan sekarang akan bergabung dalam jaringan, maka *path* yang terbentuk menjadi *unreachable* untuk beberapa saat. Kasus ini membuat Babel kurang *support* jika digunakan pada jaringan *mobile* yang mengimplementasikan *prefix aggregation* otomatis.

Babel memiliki transmisi informasi yang akan digunakan untuk proses *Routing* dengan sesama *node* Babel. Paket data yang akan dikirim oleh Protokol Babel dikemas dalam body datagram UDP. Sedangkan informasi *Routing* yang digunakan Babel akan diletakkan kedalam sebuah *Type Length value* (TLV) yang berjumlah satu atau lebih pada setiap paket Babel. Babel akan meminta *acknowledgement* dari semua paket Babel yang dikirim dengan menambah *Acknowledgement Request* TLV. Perhitungan *cost* dari *node* tetangga Babel didapat dari informasi pesan *Hello* dan IHU("*I Heard You*") yang didapat dengan melakukan *broadcast* secara periodik (Margolang, 2014). Pengiriman pesan *Hello* dilakukan secara *broadcast* dan disertai dengan pesan IHU untuk masing-masing *node* tetangga Babel. Pembaharuan pada *Routing* Babel dapat dilakukan masing-masing *node* dengan mengirimkan update pada berbagai keadaan (Chroboczek, 2014). *Node* Babel memiliki perbedaan tabel *Routing* setiap melakukan *update* data *Routing* antara satu dengan yang lain. Data keragaman ini dihitung sebagai berikut:

a. jika *update* untuk distribusi ulang *path* lokal, maka nilai tabel *Routing* bergantung pada implementasi;
b. jika update diperuntukkan *path* dalam *Routing* Babel, maka perbedaan informasi diambil dari tabel *Routing* Babel.

Babel memiliki kondisi kehandalan yang menjamin tidak adanya perulangan *Routing*. Pada kondisi ini Babel akan mengabaikan *update* rute yang tidak memenuhi kondisi kehandalan. Rute handal akan terpenuhi ketika metrik *node* lokal bernilai tidak lebih besar dari metrik *node* terpilih saat ini dan tidak memungkinkan terjadi perulangan *Routing.* Ketika Babel kehabisan rute yang handal, maka akan terjadi *starvation*(kelaparan). *Routing* Babel menghadapi keadaan *starvation* dengan menggunakan rute yang berurutan. Setiap rute mengandung nomor urut s dan matrik m untuk sebuah *node* n. Nomor urut s merupakan penentu pembaharuan penyebaran rute yang disebarluaskan melalui jaringan dan hanya bertambah oleh n.

A. Pemilihan Path Babel

Keberhasilan pengiriman paket data banyak dipengaruhi oleh kepadatan lalulintas data, jarak *Routing*, *noise* ,kekuatan jaringan, dan pemilihan *path* yang tepat. Pemilihan *path* merupakan salah satu proses penting dalam melakukan pengiriman data. *Path* yang dipilih merupakan *path* tunggal untuk meneruskan dan menyebarkan paket-paket data menuju *node* lain. Babel melakukan pemilihan

path secara fleksibel berdasarkan metrik, *router-id*, *node* tetangga, dan keadaan *path* pada waktu itu. Sifat-sifat yang perlu diperhatikan dalam pemilihan *path* meliputi:

a) *Path* dengan matrik tidak terbatas tidak akan pernah dipilih.
b) *Path* yang memliki kemungkinan untuk melakukan pengulangan *Routing* tidak akan pernah dipilih.

Ekstensi keragaman Protokol memungkinkan *router* Babel melampirkan informasi mengenai frekuensi radio yang digunakan dengan tujuan dapat mempertahankan *path* yang disebut " perbedaan informasi " *path*. "Kami berasumsi bahwa semua *link* dapat dikategorikan ke dalam salah satu dari kategori berikut"(Chroboczek, 2014):

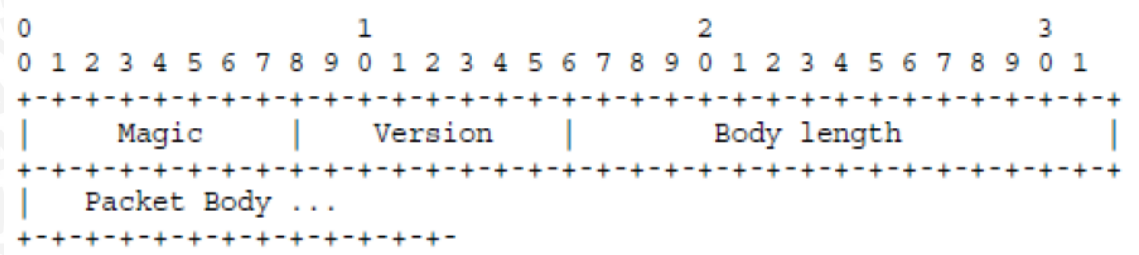
a) *Non-interfering link*, misalnya *link* kabel;
b) *Link* yang memiliki frekuensi yang terdefinisi dengan baik dan hanya mengganggu *link* lain pada frekuensi yang sama;
c) Proses ini dideskripsikan melalui satu nomor *channel* dengan *range integer* antara 1 dan 254;
d) *interfering link*, merupakan *link* yang mengganggu *link* lain kecuali *non-interfering link*.

B. Konvergensi Babel

Ketika terjadi perubahan topologi yang menyebabkan terjadinya konvergensi, *node* Babel menyikapi dengan meminta nomor urut baru(dengan *sequence number* TLV) pada saat diperlukan. *Routing* Babel dapat mengurangi waktu tunggu pengiriman nomor urut baru dalam interval periode berikutnya seperti pada BSDV. Konvergensi dapat dicapai dengan pemilihan rute terbaik, Babel membutuhkan *delay* sebelum optimasi tabel *Routing*. *Delay* yang terjadi ketika melakukan konvergensi dalam optimasi rute Babel dipengaruhi oleh adanya *packet loss*. Sedangkan besar waktu *delay* yang dibutuhkan sekitar 1 menit(dengan interval *update default* 20 detik) (Margolang, 2014).

C. Format Paket Babel

Babel memiliki format pengiriman paket yang digunakan dalam pengiriman paket datagram. Format yang digunakan Babel dalam melakukan pengiriman paket data memuat 4 oktet pada *header* dan diikuti dengan paket TLV pada *body* datagram. Ketika terdapat paket data yang ikut dalam paket *header*, Babel akan mengabaikan paket data selebihnya. Paket Babel dapat dilihat pada gambar 2.6.



Gambar 2.6 Format Paket *Header*

Sumber: (Chroboczek, 2014)

Keterangan:

a. *Magic* : merupakan nilai sementara, bernilai 42 (desimal). Paket dengan oktet pertama yang memiliki nilai berbeda dengan 42 akan diabaikan.
b. *Version* : menjelaskan mengenai versi Protokol Babel. Babel akan mengabaikan paket yang berbeda dengan versi Protokol saat ini.
c. *Body length* : merupakan panjang oktet Babel.
d. *Packet body* : merupakan paket TLV.

D. Format TLV

TLV merupakan paket yang ikut serta dalam paket *header* Babel. Paket yang dimiliki TLV ada 10 *sub*-TLV lagi dengan fungsi masing-masing. Perbedaan *Sub*-TLV ditandai dari *field type* pada paket TLV. Adapun 10 *sub*-TLV tersebut adalah (Chroboczek, 2014):

a. *Type 0*
   Paket sub TLV pad1. Paket yang diabaikan pada saat transmisi.
b. *Type 1*
   Paket sub TLV padN. Paket yang diabaikan pada saat transmisi.
c. *Type 2*
   Paket sub TLV *Acknowledgement Request*. Berfungsi untuk meminta penerima TLV mengirim sebuah TLV *Acknowledgement*. TLV ini memiliki besaran *centisecond* yang ditentukan oleh *Interval field*.
d. *Type 3*
   Paket sub TLV *Acknowledgement.* TLV yang dikirim oleh node penerima *Acknowledment Request.*
e. *Type 4*
   Paket sub TLV *Hello.* TLV yang digunakan sebagai pencarian tetangga dan menentukan *cost* penerimaan *link*.
f. *Type 5*
   Paket sub TLV IHU. TLV untuk konfirmasi pencapaian dua arah dan membawa sebuah *cost* transmisi *link*.
g. *Type 6*
   Paket sub TLV *Router-Id*. TLV *Router*-Id yang membuat sebuah *router*-Id dari bagian TLV *Update*.

h. *Type 7*

Paket sub TLV *Next Hop*. TLV untuk membangun alamat *next-hop* dari bagian TLV *Update*.

i. *Type 8*

Paket sub TLV *Update*. TLV untuk menyebarkan atau membatalkan sebuah rute. Sebagai optimisasi, TLV ini dapat memiliki efek samping dalam membangun sebuah *router*-id dan sebuah *next-hop* baru.

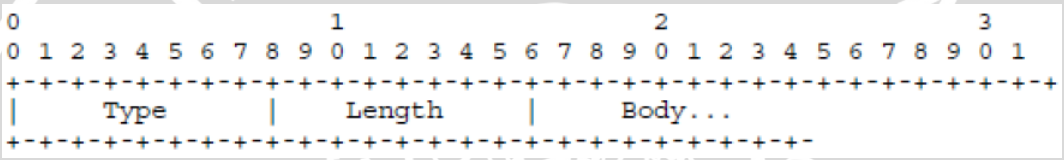
j. *Type 9*

Paket sub TLV *Route Request*. TLV untuk meminta penerima mengirimkan update dari *prefix* yang diberikan atau tabel *Routing* yang ada pada penerima dengan lengkap.

k. *Type 10*

Paket sub TLV *Seqno Request*. TLV berfungsi meminta penerima mengirimkan pembaharuan untuk prefix tertentu dengan nomor urut tertentu, atau meminta melakukan *forward request* jika tidak memuaskan.

Format paket TLV pada header paket Babel dapat dilihat pada gambar 2.7.



Gambar 2.7 Format Paket TLV

Sumber: (Chroboczek, 2014)

Keterangan:

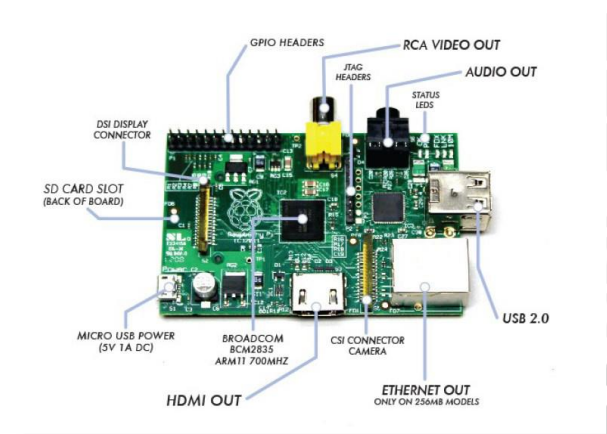
a. Type: merupakan field yang menjelaskan tipe dari paket TLV.

b. *Length*: merupakan panjang dari *body*. Jika *body* lebih panjang dari panjang tipe TLV yang seharusnya, semua sisa data yang lebih akan diabaikan.

c. *Body*: merupakan *body* dari TLV, isinya tergantung dari tipe TLV.

2.2.3 Raspberry pi

Raspberry pi adalah sebuah SBC(*Single Board Computer*) sebesar kartu nama yang dikembangkan oleh yayasan *Raspberry Pi* di Inggris(UK) dengan maksud dan tujuan dapat memacu perkembangan ilmu pengetahuan di bidang komputer dasar. Komputer mini ini banyak digunakan oleh para ilmuan untuk melakukan penelitian, hal ini didukung oleh kinerja *Raspberry Pi* yang efisien, mudah digunakan, dan memiliki fungsi menyerupai personal komputer pada umumnya. Bentuk *Raspberry Pi* dapat dilihat pada gambar 2.8.



Gambar 2.8 *Board Raspberry Pi*

Sumber: (Mariana ,2012)

Spesifikasi *Raspberry Pi* dapat dilihat di Tabel 2.1.

Tabel 2.1 Spesifikasi *Raspberry Pi*

Sumber: (Mariana ,2012)

Komponen	Keterangan
CPU	700 MHz ARM1176JZF-S core (ARM6 *family*)
Memory	512 MB/256 MB (shared with GPU)
USB ports	2 (via integrated USB hub)
Onboard Storage	SD/MMC/SDIO card slot
Onboard Network	10/100 Ethernet (RJ45)
Low-level Peripherals	8 × GPIO, UART, I²C bus, SPI bus with two chip selects, +3.3 V, +5 V, ground[58][63]
Power Source	5 volt via MicroUSB or GPIO Header
Planned operating systems	Debian GNU/Linux, Fedora, Arch Linux ARM, RISC OS, OpenWRT

Raspberry Pi memiliki 5 model dengan spesifikasi yang berbeda, diantaranya *Raspberry Pi* 2 model B, *Raspberry Pi* model A, *Raspberry Pi* model B, *Raspberry Pi* B+, *Raspberry Pi* A+. *Raspberry Pi,* tidak dibekali *Network Interface Card* pada

komponen internalnya. Sehingga diperlukan *wifi-dongle* untuk mengambil peranan tersebut. Pemilihan *wifi-dongle* perlu diperhatikan, hal ini dikarenakan tidak semua jenis *wifi-dongle* langsung *support* pada *Raspberry Pi.* Jika terjadi hal seperti ini, maka perlu dilakukan pengaturan tambahan pada *Interface Raspberry Pi.*

2.2.4 Parameter QoS

Quality of Service(QoS) adalah kamampuan jaringan untuk menyediakan layanan yang memuaskan pengguna. Penjelasan ini dapat diartikan secara spesifik sebagai kemampuan jaringan untuk menyediakan layanan yang lebih baik kepada kelas paket tertentu prioritas jaringan(Syahrial, 2014).

2.2.4.1 Throughput

Throughput merupakan kemampuan sebenarnya sebuah jaringan dalam melakukan pengiriman data(Friginal,dkk., 2012). *Throughput* memiliki pengertian yang hampir sama dengan *bandwidth*. *Throughput* dapat diartikan sebagai *bandwidth* dengan kondisi sebenarnya. Yang menjadi pembeda adalah pada *bandwidth* bersifat *fix* karena sudah ditentukan di awal pengiriman paket data. Sedangkan *throughput* memiliki sifat dinamis menyesuaikan trafik yang sedang terjadi. Nilai *throughput* dapat diperoleh dariu hasil jumlah data yang dikirim dibagi waktu pengiriman data.

2.2.4.2 Jitter

Jitter merupakan perbedaan waktu kedatangan paket dari pengirim ke penerima data dengan waktu yang diharapkan(Friginal,dkk., 2012). *Jitter* berpengaruh pada adanya sampling pada penerima data yang menyebabkan kerusakan informasi ketika paket data diterima oleh penerima. *Jitter* dapat dihitung menggunakan persamaan J(i) = J(i-1) + (|D(i-1,i)| - J(i-i))/16. Bentuk *jitter* dapat dilihat pada gambar 2.9.

Gambar 2.9 Contoh *Jitter.*

Sumber : (Syahrial, 2014)

2.2.4.3 Packet Loss

Packet Loss merupakan jumlah peket yang hilang pada pengiriman paket data dalam jaringan(Syahrial, 2014). Ketika *Packet Loss* bernilai besar, maka aktifitas jaringan sibuk atau terjadi overload. Jaringan dapat dikatakan buruk, apabila nilai *Packet Loss* besar(Romadhon, 2014). *Packet Loss* terjadi karena *collision* dan *congestion* pada jaringan yang berpengaruh pada semua aplikasi. Karena *retransmisi* akan mengurangi efisiensi jaringan secara keseluruhan. Beberapa penyebab terjadinya *Packet Loss* adalah:

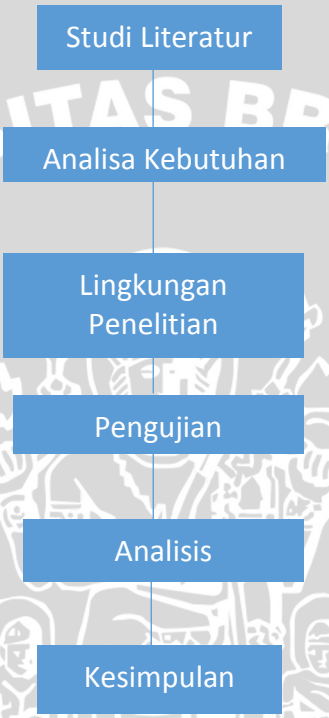
a. Congestion, adalah terjadinya antrian yang berlebihan dalam jaringan.

b. Node yang bekerja melebihi kapasitas buffer.
c. Memory yang terbatas pada node.
d. Policing atau kontrol terhadap jaringan, berfungsi memastikan bahwa jumlah *traffic* yang mengalir sesuai dengan besar bandwidth. Policing control akan membuang kelebihan trafik yang ada ketika besarnya trafik yang mengalir didalam jaringan melebihi dari kapasitas bandwidth yang ada.

BAB 3 METODE PENELITIAN

Untuk dapat melakukan perpandingan performansi protokol *Routing*, terdapat proses-proses yang saling berkaitan agar dapat dilakukan pengujian performansi. Pada bab ini akan dijelaskan metode yang digunakan dan langkah-langkah yang dilakuan dalam Analisa Perbandingan Performansi Protokol *Routing* OLSR, BATMAN dan Babel pada Jaringan Mesh.

Tahapan penelitian dapat digambarkan dalam bentuk diagram blok yang ditunjukkan gambar 3.1



Gambar 3.1 Diagram Alur Metode Penelitian

Jenis penelitian yang dilakukan adalah non-implementatif analitik. Di karenakan penelitian ini bertujuan menganalisa hasil perbandingan performansi Protokol *Routing* OLSR, BATMAN dan Babel ketika melakukan pengiriman paket data UDP pada jaringan WMN.

3.1 Studi Literatur

Penelitian ini memerlukan studi literatur untuk mencari dasar-dasar teori sebagai bahan referensi dalam melakukan perbandingan performansi protokol *Routing* pada WMN. Sumber mengenai WMN, protokol *Routing* sebagai dasar penelitian diperoleh dari buku, jurnal, maupun artikel dari *internet* dengan tujuan dapat digunakan sebagai landasan dasar untuk melakukan analisa perbandingan performansi protokol *Routing* pada WMN.

3.2 Analisa Kebutuhan

Kebutuhan yang diperlukan dalam penelitian analisa perbandingan performansi protokol *Routing* pada WMN terdiri dari kebutuhan *fungsional* dan *non-fungsional*. Kebutuhan *fungsional* yang terkait meliputi:

a. Terbentuknya jaringan WMN.
b. Terpasangnya protokol *Routing* OLSR, BATMAN, dan Babel.
c. Dapat melakukan pengiriman data dengan mereplikasikan masing-masing protokol *Routing* secara bergantian.

Analisa kebutuhan *non-fungsional* diperlukan untuk mengetahui kebutuhan penelitian mengenai perangkat keras dan perangkat lunak. Kebutuhan *non-fungsional* yang diperlukan pada penelitian ini antara lain:

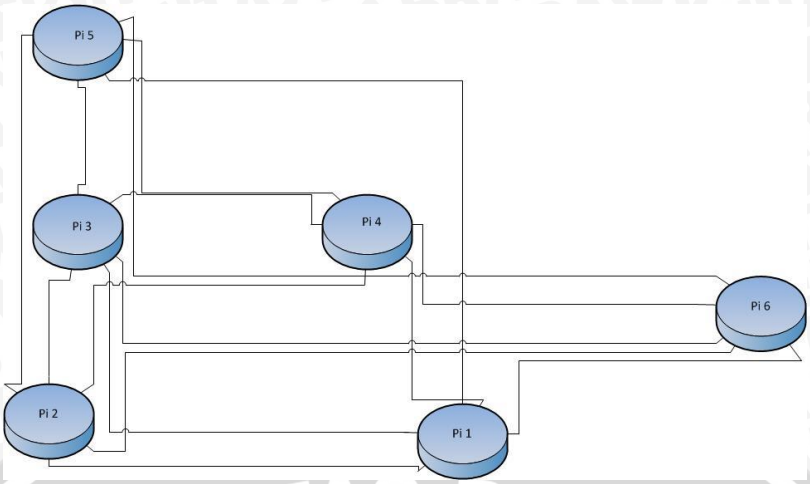
a. Kebutuhan perangkat keras:
   - 3 unit *Raspberry Pi* model B
   - 3 unit *Raspberry Pi* model B+
   - 6 unit microsd 8Gb
   - 6 unit *wifi-dongle* TP-link
   - 6 unit adaptor
   - 1 unit monitor
   - 1 unit *keyboard*
   - 1 unit kabel HDMI
b. Kebutuhan perangkat lunak:
   - Debian
   - Babel
   - OLSR
   - BATMAN
   - Iperf

3.3 Lingkungan Penelitian

Lingkungan penelitian digunakan untuk menentukan spesifikasi pembangunan jaringan. Lingkungan penelitian mencakup pembangunan WMN dan konfigurasi dalam penelitian analisa perbandingan performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan mesh. Jaringan WMN diperlukan sebagai wadah melakukan analisa perbandingan performansi protokol *Routing*. Lokasi penelitian dilakukan digedung C FILKOM UB. Pemilihan lokasi berdasarkan kemudahan dalam melakukan kontrol keamanan *Raspbery Pi* yang terpasang secara permanen pada Gedung C FILKOM UB.

3.3.1 Topologi Jaringan

Dalam pembangunan WMN diperlukan lebih dari 2 *node* untuk menciptakan adanya komunikasi antar *node* yang bersifat *multi-hoop*. Pengurangan *interferensi signal* dilakukan dengan mengatur *channel* pada setiap *node* menggunakan *channel* 4. Topologi WMN dapat dilihat pada gambar 3.2.



Gambar 3.2 Topologi WMN

3.4 PENGUJIAN

Pengujian pada penelitian ini terdiri dari beberapa tahap. Pengujian dilakukan pada semua *node* yang berada dalam jaringan WMN. Pengujian digolongkan berdasarkan komunikasi *node* yakni: *single-hoop* dan *multi-hoop*.

Pengiriman paket data dilakukan sebanyak 100 kali dalam sekali pengiriman paket UDP. Variasi pengiriman data pada satu skenario pengiriman paket UDP terdiri dari 5 interval. Besar paket data pada pengiriman paket UDP yakni 13MB, 16MB, 19MB, 22MB dan 25MB.

Analisa perbandingan performansi Protokol *Routing* OLSR, BATMAN dan Babel didapatkan menggunakan QOS sebagai parameter perbandingan. QOS yang digunakan adalah Throughput(MBps), Jitter(ms) dan Packet Loss(%). Akumulasi QOS dari semua hasil pengujian yang didapatkan, akan diambil nilai rata-ratanya dan kemudian dibandingkan diantara ketiga Protokol tersebut. Dengan demikian akan didapatkan Protokol *Routing* terbaik untuk diimplementasikan pada *WMN*.

3.5 ANALISIS

Analisa data diambil dari pengujian yang dilakukan pada masing-masing tahapan pengujian berupa QOS(Throughput(MBps), Jitter(ms), dan Packet Loss(%)). Data pengujian berupa QOS akan diolah sebagai penentu tindakan selanjutnya. Data QOS yang telah didapat, akan diambil nilai rata-ratanya pada setiap protokol *Routing* berdasarkan komunikasi *single-hoop* dan *multi-hoop*. Data yang telah diolah digunakan sebagai dasar dari analisa perbandingan performansi protokol *Routing* OLSR, BATMAN dan Babel.

3.6 KESIMPULAN

Pada tahap ini akan dilakukan pengambilan kesimpulan dari hasil analisis perbandingan performansi Protokol *Routing*. Kesimpulan akan memberikan akurasi atas rumusan masalah yang ada.

BAB 4 LINGKUNGAN PENELITIAN

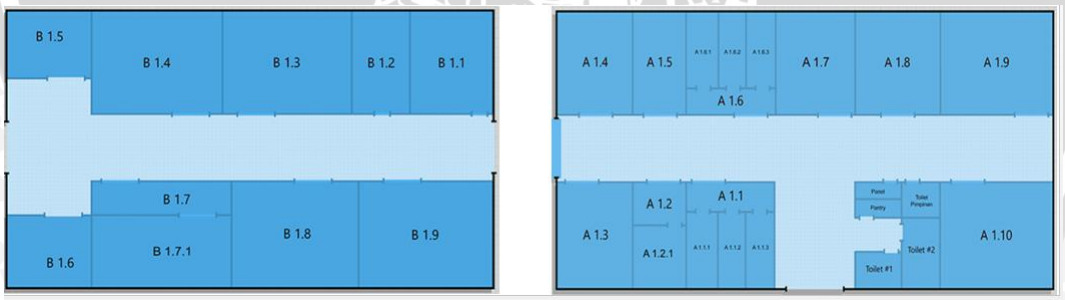
Pada bab ini dijelaskan mengenai tahapan pembangunan *Wireless Mesh Network*(WMN) menggunakan *Raspberry Pi* sebagai *node mesh client*. Dengan tujuan utama melakukan pengujian perbandingan performansi protokol *Routing* pada jaringan *Mesh*. Berikut merupakan langkah-langkah yang akan dilakukan dalam penelitian perbandingan performansi protokol *Routing* pada WMN, diantaranya*:*

1. Pembangunan lingkungan WMN
2. Konfigurasi WMN
3. Instalasi paket dan konfigurasi protokol *Routing* OLSR
4. Instalasi paket dan konfigurasi protokol *Routing* BATMAN
5. Instalasi paket dan konfigurasi protokol *Routing* Babel
6. Instalasi *Iperf*
7. Pembentukan Skenario Pengujian

4.1 Pembangunan Lingkungan WMN

Pembangunan WMN dilakukan dengan memanfaatkan 6 Raspberry Pi, 6 Wifi-dongle, 6 koneksi internet dan 6 MicroSD sebagai node client pada WMN. Lokasi pembangunan WMN berada pada 2 Lingkungan yang berbeda.

Lingkungan pertama, WMN dibangun non permanen di Gedung A dan B FILKOM sehingga setiap client dapat dirubah letak dan posisinya. Selain itu ketika dilakukan pemasangan secara non permanen, peneliti diharuskan mendatangi setiap perangkat untuk melakukan konfigurasi dan menjalankan client. Denah lokasi pemasangan node WMN dapat dilihat pada gambar 4.1.



Gambar 4.1 Denah Gedung B dan A FILKOM

Lingkungan yang kedua, *node* WMN dipasang secara permanen pada beberapa ruangan di Gedung C FILKOM. Perancangan desain lokasi dilakukan untuk penempatan masing-masing *node* WMN. Jarak antar *node* perlu diperhatikan dalam melakukan perancangan desain lokasi penempatan *node* WMN. Jarak antar *node* berpengaruh pada cakupan *signal wireless(wifi-dongle)* yang digunakan pada pembangunan *node* WMN. Kabel UTP diperlukan pada setiap ruangan untuk dihubungkan dengan eth0 pada *Raspberry Pi* agar dapat melakukam *remote* dengan laptop atau komputer sehingga memudahkan akses dan pemantauan setiap *node-client mesh* di setiap ruangan yang terpasang *Raspberry Pi*. *Node*

WMN juga dapat di *remote* dengan menggunakan terminal menggunakan SSH dengan ketentuan komputer/Laptop yang digunakan terhubung dalam jaringan internet UB. Denah lokasi pemasangan *node* WMN pada gedung C FILKOM dapat dilihat pada gambar 4.2.

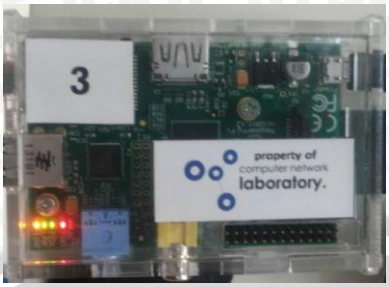
Gambar 4.2 Denah Ruangan Gedung C FILKOM UB

Berdasarkan denah pada gambar 4.2 ditentukan lokasi pemasangan *node* WMN pada ruangan yang diperlukan bertempat di Gedung C FILKOM UB. Sebelum melakukan pemasangan *node* WMN, diberikan penomoran fisik pada setiap kasing *Raspberry Pi*. Penomoran pada kasing *Raspberry Pi* dimulai dari 1-6 dengan tujuan dapat dengan mudah dikenali. Tujuan utama dari aktivitas melakukan *remote node* WMN yaitu dalam melakukan penelitian mengenai perbandingan performansi protokol *Routing* OLSR, BATMAN dan Babel dapat dilakukan dengan menggunakan komputer/laptop dengan ketentuan terhubung pada jaringan *internet* UB. Penomoran fisik pada setiap kasing *Raspberry Pi* dapat dilihat pada tabel 4.1.

Tabel 4.1 Penomoran *Raspberry Pi* dan pemasangan di ruangan

Nomor Raspberry Pi	Ruang
1	C1.12
2	C1.14
3	C1.11
4	Ruang Panel
5	C1.9
6	C1.6

Gambar penomoran fisik pada setiap kasing *Raspberry Pi* dapat dilihat pada gambar 4.3.



Gambar 4.3 Contoh Peno,oran *Raspberry Pi*

4.2 Konfigurasi Wireless Mesh Network

Sebelum melakukan pemasangan *node* permanen, perlu dilakukan konfigurasi *network interface* dan *hostname* pada setiap perangkat *Raspberry Pi*. Konfigurasi *hostname,* disesuaikan berdasarkan ruang pemasangan *Raspberry Pi*. Konfigurasi *hostname* dilakukan pada *file* /etc/hostname dan /etc/hosts. Setelah selesai melakukan pergantian *hostname* langkah selanjutnya adalah melakukan *restart* pada *Raspberry Pi*. Konfigurasi *hostname* dapat dilihat pada gambar 4.4.

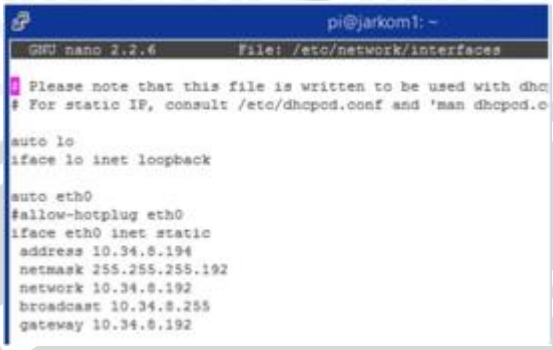
Gambar 4.4 Konfigurasi *Hostname Raspberry Pi*

Langkah selanjutnya dilakukan konfigurasi alamat eth0 dengan menggunakan pengalamatan statik. Pengalamatan statik dimaksudkan untuk mematenkan pengalamatan IP eth0 pada *Raspberry Pi*, sehingga memudahkan *user* dalam mengakses *Raspberry Pi*. Alamat IP eth0 disesuaikan dengan nomor *Raspberry Pi* yang dipasang pada *subnet* jaringan di setiap ruangan Gedung C FILKOM. *Subnet* jaringan yang digunakan dalam pemasangan *Raspberry Pi* di gedung C FILKOM adalah 26. Pengalamatan statik eth0 pada masing – masing *Raspberry Pi* dapat dilihat pada tabel 4.2.

Tabel 4.2 Alamat Ip eth0 pada setiap Raspberry

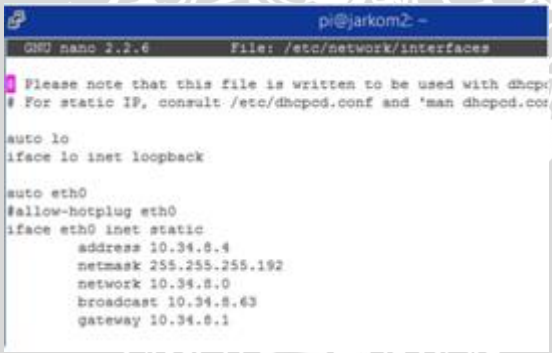
Nomor Raspberry Pi	Hostname	Ruang	Alamat Ip
1	Jarkom1	C1.12	10.34.8.194/26
2	Jarkom2	C1.14	10.34.8.4 /26
3	Jarkom3	C1.11	10.34.19.3 /26
4	Jarkom4	Ruang Panel	10.34.8.6 /26
5	Jarkom5	C1.9	10.34.8.196/26
6	Jarkom6	C1.6	10.34.9.66/26

Konfigurasi *Network Interfaces* pada dilakukan pada *file* /etc/network/interfaces. Konfigurasi *Network Interfaces* dilakuakan pada semua *node*. Pada *node* 1, dilakukan pengaturan *Network Interfaces* dengan isi menentukan *Interface* eth0 dalam kondisi *static*, *IP Address* adalah 10.34.8.194, *netmask* : 255.255.255.192, *network*: 10.34.8.192, *broadcast:* 10.34.8.255 dan *gateway*: 10.34.8.192. Untuk lebih jelasnya, dapat dilihat pada gambar 4.5.



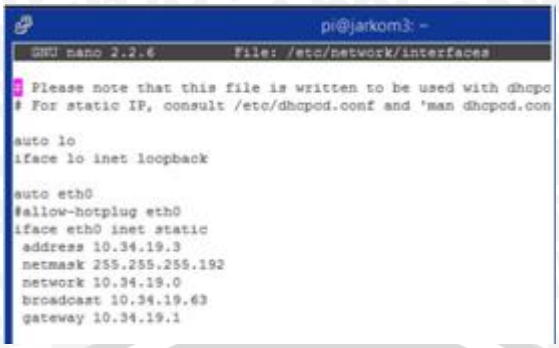
Gambar 4.5 *Konfigurasi* network interfaces eth0 node *1*

Konfigurasi *Network Interfaces* pada *node* 2 dengan melakukan pengaturan *Interface* eth0 dalam kondisi *static*, *IP Address* adalah 10.34.8.4, *netmask* : 255.255.255.192, *network*: 10.34.8.0, *broadcast:* 10.34.8.63 dan *gateway*: 10.34.8.1. Untuk lebih jelasnya, dapat dilihat pada gambar 4.6.



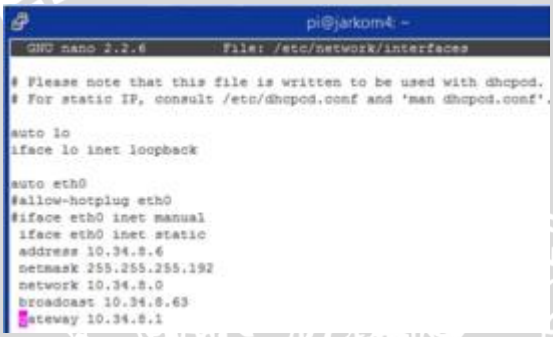
Gambar 4.6 Konfigurasi interface eth0 *node* 2

Konfigurasi *Network Interfaces* pada *node* 3 dengan melakukan pengaturan *Interface* eth0 dalam kondisi *static*, *IP Address* adalah 10.34.19.3, *netmask* : 255.255.255.192, *network*: 10.34.19.0, *broadcast:* 10.34.19.63 dan *gateway*: 10.34.19.1. Untuk lebih jelasnya, dapat dilihat pada gambar 4.7.



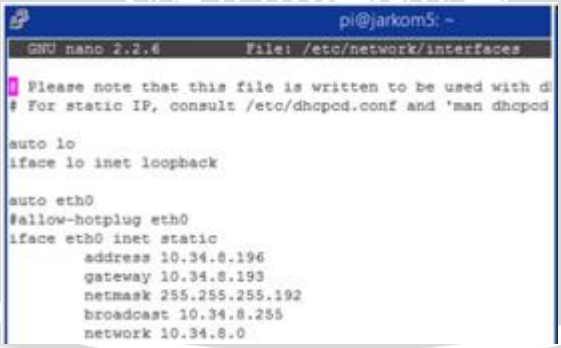
Gambar 4.7 Konfigurasi interface eth0 *node* 3

Konfigurasi *Network Interfaces* pada *node* 3 dengan melakukan pengaturan *Interface* eth0 dalam kondisi *static*, *IP Address* adalah 10.34.8.6, *netmask* : 255.255.255.192, *network*: 10.34.8.0, *broadcast:* 10.34.8.63 dan *gateway*: 10.34.8.1. Untuk lebih jelasnya, dapat dilihat pada gambar 4.8.



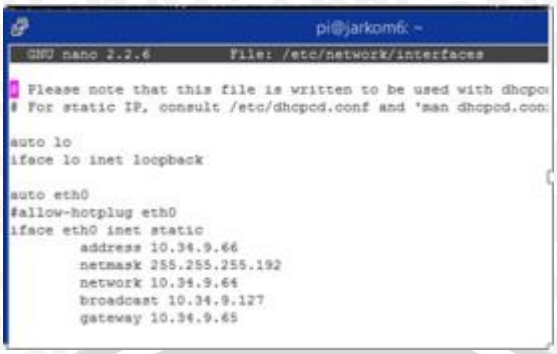
Gambar 4.8 Konfigurasi interface eth0 *node* 4

Konfigurasi *Network Interfaces* pada *node* 3 dengan melakukan pengaturan *Interface* eth0 dalam kondisi *static*, *IP Address* adalah 10.34.8.196, *netmask* : 255.255.255.192, *network*: 10.34.8.0, *broadcast:* 10.34.8.255 dan *gateway*: 10.34.8.193. Untuk lebih jelasnya, dapat dilihat pada gambar 4.9.



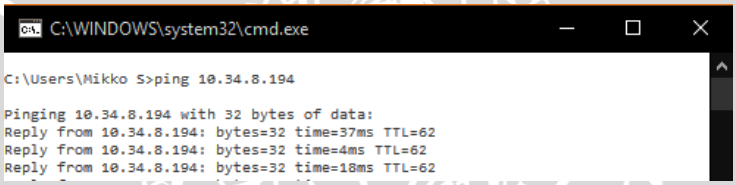
Gambar 4.9 Konfigurasi interface eth0 *node* 5

Konfigurasi *Network Interfaces* pada *node* 3 dengan melakukan pengaturan *Interface* eth0 dalam kondisi *static*, *IP Address* adalah 10.34.9.66, *netmask* : 255.255.255.192, *network*: 10.34.9.64, *broadcast:* 10.34.9.127 dan *gateway*: 10.34.8.9.65. Untuk lebih jelasnya, dapat dilihat pada gambar 4.10.



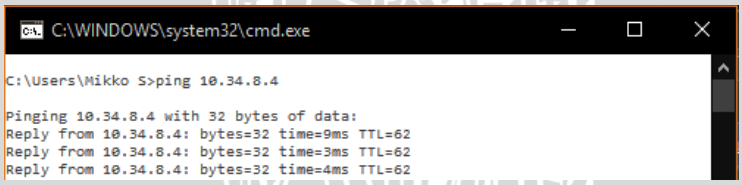
Gambar 4.10 Konfigurasi interface eth0 *node* 6

Konfigurasi statis tidak dibutuhkan untuk perangkat komputer/laptop dalam melakukan *remote* pada *Raspberry Pi*. Perangkat komputer/laptop cukup menggunakan konfigurasi dinamis dan terhubung dalam jaringan Lokal UB. Untuk memastikan apakah setiap *node-client* dapat berjalan dengan baik setelah pemasangan permanen, dilakukan ping ke setiap *node* melalui IP eth0 yang telah dilakukan konfigurasi. Pengecekan pada *node* 1 dilakukan dengan perintah "ping 10.34.8.194". Visualisasi proses ini dapat dilihat pada gambar 4.11.



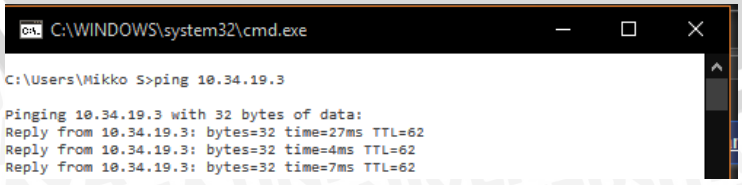
Gambar 4.11 Ping Laptop ke *node* 1

Pengecekan pada *node* 2 dilakukan dengan perintah "ping 10.34.8.4". Visualisasi proses ini dapat dilihat pada gambar 4.12.



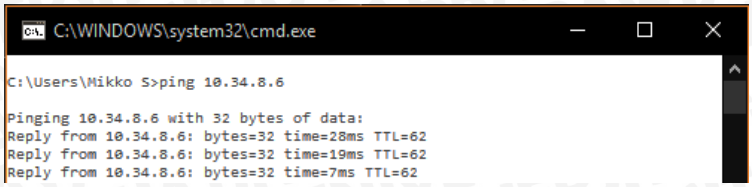
Gambar 4.12 Ping Laptop ke *node* 2

Pengecekan pada *node* 3 dilakukan dengan perintah "ping 10.34.19.3". Visualisasi proses ini dapat dilihat pada gambar 4.13.



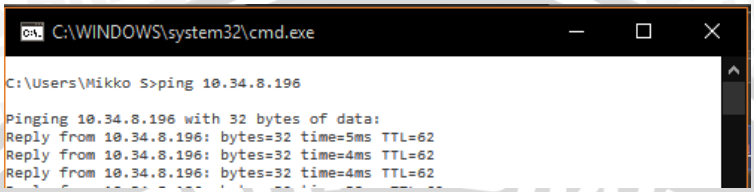
Gambar 4.13 Ping Laptop ke *node* 3

Pengecekan pada *node* 4 dilakukan dengan perintah "ping 10.34.8.6". Visualisasi proses ini dapat dilihat pada gambar 4.14.



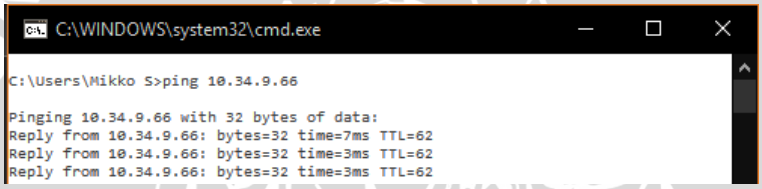
Gambar 4.14 Ping Laptop ke *node* 4

Pengecekan pada *node* 5 dilakukan dengan perintah "ping 10.34.8.196". Visualisasi proses ini dapat dilihat pada gambar 4.15.



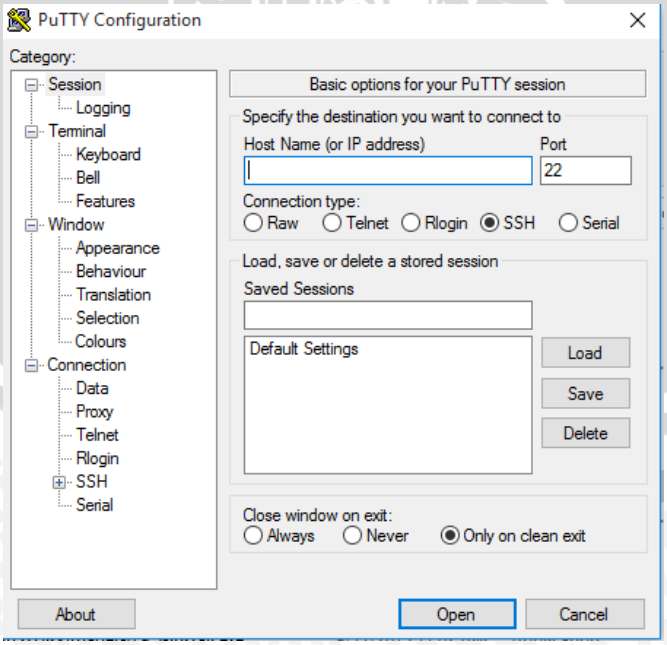
Gambar 4.15 Ping Laptop ke *node* 5

Pengecekan pada *node* 6 dilakukan dengan perintah "ping 10.34.9.66". Visualisasi proses ini dapat dilihat pada gambar 4.16.



Gambar 4.16 Ping Laptop ke *node* 6

Remote pada setiap *node* WMN menggunakan perangkat dengan system operasi *windows*, diperlukan perangkat tambahan. Perangkat tambahan yang dibutuhkan adalah *software putty*. *Interface putty* dapat dilihat pada gambar 4.17.



Gambar 4.17 Software Putty

Bagian *Wireless Interface* dilakukan konfigurasi secara statik, bertujuan untuk pengiriman data pada setiap *node* WMN. Hal ini dikarenakan *node-server* memiliki pengalamatan IP yang tetap, sehingga *node-client* dapat mengetahui dengan pasti alamat server yang akan dituju. Pengaruh dari koneksi kabel LAN yang terpasang ketika akan konfigurasi *interface wireless* pada *Raspberry Pi* dapat dihindari dengan menjalankan perintah "service ifplugd stop" pada terminal *Raspberry Pi*. Perintah ifconfig wlan0 down digunakan untuk mematikan aktifitas wlan0 sebelum melakukan konfigurasi *interface wireless* pada *Raspberry Pi*. *Channel Wireless* yang akan digunakan pada jaringan mesh dapat dilakukan dengan melakukan analisis *wifi* dengan menggunakan *tools Wi-Fi Analyzer*. Tools *Wi-Fi Analyzer* memungkinkan *user* untuk: melihat trafik dari keadaan kondisi jaringan pada area tertentu, mengetahui *channel* yang sudah digunakan. Hasil analisis dengan *tools Wi-Fi Analyzer* yang telah dilakukan, ditentukan *ssid* adalah *mesh* dan *channel* adalah 4 yang akan digunakan karena *ssid mesh* dan *channel* 4 minim terhadap interferensi. *Kemudian setup ssid*, *channel* dan *mode* dengan perintah iwconfig wlan0 mode ad-hoc channel 4 essid mesh. Langkah selanjutnya adalah mengaktifkan *interface* wlan0 menggunakan perintah ifconfig wlan0 up dan iwconfig untuk melihat hasil konfigurasinya. Hasil konfigurasi yang telah dilakukan dapat dilihat pada gambar 4.18.

Gambar 4.18 Konfigurasi *Wireless* pada *Raspberry*

Yang perlu diperhatikan ketika melakukan konfigurasi *Wireless* pada setiap *Raspberry Pi* adalah hasil dari konfigurasi, pastikan *mode*, *frequency*, *essid* dan *cell* satu perangkat dengan perangkat lainnya sama. Hasil dari konfigurasi dapat dilihat pada gambar 4.18. Jika terdapat perbedaan hasil dari perintah *iwconfig* pada setiap *node*, maka hubungan antar *node* tidak dapat dilakukan.

4.3 Instalasi Paket dan Konfigurasi *Routing* OLSR

OLSR adalah protokol *Routing* proaktif yang akan diimplementasikan dan dilakukan uji performansi pada WMN. OLSR merupakan protokol *Routing* yang bekarja pada layer 3. Adapun pertimbangan untuk penerapan OLSR pada WMN adalah karakteristik WMN yang dapat menjaga kualitas jaringan. Hal ini didukung oleh kemampuan WMN yaitu *self-heling* dan *self-configuring*.

4.3.1 Instalasi Paket *Routing* OLSR

Paket OLSR dapat di *download* pada http://www.olsr.org/?q=download. Proses instalasi paket *Routing* OLSR dengan mengikuti langkah-langkah pada http://wiki.samsul.web.id/linux/Dokumentasi.OLSR.di.Destika.2014. Langkah-langkah yang digunakan pada tahap instalasi paket adalah :

1.	Sudo aptitude install olsrd olsrd-plugins
2.	sudo aptitude install kernel-package libncurses5-dev fakeroot wget \ bzip2 g++ libssl-dev libxml2-dev doxygen bison flex libc6
3.	tar -xvjf olsrd-0.6.2.tar.bz2
4.	cd olsrd-0.6.2
5.	make all
6.	sudo make install

Pada tahap ini, proses instalasi paket *Routing* OLSR telah selesai dilakukan. Perintah "olsrd" pada terminal *Raspberry Pi* digunakan untuk memastikan instalasi paket OLSR berhasil. OLSR dapat dioperasikan secara optimal ketika konfigurasi *Routing*-nya telah terpenuhi. Untuk itu, akan dilakukan konfigurasi *Routing* pada OLSR terlebih dahulu.

4.3.2 Konfigurasi Paket *Routing* OLSR

Tutorial yang digunakan untuk melakukan konfigurasi *Routing* OLSR dapat dilihat pada http://wiki.samsul.web.id/linux/Dokumentasi.OLSR.di.Destika.2014. Langkah-lanhkah yang harus dilakukan dalam konfigurasi paket OLSR adalah:

1.	sudo cp /etc/olsrd.conf /etc/olsrd.conf.orig
2.	sudo vim /etc/olsrd.conf

Kemudian lakukan penyesuaan pada blok paling bawah dengan memastikan *interface* aktif merupakan *interface* yang digunakan. Pada kasus ini, penulis menggunakan "wlan0", untuk lebih jelasnya dapat dilihat pada gambar 4.19.

```
Interface "wlan0"
{
    # Interface Mode is used to prevent unnecessary
    # packet forwarding on switched ethernet interfaces
    # valid Modes are "mesh" and "ether"
    # (default is "mesh")

    # Mode "mesh"
}
```

Gambar 4.19 Tampilan *Interface* Wlan0.

Ubah isi dari file "olsrd.conf" pada bagian "LoadPlugin "olsrd_txtinfo.so.0.1"" untuk di-non aktifkan dengan menambahkan tanda "#" di depan pada tiap-tiap barisnya. *File* "olsrd.conf" dapat dibuka dengan melakukan seperti pada poin 3.

3.	/etc/olsrd.conf

Contoh file "olsrd.conf" dapat dilihat pada gambar 4.20.

```
# LoadPlugin "olsrd_txtinfo.so.0.1"
# {
    # port number the txtinfo plugin will be listening, default 2006
#   PlParam     "port"    "81"
    # ip address that can access the plugin, use "0.0.0.0"
    # to allow everyone
#    PlParam     "Accept"    "127.0.0.1"
#}
```

Gambar 4.20 Pengaturan *File* Olsrd.conf.

Setelah ini, OLSR dapat digunakan dengan mengaktifkan perintah yang ada pada poin 4.

4.	sudo olsrd -f /etc/olsrd.conf atau cukup dengan perintah sudo olsrd

4.4 Instalasi Paket dan Konfigurasi *Routing* BATMAN

BATMAN merupakan salah satu protokol *Routing* yang akan diimplementasikan dan dilakukan uji performansi pada WMN. BATMAN digolongkan sebagai proaktif protokol *Routing* yang bekerja pada layer 2. Protokol *Routing* BATMAN dapat diterapkan pada WMN dengan pertimbangan karakteristik WMN yang menjaga kualitas jaringan.

4.4.1 Instalasi Paket *Routing* BATMAN

Instalasi protokol *Routing* BATMAN dilakukan dengan mengunduh paket Batman-adv pada website https://www.open-mesh.org/projects/open-mesh/wiki/Download. Proses *debugging* dan konfigurasi modul kernel batman-adv memerlukan batctl tool. Batctl tool bisa diperoleh dengan mengetikkan perintah "apt-get install batctl" pada terminal *Raspbian*(OS *Raspberry Pi*). Langkah-langkah yang harus dilakukan dalam proses instalasi batman-adv adalah:

1.	apt-get install batctl
2.	modprobe batman-adv

Untuk lebih jelasnya dapat dilihat pada gambar 4.21.

```
root@jarkom1:/home/pi# apt-get install batctl
Reading package lists... Done
Building dependency tree
Reading state information... Done
batctl is already the newest version.
0 upgraded, 0 newly installed, 0 to remove and 3 not upgraded.
root@jarkom1:/home/pi# modprobe batman-adv
```

Gambar 4.21 Instalasi Paket batman-adv.

Setelah selesai melakukan proses instalasi paket batman-adv, penulis melakukan konfigurasi interface bat0 dengan batman-adv yang akan dijelaskan lebih lanjut pada sub bab 4.4.2.

4.4.2 Konfigurasi Bat0

Bat0 merupakan *interface* tambahan pada BATMAN yang berfungsi sebagai sarana dalam memenuhi pengalamatan logika oleh batman-adv pada layer-3. Konfigurasi bat0 dapat dilakukan dengan:

1.	batctl if add eth0
2.	ifconfig wlan0 down
3.	batctl if add wlan0
4.	ifconfig wlan0 up
5.	ifconfig bat0 up

Proses konfigurasi batman dapat dilihat pada gambar 4.22.

```
root@jarkom1:/home/pi# batctl if add eth0
root@jarkom1:/home/pi# ifconfig wlan0 down
root@jarkom1:/home/pi# batctl if add wlan0
root@jarkom1:/home/pi# ifconfig wlan0 up
root@jarkom1:/home/pi# ifconfig bat0 up
root@jarkom1:/home/pi#
```

Gambar 4.22 Konfigurasi bat0.

Pengecekan *originator*, *traceroute* dan *ping* menggunakan perintah "batctl o" ," batctl tr alamat_ip_tetangga_node", "batctl p alamat_ip_tetangga_node" pada terminal *Raspbian* dilakukan untuk memastikan semua paket batman terlah terinstall dan terkonfigurasi dengan benar. Lebih jelasnya dapat dilihat pada gambar 4.23**.**

```
root@jarkom1:/home/pi# batctl o
[B.A.T.M.A.N. adv 2015.0, MainIF/MAC: eth0/b8:27:eb:77:1c:3a (bat0 BATMAN_IV)]
  Originator      last-seen (#/255)           Nexthop [outgoingIF]:   Potential
nexthops ...
No batman nodes in range ...
```

```
root@jarkom5:/home/pi# batctl tr 169.254.60.134
traceroute to 169.254.60.134 (b8:27:eb:5b:22:bb), 50 hops max, 20 byte packets
 1: b8:27:eb:77:1c:3a  0.412 ms  0.563 ms  0.381 ms
 2: b8:27:eb:5b:22:bb  1.380 ms  1.629 ms  1.536 ms
root@jarkom5:/home/pi#
```

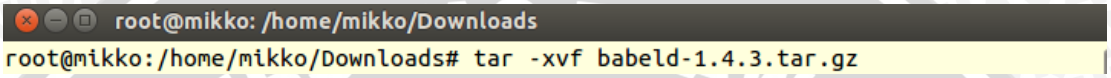
Gambar 4.23 Pengecekan protokol *Routing* BATMAN.

4.5 Instalasi Paket dan Konfigurasi *Routing* Babel

Babel merupakan salah satu protokol Routing yang akan diimplementasikan dan dan dilakukan uji performansi pada WMN. Protokol Routing Babel digolongkan sebagai protokol Routing proaktif yang bekerja pada layer 3. Dengan demikian, protokol Routing Babel dapat diaplikasikan pada WMN dengan pertimbangan karakteristik pada WMN yang menjaga kualitas jaringan.

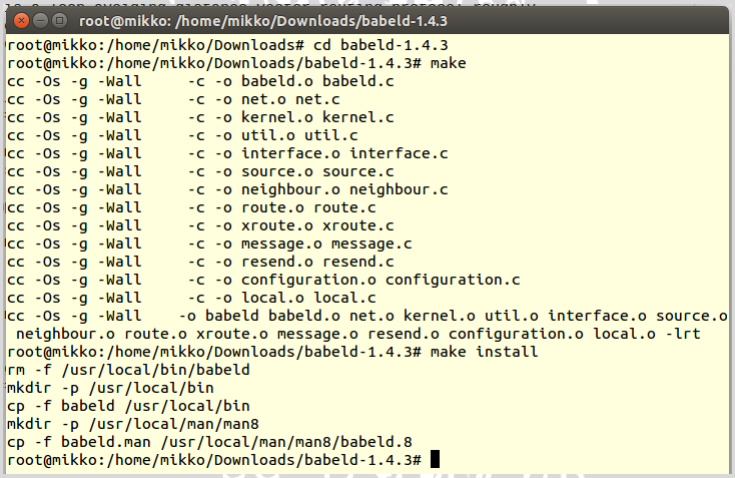
4.5.1 Instalasi Paket *Routing* Babel

Instalasi paket *Routing* Babel dilakukan dengan mengunduh paket di website http://www.pps.univ-paris-diderot.fr/~jch/software/files/. Proses *Downloads* paket disajikan dalam gambar 4.24.



Gambar 4.24 Proses Ekstrak paket Babel

Setelah paket berhasil diunduh, ekstrak paket dengan ekstensi tar.gz menggunakan perintah tar –xvjf nama_paket. Masuk kedalam direktori paket yang sudah di ekstrak. Jalankan perintah make dan make install untuk membangun dan *install* system. Proses kompilasi paket dapat dilihat pada gambar 4.25.

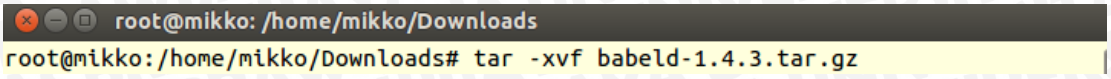


Gambar 4.25 Proses *compile* paket Babel

Paket *Routing* Babel yang sudah terpasang dapat dijalankan dengan perintah Babeld wlan0.

4.6 Instalasi Paket *Iperf*

Instalasi paket *Routing* Babel dilakukan dengan mengunduh paket di website http://www.pps.univ-paris-diderot.fr/~jch/software/files/. Proses *Downloads* paket disajikan dalam gambar 4.26.



Gambar 4.26 Proses Ekstrak Paket Babel

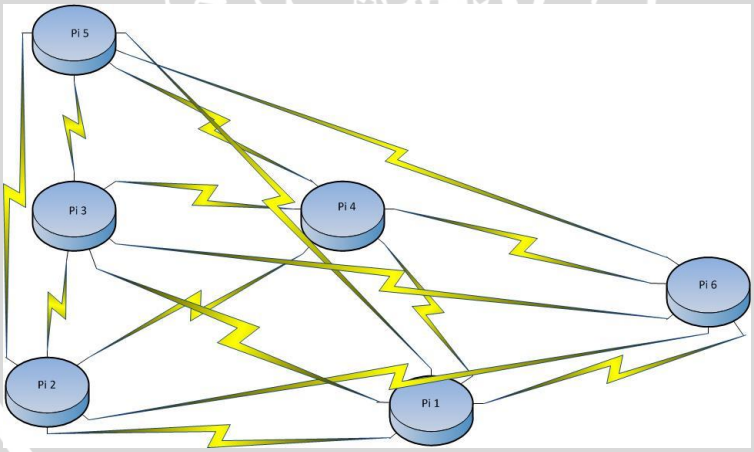
4.7 Pembentukan Skenario Pengujian

Pembentukan lingkungan pengujian dimaksudkan untuk mendapatkan data uji dengan menerapkan skenario pada lingkungan pengujian. Sehingga memperoleh data uji dari perbandingan performansi protokol *Routing* OLSR, BATMAN dan Babel pada WMN berupa hasil pengukuran QOS berupa *Throughput, Jitter* dan *Packet loss* yang berlokasi di gedung C FILKOM. Skenario pengujian dibuat dengan tujuan mendapatkan QOS. Pengujian dilakukan dengan 1 skenario pokok dan 2 skenario tambahan. Jumlah total skenario pengujian performansi protokol *Routing* OLSR, BATMAN dan Babel pada WMN adalah 3 yaitu:

1. Skenario pengujian 1;
2. Skenario pengujian 2;
3. Skenario pengujian 3.

4.7.1 Skenario Pengujian 1

Pada Skenario Pengujian 1 dilakukan pengiriman paket data menggunakan UDP. *Node* yang terlibat dalam sekenario ini ada 6 yang merupakan *node* WMN. Pengujian yang dilakukan yaitu masing-masing *node* WMN berinteraksi dengan semua *node* lain dalam jaringan WMN menggunakan paket UDP. Skenario ini diterapkan pada ketiga protokol *Routing* OLSR, BATMAN, dan Babel. Pengiriman paket UDP untuk satu kali tindakan dilakukan dalam kurun waktu 100 detik. Variasi besar paket UDP yang dikirimkan untuk 1 kali interaksi antar *node* adalah 13Mb, 16Mb, 19Mb, 22Mb, 25Mb. Satu variasi besar paket UDP, akan diambil 5 *sample* hasil berupa QOS. Visualisasi skenario 1 dapat dilihan pada gambar 4.27.



Gambar 4.27 Skenario Pengujian 1

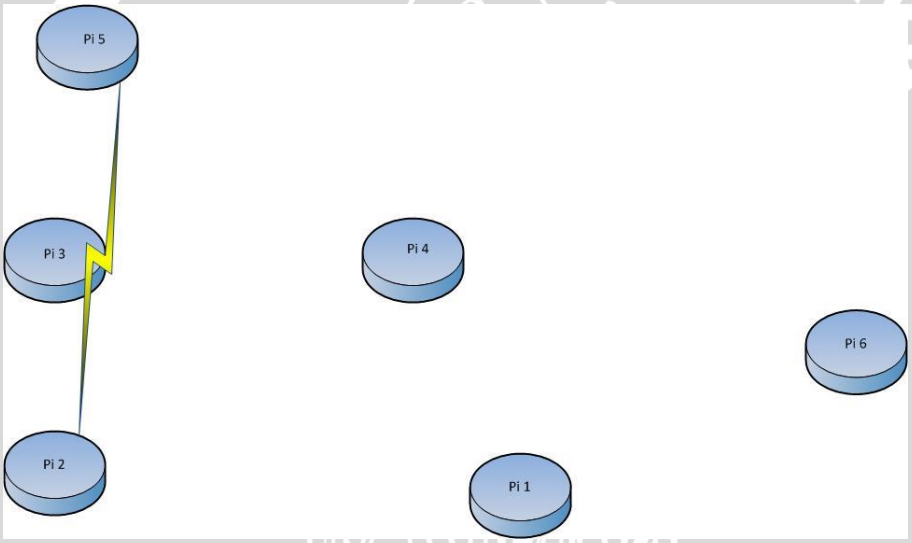
4.7.2 Skenario Pengujian 2

Skenario pengujian 2 merupakan skenario tambahan yang digunakan untuk mengetahui performansi prototokol *Routing* OLSR, BATMAN, dan Babel yang bekerja pada *WMN***.** Skenario pengujian ini akan melakukan pengiriman paket UDP pada protokol *Routing* OLSR, BATMAN, dan Babel dijalankan dalam keadaan *multihop* dengan jarak 1 *hop* tetangga *node*. *Multihop* dengan jarak 1 *hop* tetangga adalah interaksi antara *node client* dan *node server* dengan melewati 1

hop tetangganya. Pemutusan 1 *node* tetangga dilakukan ketika proses pengiriman paket UDP sedang berlangsung berjalan pada 30 detik waktu awal pengiriman paket. Waktu total yang digunakan dalam pengiriman paket adalah 100 detik. Variasi besar paket UDP yang dikirimkan adalah 13Mb, 16Mb, 19Mb, 22Mb, 25Mb yang masing-masing dari besaran pengiriman paket data akan diambil 5 *sample* berupa QOS. Data QOS yang diperoleh dari hasil pengujian, nantinya akan diolah dan dijadikan tolak ukur untuk perbandingan performansi protokol *Routing* OLSR, BATMAN, dan Babel pada *WMN.* Pada skenario ini akan dibagi menjadi 2 sub skenario pengujian.

4.7.2.1 Pengiriman Paket Antara *Node* 2 dan 5

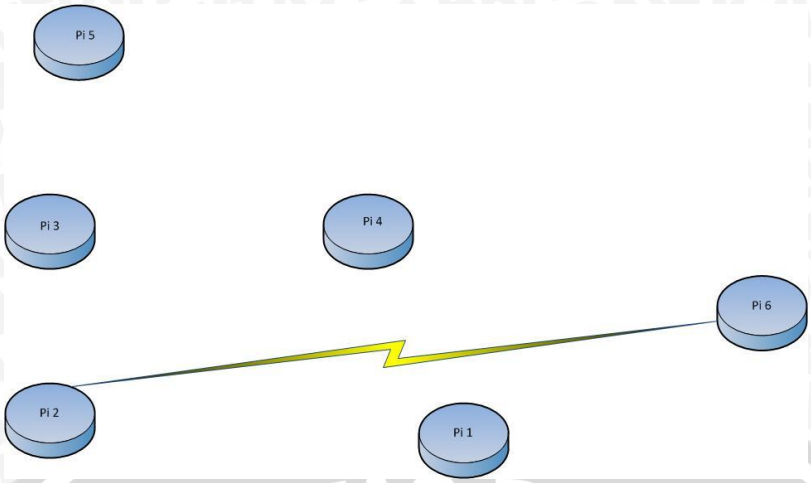
Pada sub skenario ini dilakukan pengiriman paket data UDP antara *node* 2 dan 5. Pengiriman paket dilakukan pada kondisi *multi-hop* dengan mematikan satu *node* diantara *node* 2 dan 5. Proses mematikan *node* akan dilakukan ketika pengiriman paket data sedang berjalan pada waktu 30s. *Node* 3 merupakan *node* yang dimatikan dalam sub skenario ini. Visualisasi sub skenario ini dapat dilihat pada gambar 4.28.



Gambar 4.28 Skenario Pengujian 2.1

4.7.2.2 Pengiriman Paket Antara *Node* 2 dan 6

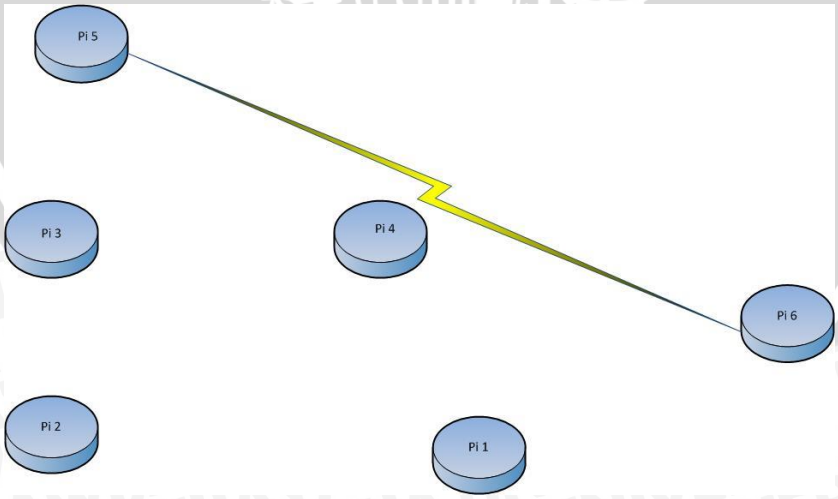
Pada sub skenario ini dilakukan pengiriman paket data UDP antara *node* 2 dan 6. Pengiriman paket data dilakukan pada keadaan *multi-hop* dengan mematikan 1 *node* yang berada diantara *node* 2 dan 6. Adapun *node* yang akan dimatikan adalah *node* 1. *Node* 1 akan dimatikan ketika proses pengiriman data berlangsung pada 30s. Visuali sub skenario dapat dilihat pada gambar 4.29.



Gambar 4.29 Skenario Pengujian 2.2

4.7.3 Skenario Pengujian 3

Pada skenario 3 dilakukan pengiriman paket UDP pada protokol *Routing* OLSR, BATMAN, dan Babel yang akan dijalankan dalam keadaan *multihop* dengan 2 *hop* tetangganya. Keadaan *multihop* yangdimaksudkan adalah dimana interaksi antara *node client* dan *node server* melewati 2 *hop* tetangganya. Ketika proses pengiriman paket UDP sedang berlangsung akan dilakukan pemutusan(mematikan) 2 *node* yang berada diantara *node client* dan *node server*. Pemutusan *node* dilakukan secara berurutan ketika pengiriman paket UDP berjalan pada detik ke 20 dan 30. Sedangkan variasi besar paket yang dikirimkan ialah 13Mb, 16Mb, 19Mb, 22Mb, 25Mb yang nantinya dari masing-masing besaran paket UDP akan diambil 5 *sample* data QOS. Ke-5 *sample* data QOS yang telah diambil nantinya akan diolah pada bab hasil penelitian dan digunakan sebagai data perbandingan performansi protokol *Routing* OLSR, BATMAN, dan Babel pada WMN. Untuk lebih jelasnya, dapat dilihat pada gambar 4.30.



Gambar 4.30 Skenario Pengujian 3

4.8 Tabel dan Informasi *Routing* OLSR, BATMAN dan Babel

Pada sub bab ini kan menunjukkan tabel *Routing* dari masing-masing Protokol *Routing* OLSR, BATMAN dan Babel pada jaringan MESH. Tabel *Routing* merupakan informasi yang dimiliki oleh masing-masing *node* WMN mengenai keberadaan *node* tetangganya. Dengan adanya tabel *Routing*, dapat membantu proses pengiriman data untuk mendapatkan jalur tercepat menuju *node* tujuan.

4.8.1 Tabel *Routing* OLSR

Tabel *Routing* OLSR merupakan informasi keberadaan *node* lain yang terdeteksi dalam jaringan WMN. Pada gambar 4.31 ditampilkan informasi *node* tetangga yang terdeteksi sebagai tetangga *single-hop* adalah *node* 3,4. Sedangkan yang terdeteksi sebagai tetangga *multi-hop* adalah *node* 2 dengan jalur alternatif menggunakan *node* 3 dan *node* 1 dengan jalur alternatif *node* 3,4.

```
       *** olsr.org -  0.6.2-git_-hash_b379904453f69417b8cb357fdb06e7af  - (201
-10-20 07:06:25 on raspberrypi) ***

--- 23:00:26.651490 ------------------------------------------------- LINKS

IP address       hyst         LQ       ETX
192.168.1.3      0.000  1.000/0.940    1.063
192.168.1.4      0.000  0.689/0.000    INFINITE

--- 23:00:26.651738 ---------------------- TWO-HOP NEIGHBORS

IP addr (2-hop)  IP addr (1-hop)  Total cost
192.168.1.2      192.168.1.3      2.983
192.168.1.1      192.168.1.4      INFINITE
                 192.168.1.3      4.915
```

Gambar 4.31 Tabel *Routing* OLSR

4.8.2 Informasi *Routing* BATMAN

Pada Protokol *Routing* BATMAN tidak terdapat tabel *Routing*. Informasi *Routing* pada BATMAN terletak di masing-masing *node* tetangga yang dikenali dalam jaringan. Untuk menentukan jalur tercepat dalam melakukan pengiriman paket data, BATMAN melakukan *broadcast* pesan OGM ke semua *node* yang terdeteksi dalam jaringan. *Node* yang memberikan respon paling baik akan dipilih sebagai tetangga terdekat dan begitu seterusnya sampai pada *node* tujuan. Untuk mengetahui jalur *Routing* terdekat dari *node* BATMAN 5 menuju *node* BATMAN 6 dapat dilakukan dengan mengetikkan perintah "batctl tr IP_addres" yang ditampilkan pada gambar 4.32. Pada kasus ini BATMAN menganggap jalur jalur terdekat dari *node* 5 menuju *node* 6 adalah *node* 3 dan *node* 1 dengan menampilkan pengalamatan IPV6.

```
root@jarkom5:/home/pi# batctl tr 169.254.60.134
traceroute to 169.254.60.134 (b8:27:eb:5b:22:bb), 50 hops max, 20 byte packets
 1: b8:27:eb:77:1c:3a  0.412 ms  0.563 ms  0.381 ms
 2: b8:27:eb:5b:22:bb  1.380 ms  1.629 ms  1.536 ms
root@jarkom5:/home/pi#
```

Gambar 4.32 Informasi *Routing* BATMAN

4.8.3 Tabel *Routing* Babel

Tabel *Routing* pada babel menyimpan data informasi keberadaan masing-masing *node* tetangga babel. Informasi *routing* digunakan dalam menentukan jalur terpendek ketika melakukan pengiriman paket data. Untuk lebih jelasnya, tabel *Routing* babel disajikan dalam gambar 4.33. Gambar 4.33 merupakan visualisasi tabel *Routing* pada *node* babel 2. Pada gambar ini, ditunjukkan *node* babel yang berjarak 1 hop antara lain *node* 3,1,6,5 dan 4. Sedangkan *node* babel yang berjarak 2 hop antara lain *node* 5 dengan melewati *node* 4, *node* 1 dengan alternatif jalur *node* 4 dan 6, *node* 4 dengan alternatif jalur *Routing* *node* 6,5,1 dan *node* 6 dengan alternatif jalur *node* 4 ,1.

```
pi@jarkom2: ~
IP address       hyst          LQ        ETX
192.168.1.3      0.000   0.721/0.372    3.720
192.168.1.1      0.000   0.886/1.000    1.128
192.168.1.6      0.000   0.944/0.776    1.362
192.168.1.5      0.000   0.407/0.000    INFINITE
192.168.1.4      0.000   0.940/0.553    1.921

--- 23:10:33.589891 ---------------------- TWO-HOP NEIGHBORS

IP addr (2-hop)  IP addr (1-hop)  Total cost
192.168.1.5      192.168.1.4      6.382
192.168.1.1      192.168.1.4      3.467
                 192.168.1.6      2.362
192.168.1.4      192.168.1.6      2.561
                 192.168.1.5      INFINITE
                 192.168.1.1      2.256
192.168.1.6      192.168.1.4      3.595
                 192.168.1.1      2.128
```

Gambar 4.33 Tabel *Routing* Babel

BAB 5 PENGUJIAN DAN ANALISIS

BAB 5 memuat hasil pengujian perbandingan performansi protokol *Routing* OLSR, BATMAN dan Babel pada WMN dengan mengikuti parameter uji yang sudah ditentukan untuk setiap skenario pengujian. Data yang didapat melalui pengujian perbandingan performasi protokol *Routing* OLSR, BATMAN dan Babel dilakukan analisis data yang dihasilkan saat pengujian. Tujuan penelitian ini untuk mengetahui diantara protokol *Routing* OLSR, BATMAN dan Babel yang memiliki performansi paling bagus jika diterapkan dalam WMN.

5.1 Pengujian Untuk masing-masing *Node*

Pengujian bertempat di Gedung C FILKOM UB dengan posisi masing-masing *node* seperti pada skenario yang telah ditetapkan sebelumnya. Pada sub bab ini akan dilakukan pengujian performansi protokol *Routing* OLSR BATMAN, dan Babel untuk 6 *node* WMN. Pengujian dilakukan dengan memastikan semua *node* aktif(hidup) dan tergabung pada satu WMN. Pengujian performansi dilakukan pada 3 protokol *Routing* OLSR, BATMAN, dan Babel secara bergantian. Protokol *Routing* yang akan digunakan disesuaikan dengan kebutuhan. Perintah `ps ax|greep nama_protokol_routing` digunakan untuk mengetahui protokol *Routing* yang saat ini aktif. Pengujian perbandingan performansi dilakukan dengan menggunakan pengiriman paket UDP. Dalam pengujian ini dilakukan 5 sub bab yang mendukung.

5.1.1 Perangkat yang dibutuhkan

Untuk melakukan pengujian, diperlukan beberapa perangkat pendukung. Kebutuhan perangkat pada pengujian ini akan digolongkan menjadi 2 bagian. Bagian pertama meliputi perangkat keras:

a. 6 *Node* WMN
b. PC/Laptop

Bagian kedua meliputi perangkat lunak:

a. Debian
b. OLSR
c. BATMAN
d. Babel
e. Iperf

5.1.2 Tujuan

Skenario pengujian ini bertujuan untuk mengetahui perbandingan performansi protokol *Routing* OLSR BATMAN, dan Babel pada WMN. Metode yang digunakan adalah dengan melakukan pengiriman paket data UDP dilakukan antar *node* WMN secara bergantian.

5.1.3 Prosedur Pengujian

Prosedur pengujian yang dilakukan dalam pengujian perbandingan performansi protokol *Routing* OLSR, BATMAN dan Babel pada WMN meliputi:

- Memastikan ke-6 *node* WMN telah aktif dengan salah satu protokol *Routing*.
- Persiapkan PC/laptop untuk melakukan pengujian dengan me-*remote node* WMN.
- Pengujian dilakukan mulai pukul 21.00-08.00 WIB.
- Pengiriman paket UDP antar *node* dengan variasi ukuran besar paket adalah 13MB, 16MB, 19 MB, 22MB, 25MB dengan 5 sampel data untuk setiap besar paket data.

Skenario ini berlaku pada masing-masing *node* WMN untuk berinteraksi dengan *node* lainnya.

5.1.4 Hasil

Hasil diperoleh dari pengujian performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan mesh. Hasil dibagi menjadi dua, pengujian performansi dengan keadaan *single-hop* dan pengujian performansi dengan keadaan *multi-hop*.

Hasil pengujian performansi *single-hop* didapatkan hasil berupa *Throughput*(MBps), *Jitter*(ms) dan *Packet Loss*(%) yang akan ditunjukkan oleh tabel 5.1.

Tabel 5.1 Pengujian Performansi Single-hop.

	Jitter(ms)	*Throughput*(MBps)	*Packet Loss*(%)
OLSR	48.04952	0.2012	33.26038
BATMAN	58.69408	0.44704	39.59874
Babel	52.77604	0.28652	37.72618

Hasil pengujian performansi *multi-hop* didapatkan hasil berupa *Throughput*(MBps), *Jitter*(ms) dan *Packet Loss*(%) yang akan ditunjukkan oleh tabel 5.2.

Tabel 5.2 Pengujian Performansi Multi-hop.

	Jitter(ms)	*Throughput*(MBps)	*Packet Loss*(%)
OLSR	92.98636	0.0368	77.024
BATMAN	43.59466	0.45256	52.50012
Babel	54.40935	0.1096	59.32

5.1.5 Analisis

Hasil pengujian performansi *single-hop* pada protokol *Routing* OLSR, BATMAN dan Babel didapatkan nilai rata-rata paling bagus adalah protokol *Routing* OLSR. Data pengujian performansi yang didapat adalah *Throuhput* 0.2012 MBps, *Jitter* 48.04952 ms, dan *Packet Loss* sebesar 33.26038%.

Hasil pengujian performansi *multi-hop* pada protokol *Routing* OLSR, BATMAN dan Babel, disimpulkan bahwa protokol *Routing* BATMAN memiliki performansi paling bagus. Nilai rata-rata dari pengujian performansi protokol *Routing* BATMAN berupa *Throughput* 0.45256 MBps, *Jitter* 43.59466 ms, dan *Packet Loss* sebesar 52.50012%.

5.2 Pengujian *multi-hop* dengan 1 *node* dimatikan

Pengujian yang dilakukan pada sub bab ini berbeda dengan sebelumnya, yakni dengan mematikan satu *node* yang berada diantara kedua *node* sebagai pengirim dan penerima data. Proses mematikan 1 *node* dilakukan ketika waktu pengiriman paket UDP *client-server* menginjak detik ke 30. Tujuan penulis melakukan skenario ini untuk mengetahui pengaruh pemutusan 1 *node* di tengah pengiriman paket UDP terhadap QOS yang didapat. Variasi besar paket UDP yang dikirimkan adalah 13MB, 16MB, 19 MB, 22MB, 25MB dengan 5 sampel data untuk setiap besar paket data.

5.2.1 Perangkat yang dibutuhkan

Untuk melakukan pengujian, dibutuhkan beberapa perangkat pendukung. Kebutuhan perangkat akan dibagi menjadi dua , meliputi perangkat keras dan perangkat lunak. Adapun kebutuhan perangkat keras adalah:

a. 6 *Node* WMN
b. PC/Laptop

Kebutuhan perangkat lunak yang harus terpenuhi meliputi:

a. Debian
b. OLSR
c. BATMAN
d. Babel
e. Iperf

5.2.2 Tujuan

Skenario ini bertujuan untuk mengetahui performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan mesh ketika berada dalam kondisi salah satu *node* yang berada diantara *node* pengirim dan penerima paket UDP dimatikan saat pengiriman paket berjalan 30s.

5.2.3 Prosedur pengujian

Prosedur pengujian yang dilakukan dalam pengujian perbandingan performansi protokol *Routing* OLSR, BATMAN dan Babel pada WMN meliputi:

- Memastikan ke-6 *node* WMN telah aktif dengan salah satu protokol *Routing*.
- Persiapkan PC/laptop untuk melakukan pengujian dengan me-*remote node* WMN.
- Pengujian dilakukan mulai pukul 21.00-08.00 WIB.
- Pengiriman paket UDP antar *node* dengan variasi ukuran besar paket adalah 13MB, 16MB, 19 MB, 22MB, 25MB dengan 5 sampel data untuk setiap besar paket data.
- Digunakan pada pengiriman paket UDP antara *node* 2 dan 5 dengan mematikan *node* 3 ketika proses pengiriman data berlangsung selama 30s.
- Digunakan pada pengiriman paket UDP antara *node* 2 dan 6 dengan mematikan *node* 1 ketika proses pengiriman data berlangsung selama 30s.

5.2.4 Hasil

Hasil diperoleh dari pengujian performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan mesh. Hasil pengujian akan disajikan dalam 2 tabel yang mewakili 2 sub skenario yang berbeda.

Hasil pengujian *multi-hop node* 2 dan 5 dengan mematikan *node* 3 dapat dilihat pada tabel 5.3.

Tabel 5.3 Pengujian *Multihop* 2 ke 5 dengan *Node* 3 dimatikan.

	Jitter(ms)	*Throughput*(MBps)	*Packet Loss*(%)
OLSR	36.91883	0.0984	54.32
BATMAN	9.44511	1.371	33.8543
Babel	85.13933	0.0416	84.33

Hasil pengujian *multi-hop node* 2 dan 6 dengan mematikan *node* 1 dapat dilihat pada tabel 5.4.

Tabel 5.4 Pengujian *Multihop* 2 ke 6 dengan *Node* 1 dimatikan.

	Jitter(ms)	*Throughput*(MBps)	*Packet Loss*(%)
OLSR	170.56668	0.0276	89.92
BATMAN	153.94188	0.0184	95.96
Babel	65.26408	0.064	89.72

5.2.5 Analisis

Hasil pengujian performansi *multi-hop* antara *node* 2 dan 5 disertai dengan mematikan *node* 1 pada protokol *Routing* OLSR, BATMAN dan Babel, disimpulkan bahwa protokol *Routing* BATMAN memiliki performansi paling bagus. Nilai rata-rata dari pengujian performansi protokol *Routing* BATMAN berupa *Throughput* 1.371 MBps, *Jitter* 9.44511 ms, dan *Packet Loss* sebesar 33.8543%.

Hasil pengujian performansi *multi-hop* antara *node* 2 dan 5 disertai dengan mematikan *node* 1 pada protokol *Routing* OLSR, BATMAN dan Babel, disimpulkan bahwa protokol *Routing* Babel memiliki performansi paling bagus. Nilai rata-rata dari pengujian performansi protokol *Routing* Babel berupa *Throughput* 0.064 MBps, *Jitter* 65.26408 ms, dan *Packet Loss* sebesar 89.72%.

5.3 Pengujian *Multihop* dengan 2 *Node* dimatikan

Pada sub bab ini dilakukan pengujian dengan mematikan 2 *node* yang berada diantara *node client* dan *server*. 2 *node* akan dimatikan ketika proses pengiriman paket UDP sedang berlangsung pada detik ke 20 dan 30 secara bergantian. Skenario ini bertujuan untuk mengetahui performansi masing-masing protokol *Routing* OLSR, BATMAN, dan Babel ketika 2 *node* yang terputus di tengah-tengah proses pengiriman paket UDP. Tolak ukur yang digunakan dalam uji performansi masing-masing protokol adalah QOS. Variasi besar paket UDP yang dikirim adalah 13MB, 16MB, 19 MB, 22MB, 25MB dengan 5 sampel data untuk setiap besar paket data. Skenario ini hanya digunakan pada pengiriman paket UDP antara *node* 5 dan 6 dengan *node* 1,4 dimatikan.

5.3.1 Perangkat yang dibutuhkan

Untuk melakukan pengujian, dibutuhkan beberapa perangkat pendukung. Kebutuhan perangkat akan dibagi menjadi dua , meliputi perangkat keras dan perangkat lunak. Adapun kebutuhan perangkat keras adalah:

a. 6 *Node* WMN
b. PC/Laptop

Kebutuhan perangkat lunak yang harus terpenuhi meliputi:

a. Debian
b. OLSR
c. BATMAN
d. Babel
e. Iperf

5.3.2 Tujuan

Skenario pengujian performansi protokol *routing* OLSR, BATMAN dan Babel pada jaringan mesh dengan mematikan 2 *node* antara *node* 5 dan 6 adalah untuk

mengetahui performansi dari masing-masing protokol *routing* jika berada dalam kondisi ini.

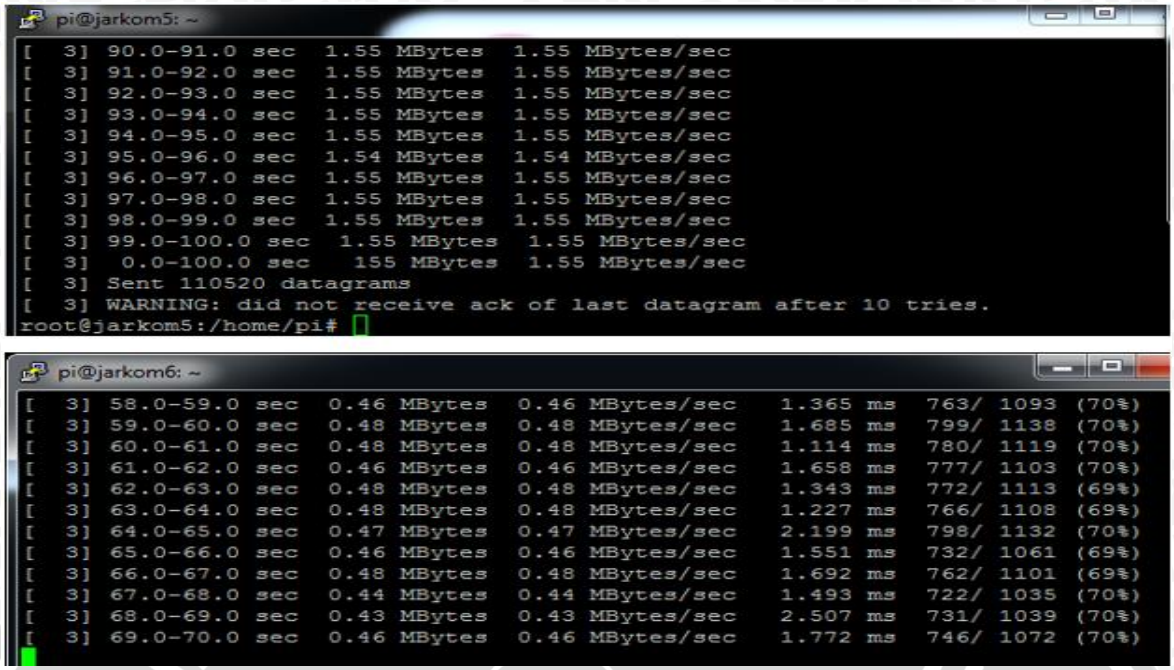
5.3.3 Prosedur pengujian

Prosedur yang diterapkan dalam pengujian perbandingan performansi protokol *Routing* OLSR, BATMAN dan Babel pada WMN meliputi:

- Memastikan ke-6 *node* WMN telah aktif dengan salah satu protokol *Routing*.
- Persiapkan PC/laptop untuk melakukan pengujian dengan me-*remote node* WMN.
- Pengujian dilakukan mulai pukul 21.00-08.00 WIB.
- Pengiriman paket UDP antar *node* dengan variasi ukuran besar paket adalah 13MB, 16MB, 19 MB, 22MB, 25MB dengan 5 sampel data untuk setiap besar paket data.
- Digunakan pada pengiriman paket UDP antara *node* 5 dan 6 dengan mematikan *node* 1 ketika proses pengiriman data berlangsung selama 20s.
- Digunakan pada pengiriman paket UDP antara *node* 5 dan 6 dengan mematikan *node* 4 ketika proses pengiriman data berlangsung selama 30s.

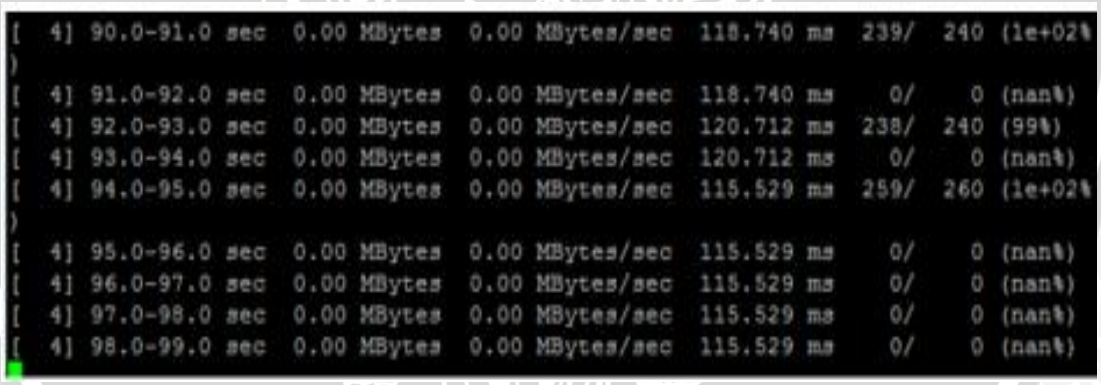
5.3.4 Hasil

Hasil pengujian performansi protokol *routing* OLSR, BATMAN dan Babel pada wmn dengan melakukan pengiriman paket UDP antara node 5 dan 6 dan mematikan 2 *node* adalah Babel memiliki performansi paling bagus. Hasil ini dikarenakan pada pengujian performasi yang dilakukan untuk protokol *routing* OLSR dan BATMAN mengalami kegagalan dalam pengiriman paket UDP. Pengiriman paket UDP pada BATMAN mengalami kegagalan untuk bagian *server*, sehingga paket data tidak dapat diterima oleh *client*. Lebih jelasnya dapat dilihat pada gambar 5.1.



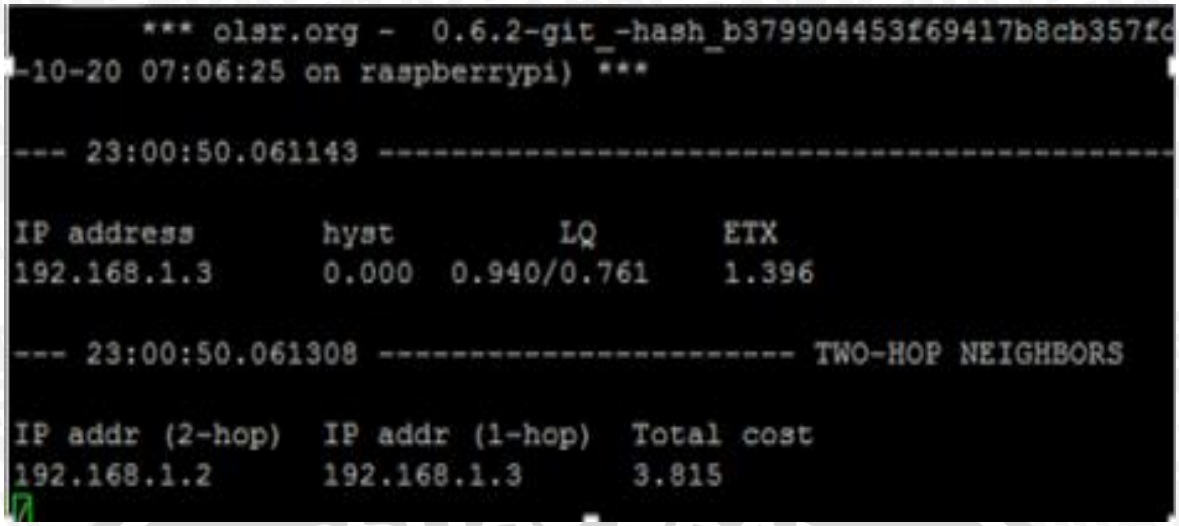
Gambar 5.1 Proses Pengiriman Paket yang Gagal pada BATMAN

Pengiriman paket UDP pada OLSR mengalami kegagalan untuk bagian *server*, sehingga paket data tidak dapat diterima oleh *client*. Lebih jelasnya dapat dilihat pada gambar 5.2.



Gambar 5.2 Proses Pengiriman Paket yang Gagal pada OLSR.

Ketika *node* OLSR yang bertindak sebagai *server* gagal melakukan pengiriman paket UDP, pada *interface* OLSR hanya mendeteksi 1 *hop* tetangga saja. 1 *hop* tetangga yang terdeteksi adalah *node* 2, sedangkan *node* 5 tidak terdeteksi. Keadaan ini yang menyebabkan kegagalan pengiriman paket UDP. Lebih jelasnya dapat dilihat pada gambar 5.3.



Gambar 5.3 *Interface* OLSR ketika *Node* 1 dan 4 Mati

Hasil uji performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan mesh dengan menggunakan skenario 1.1.3 dapat diambil kesimpulan bahwa protokol *Routing* Babel mendapatkan hasil terbaik. Protokol *Routing* OLSR dan BATMAN tidak dapat melakukan pengiriman paket data ketika diterapkan pada skenario ini. Untuk lebih jelasnya dapat dilihat pada tabel 5.5.

Tabel 5.5 Pengujian *Multihop* 5 ke 6 dengan *Node* 1,4 dimatikan.

	Jitter(ms)	*Throughput*(MBps)	*Packet Loss*(%)
OLSR	Gagal	Gagal	Gagal
BATMAN	Gagal	Gagal	Gagal
Babel	117.8135	0.0204	93.2

5.3.5 Analisis

Hasil perbandingan performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan mesh dengan menggunakan skenario mematikan *node* 1 dan 4 didapatkan protokol *Routing* Babel memiliki performansi paling bagus. Dengan nilai rata-rata *Throughput* 0.0204 MBps, *Jitter* 117.8135 dan *Packet Loss* 93.2%.

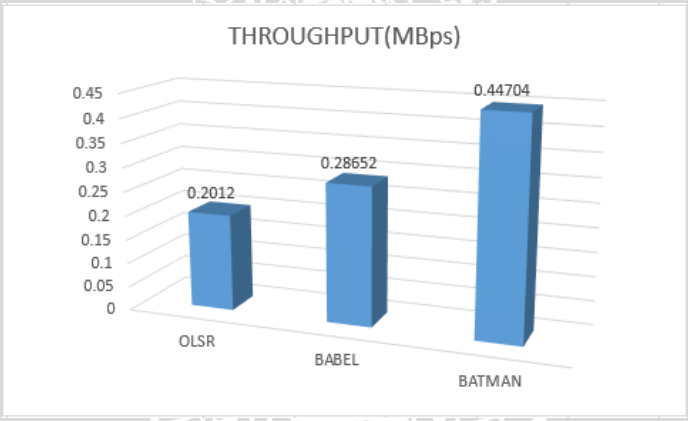
5.4 Analisis

Analisis data dilakukan untuk mengetahui performansi protokol *Routing* yang paling baik ketika diterapkan pada WMN. Hasil pengujian terdiri dari 3 bagian, masing-masing dilakukan dengan skenario berdeda-beda. Bagian pertama adalah hasil pengujian performansi protokol *Routing* antar *node* yang mencakup semua *node* pada WMN. Tujuan bagian pertama untuk mengetahui performansi masing-masing protokol *Routing* ketika diterapkan pada WMN. Bagian kedua adalah hasil pengujian performansi protokol *Routing* pada kondisi *Multihop* dengan mematikan 1 *node*. Skenario ini mematikan *node* yang berada diantara *node* yang sedang melakukan pengiriman paket UDP. Tujuan dari bagian kedua untuk

mengetahui performansi masing-masing protokol *Routing* ketika dijalankan pada kondisi ini. Tujuan dari bagian ketiga untuk mengetahui performansi protokol *Routing* terbaik ketika dijalankan pada kondisi *Multihop* dengan mematikan 2 *node* secara bergantian pada detik ke 20 dan 30 diantara kedua *node* yang sedang melakukan pertukaran data. Hasil analisa yang akan dilakukan dari ketiga bagian uji performansi protokol *Routing* OLSR, BATMAN, dan Babel pada WMN akan paparkan pada ulasan berikut.

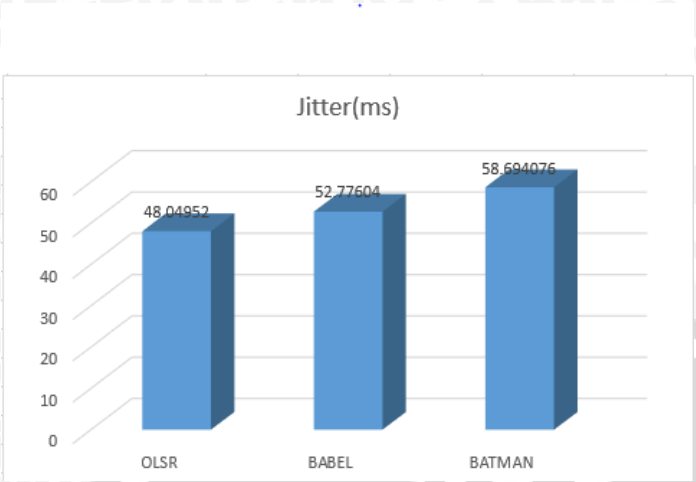
Skenario pengujian pertama dilakukan dengan mengirim paket data UDP antar *node*. Skenario pengujian ini melibatkan semua *node* WMN yang berjumlah 6. Tujuan skenario pengujian pertama untuk mengetahui protokol *Routing* dengan performansi terbaik ketika diterapkan pada WMN. Hasil pengujian akan digolongkan menjadi 2, berdasar kondisi dari interaksi antar *node*-nya yaitu *single-hop* dan *multi-hop*.

Rata-rata nilai *Throughput* dari hasil uji performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan mesh dengan interaksi *single-hop* adalah OLSR 0.2012 MBps, BATMAN 0.44704 MBps dan Babel 0.28652 MBps. Untuk lebih jelasnya dapat dilihat pada gambar 5.4.



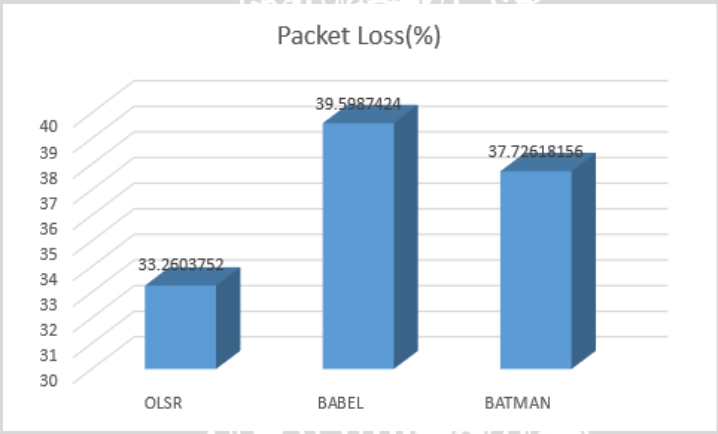
Gambar 5.4 Diagram Pengujian Performansi *Single-hop* (Throughput)

Rata-rata nilai *Jitter* dari hasil uji performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan mesh dengan interaksi *single-hop* adalah OLSR 48.04952 ms, BATMAN 58.694076 ms dan Babel 52.77604 ms. Untuk lebih jelasnya dapat dilihat pada gambar 5.5.



Gambar 5.5 Diagram Pengujian Performansi *Single-hop*(Jitter)

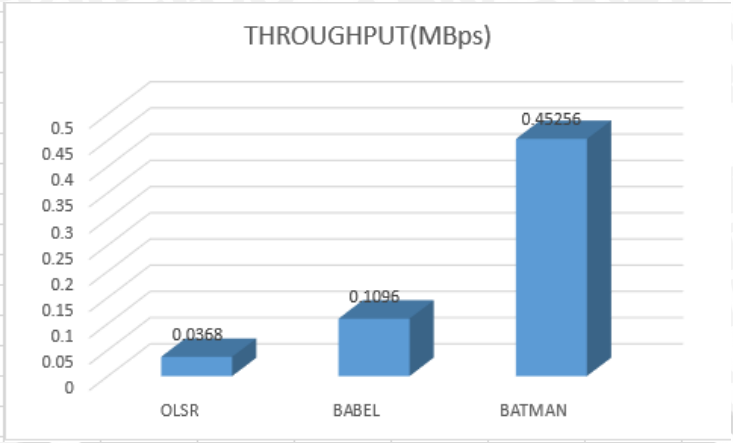
Rata-rata nilai *Packet Loss* dari hasil uji performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan mesh dengan interaksi *single-hop* adalah OLSR 33.2603752%, BATMAN 37.72618156% dan Babel 39.5987424%. Untuk lebih jelasnya dapat dilihat pada gambar 5.6.



Gambar 5.6 Diagram Pengujian Performansi *Single-hop*(Packet Loss)

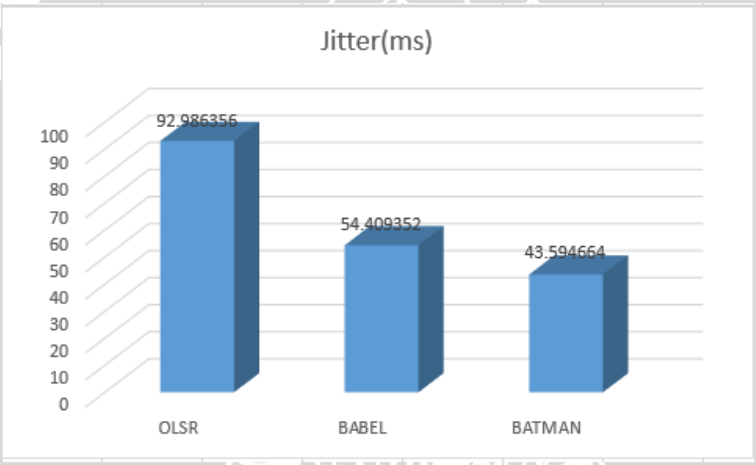
Analisis hasil dari perbandingan performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan mesh menggunakan skenario pertama dalam keadaan *single-hop* adalah OLSR memiliki performansi paling bagus. Nilai rata-rata uji performansi protokol *Routing* OLSR yang didapat adalah Throuhput 0.2012 MBps, Jitter 48.04952 ms, dan Packet Loss sebesar 33.26038%. Hal ini dipengaruhi oleh konsep kerja OLSR dalam penentuan *path*-nya dengan menggunakan MPRS.

Rata-rata nilai *Throughput* dari hasil uji performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan mesh dengan interaksi *multi-hop* adalah OLSR 0.0368 MBps, BATMAN 0.45256 MBps dan Babel 0.1096 MBps. Untuk lebih jelasnya dapat dilihat pada gambar 5.7.



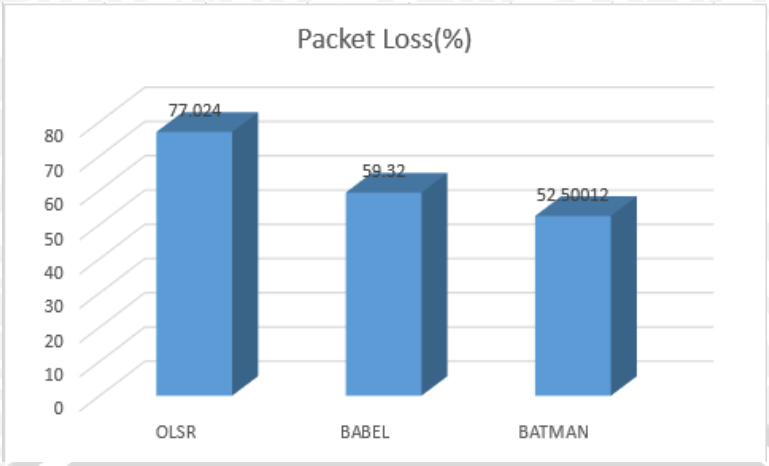
Gambar 5.7 Diagram Pengujian Performansi *Multi-hop*(Throughput)

Rata-rata nilai *Jitter* dari hasil uji performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan mesh dengan interaksi *multi-hop* adalah OLSR 92.986356 ms, BATMAN 43.594664 ms dan Babel 54.409352 ms. Untuk lebih jelasnya dapat dilihat pada gambar 5.8.



Gambar 5.8 Diagram Pengujian Performansi *Multi-hop*(Jitter)

Rata-rata nilai *Packet Loss* dari hasil uji performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan mesh dengan interaksi *multi-hop* adalah OLSR 77.024%, BATMAN 52.50012% dan Babel 59.32%. Untuk lebih jelasnya dapat dilihat pada gambar 5.9.



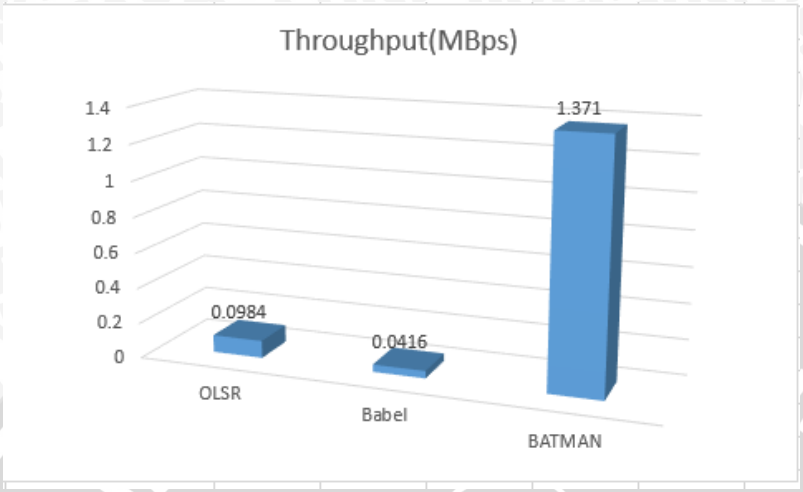
Gambar 5.9 Diagram Pengujian Performansi *Multi-hop*(Packet Loss)

Analisa yang dapat diambil dari skenario pertama mengenai perbandingan performansi protokol *Routing* OLSR, BATMAN dan Babel dalam kondisi *Multi-hop* adalah, BATMAN memiliki performansi paling bagus diantara ketiga protokol *Routing* lainnya. Nilai rata-rata dari pengujian performansi protokol *Routing* BATMAN berupa Throughput 0.45256 MBps, Jitter 43.59466 ms, dan Packet Loss sebesar 52.50012%. Hasil ini tak lepas dari konsep kerja protokol *Routing* BATMAN yang menerapkan pesan OGM untuk mengenali *node* tetangganya. Sehingga BATMAN dapat menekan resiko kegagalan yang akan dialami pada pengiriman paket data UDP.

Analisa secara menyeluruh perbandingan performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan mesh dari skenario pertama adalah BATMAN memiliki nilai rata-rata performansi yang lebih bagus dibandingan dengan OLSR dan Babel. Nilai rata-rata performansi protokol *Routing* BATMAN adalah *Throughput*: 0.4498 MBps, *Jitter*: 51.14437 ms dan *Packet Loss*: 46.04943%. Keunggulan BATMAN dibandingkan kedua protokol lainyya terletak pada metode pengenalan *node* tetangganya yang menerapkan pesan OGM. Dengan adanya konsep ini, *node* BATMAN hanya berinteraksi dengan *node* tetangganya saja. Jika sebuah *node* A akan melakukan pengiriman paket data menuju *node* B dan diharuskan melewati *node* lain, maka *node* tetangga BATMAN akan meneruskan paket menuju *node* B. Pada kasus ini, seolah *node* B menerima paket data dari *node* BATMAN lain yang dapat dijangkau oleh *node* B. Pengiriman paket data pada BATMAN, baik *node* yang bertindak sebagai pengirim maupun penerima paket data akan terlihat seolah-olah berjarak sangat dekat. Keadaan ini akan banyak berpengaruh pada performansi dari protokol BATMAN itu sendiri, terutama QOS yang dihasilkan.

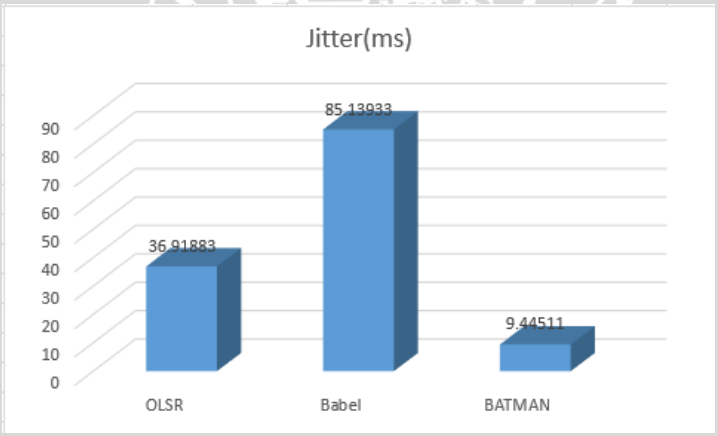
Pada pengujian kedua, dilakukan pengiriman paket data *multi-hop* dengan mematikan 1 *node* ketika proses pengiriman paket data sedang berjalan 30 detik. Skenario pengujian ini akan diterapkan pada sebagian *node* yaitu 2 ke 5 dengan *node* 3 dimatikan dan 2 ke 6 dengan *node* 1 dimatikan. Skenario dilakukan sebagai tambahan untuk mengetahui performansi masing-masing protokol *Routing* jika salah satu *node*-nya mati ketika sedang berlangsung proses pengiriman data.

Rata-rata nilai *Throughput* dari hasil uji performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan mesh *multi-hop* antara *node* 2 dan 5 dengan mematikan *node* 3 adalah OLSR 0.0984 MBps, BATMAN 1.371 MBps dan Babel 0.0416 MBps. Untuk lebih jelasnya dapat dilihat pada gambar 5.10.



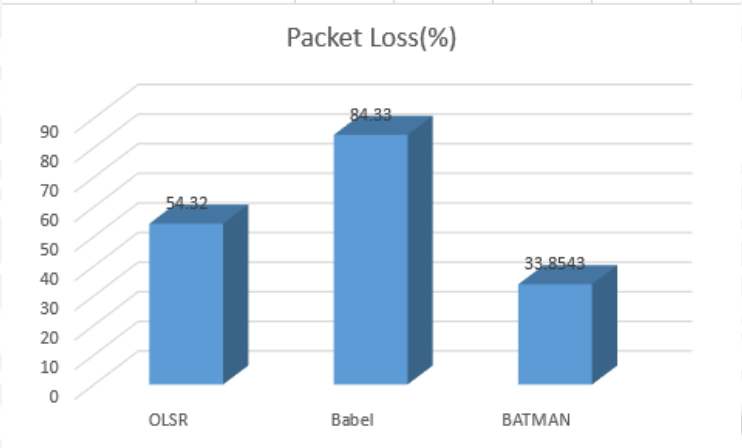
Gambar 5.10 Diagram Perbandingan Performansi *Multi-hop* 2 ke 5 dengan *node* 3 dimatikan(Throughput)

Rata-rata nilai *Jitter* dari hasil uji performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan mesh *multi-hop* antara *node* 2 dan 5 dengan mematikan *node* 3 adalah OLSR 36.91883 ms, BATMAN 9.44511 ms dan Babel 85.13933 ms. Untuk lebih jelasnya dapat dilihat pada gambar 5.11.



Gambar 5.11 Diagram Perbandingan Performansi *Multi-hop* 2 ke 5 dengan *node* 3 dimatikan(Jitter)

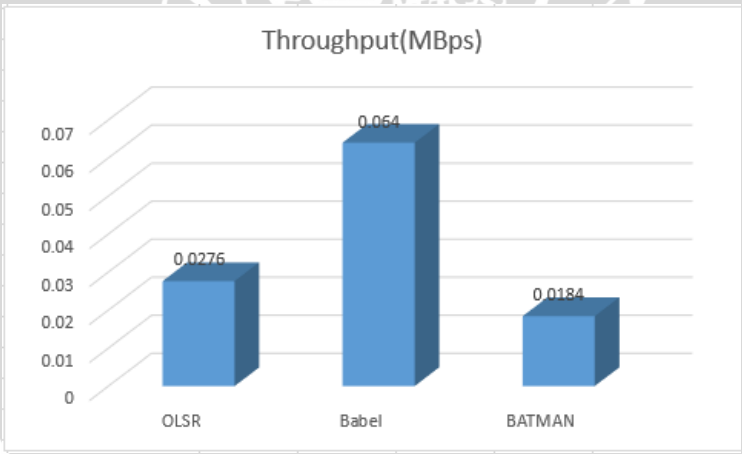
Rata-rata nilai *Packet Loss* dari hasil uji performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan mesh *multi-hop* antara *node* 2 dan 5 dengan mematikan *node* 3 adalah OLSR 54.32%, BATMAN 33.8543% dan Babel 84.33%. Untuk lebih jelasnya dapat dilihat pada gambar 5.12.



Gambar 5.12 Diagram Perbandingan Performansi *Multi-hop* 2 ke 5 dengan *node* 3 dimatikan(Packet Loss)

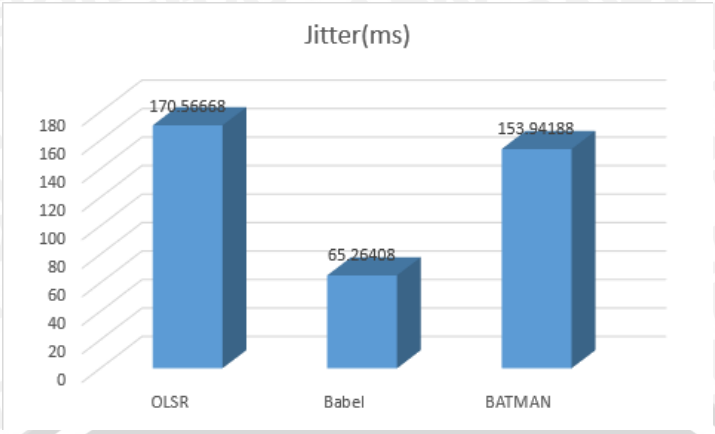
Analisa performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan mesh dari kondisi *multi-hop* dari node 5 dan 2 dengan mematikan *node* 3 adalah protokol *Routing* BATMAN memiliki nilai rata-rata performansi paling baik. Nilai rata-rata performansi yang didapat protokol *Routing* BATMAN adalah Throughput 9.44 MBps, Jitter 1.37 ms, dan Packet Loss 33.85%.

Rata-rata nilai *Throughput* dari hasil uji performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan mesh *multi-hop* antara *node* 2 dan 6 dengan mematikan *node* 1 adalah OLSR 0.0276 MBps, BATMAN 0.0184 MBps dan Babel 0.64 MBps. Untuk lebih jelasnya dapat dilihat pada gambar 5.13.



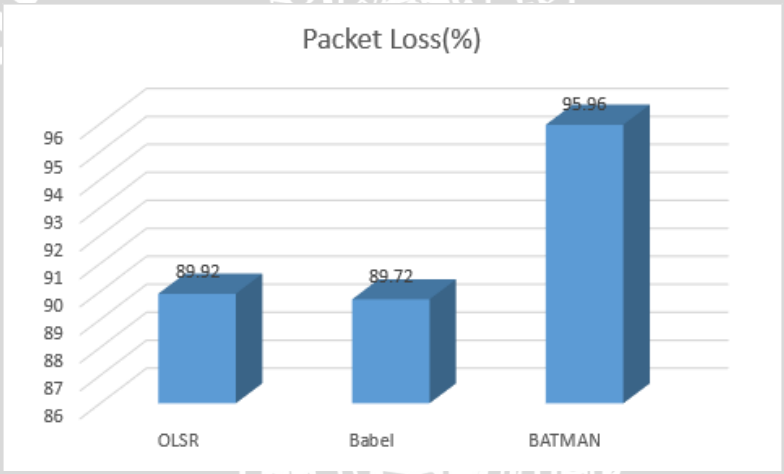
Gambar 5.13 Diagram Perbandingan Performansi *Multi-hop* 2 ke 6 dengan *node* 1 dimatikan(Throughput)

Rata-rata nilai *Jitter* dari hasil uji performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan mesh *multi-hop* antara *node* 2 dan 6 dengan mematikan *node* 1 adalah OLSR 170.56668 ms, BATMAN 153.94188 ms dan Babel 65.26408 ms. Untuk lebih jelasnya dapat dilihat pada gambar 5.14.



Gambar 5.14 Diagram Perbandingan Performansi *Multi-hop* 2 ke 6 dengan *node* 1 dimatikan(Jitter)

Rata-rata nilai *Packet Loss* dari hasil uji performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan mesh *multi-hop* antara *node* 2 dan 6 dengan mematikan *node* 1 adalah OLSR 89.92%, BATMAN 95.96% dan Babel 89.72%. Untuk lebih jelasnya dapat dilihat pada gambar 5.15.



Gambar 5.15 Diagram Perbandingan Performansi *Multi-hop* 2 ke 6 dengan *node* 1 dimatikan(Packet Loss)

Analisa performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan mesh dari kondisi *multi-hop* dari *node* 6 dan 2 dengan mematikan *node* 1 adalah protokol *Routing* Babel memiliki nilai rata-rata performansi paling baik. Nilai rata-rata performansi yang didapat protokol *Routing* Babel adalah Throughput 0.064 MBps, Jitter 65.24 ms, dan Packet Loss 89.72%.

Analisa yang dapat diambil dari perbandingan performansi protokol *Routing* OLSR, BATMAN dan Babel pada jaringan mesh menggunakan skenario kedua mendapatkan hasil bahwasannya protokol *Routing* BATMAN memiliki performansi paling bagus dibandingkan dengan OLSR dan Babel. Nilai rata-rata yang didapatkan protokol *Routing* BATMAN adalah *Throughput*: 0.6947 MBps , *Jitter*: 81.693495 ms dan *Packet Loss*: 64.90715%. BATMAN memiliki nilai rata-rata *Throughput* dan *Jitter* yang cenderung tinggi, akan tetapi nilai rata-rata *Packet Loss*

cenderung rendah. Dalam melakukan pengiriman paket pada skenario ini, BATMAN lebih bagus di segi ketersediaan paket data yang ke-*node* penerima dengan kecilnya *Packet Loss*.

Pada pengujian ketiga, dilakukan pengiriman paket data UDP dari 5 ke 6 dengan mematikan *node* 1,4 pada detik ke 20 dan 30. Skenario ini dimaksudkan untuk mengetahui performasi masing-masing protokol *Routing* ketika ada 2 *node* yang dimatikan secara bergantian pada saat ada proses pengiriman data sedang berlangsung. Hasil dari perbandingan performansi dari ketiga protokol *Routing* tersebut dapat dilihat pada tabel 5. Dari data yang tercantumkan pada table 5.6 dapat ditarik kesimpulan bahwasannya hanya protokol *Routing* Babel yang dapat bekerja pada skenario pengujian 3. Hasil yang dapat disajikan pada skenario ini berupa tabel data hasil akumulasi pengujian perbandingan performansi. Data yang diperoleh dapat dilihat pada tabel 5.6.

Tabel 5.6 Pengujian *Multihop* 5 ke 6 dengan *Node* 1,4 dimatikan.

	Jitter(ms)	Throughput(MBps)	Packet Loss(%)
OLSR	Gagal	Gagal	Gagal
BATMAN	Gagal	Gagal	Gagal
Babel	117.8135	0.0204	93.2

Dari data yang didapat, dapat diambil kesimpulan bahwasannya hanya protokol *Routing* Babel yang dapat berjalan pada kondisi skenario 3.

Kesimpulan yang diperoleh dari seluruh skenario yang telah dilakukan, terdapat faktor yang dapat mempengaruhi performansi dari masing-masing protokol *Routing*. Faktor yang paling berpengaruh adalah jangkauan jarak setiap *node* dari masing-masing protokol *Routing* yang sangat jauh. Dampaknya adalah paket data yang diterima oleh *node client* akan mengalami *Packet Loss* tinggi dan banyak mengalami kegagalan dalam proses pengiriman paket. Protokol *Routing* dengan performansi paling stabil diterapkan pada ketiga skenario pengujian adalah Babel. Nilai rata-rata yang didapatkan oleh protokol *Routing* Babel adalah *Throughput* 0.09042 MBps, *Jitter* 82.20263333 ms dan *Packet Loss* 76.24936333%. Babel memiliki dua karakteristik khas jika bekerja pada jaringan relay. Dua karakteristik khas Babel antara lain:

1. Babel menggunakan *history-sensitive* pemilihan rute untuk meminimalkan dampak *route flaps.*
2. Babel mengesekusi *update* reaktif dan memaksa untuk meminta informasi routing ketika mendetaksi kegagalan routing dari salah satu *node* tetangga.

Karakteristik yang dimiliki Babel terbukti dapat mengatasi permasalahan yang ada pada skenario 3. Sedangkan protokol *Routing* BATMAN dan OLSR terbukti tidak dapat menyelesaikan permasalahn yang ada pada skenario 3. Ketika ketiga protokol *Routing* dapat berjalan dalam skenario yang sama, BATMAN memiliki performansi lebih bagus dibandingkan protokol *Routing* lainnya. Keunggulan

BATMAN terletak pada interaksi antar *node* dengan menggunakan *broadcast* pesan OGM ke tetangga *node*-nya. Interaksi pada *node* BATMAN hanya bisa dilakukan dengan *node* tetangga terdekat. Metode ini terbukti dapat menekan nilai *Packet Loss* pada pengiriman data.

BAB 6 PENUTUP

6.1 Kesimpulan

Hasil analisis dan pengujian perbandingan performansi protokol *Routing* OLSR, BATMAN dan Babel pada WMN dapat disimpulkan bahwa :

1. Hasil pengujian performansi protokol *Routing* yang telah dilakukan pada 3 skenario meliputi:
    - OLSR pada skenario pertama didapat hasil *Throughput*: 0.119 MBps, *Jitter*: 70.51794 ms dan *Packet Loss*: 55.14219%. Skenario kedua didapat hasil *Throughput*: 0.063 MBps, *Jitter*: 103.742755 ms dan *Packet Loss*: 72.12%. Skenario ketiga didapat hasil *Throughput*: gagal, *Jitter*: gagal dan *Packet Loss*: gagal.
    - BATMAN pada skenario pertama didapat hasil *Throughput*: 0.4498 MBps, *Jitter*: 51.14437 ms dan *Packet Loss*: 46.04943%. Skenario kedua didapat hasil *Throughput*: 0.6947 MBps , *Jitter*: 81.693495 ms dan *Packet Loss*: 64.90715%. Skenario ketiga didapat hasil *Throughput*: gagal, *Jitter*: gagal dan *Packet Loss*: gagal.
    - Babel pada skenario pertama didapat hasil *Throughput*: 0.19806 MBps, *Jitter*: 53.592695 ms dan *Packet Loss*: 48.52309%. Skenario kedua didapat hasil *Throughput*: 0.0528 MBps, *Jitter*: 75.201705 ms dan *Packet Loss*: 87.025%. Skenario ketiga didapat hasil *Throughput*: 0.0204 MBps , *Jitter*: 117.8135 ms dan *Packet Loss*: 93.2 %.
2. Dari perbandingan performansi yang dilakukan, Babel merupakan protokol *Routing* terbaik dengan nilai rata-rata *Throughput* 0.09042 MBps, *Jitter* 82.20263333 ms dan *Packet Loss* 76.24936333%.

6.2 Saran

Saran yang dapat disampaikan untuk pengembangan lebih lanjut adalah sebagai berikut:

1. Perlu dilakukan penelitian lebih lanjut mematikan lebih banyak *node* ketika pengiriman paket data sedang berlangsung antara ujung *node* pada topologi jaringan mesh dengan menambah jumlah *node*.
2. Perlu dilakukan pengujian untuk mengetahui besar pengaruh interferensi pada jaringan mesh.

DAFTAR PUSTAKA

P.P, Takhare., M.A Joshi & A.D Routh., 2012. *A review paper on routing protocols of wirwlwss ad-hoc network technology*. Inter-national journal of networking. ISSN: 2449-278X & E-ISSN: 2449-2798. Volume 2. Issue-1. pp-35-39.(diakses pada 8 agustus 2015)

Friginal, jesus., Ruiz, Juan-Carlos., Andres, David de & Bustos Antonio, 2012. Mitigating the Impact of Ambient Noise on Wireless Mesh Network Using Adaptive Link-Quality-Based Packet Replication.Spanyol.(diakses pada 8 agustus 2015)

Ratelinggi, Parma Hadi & Djanali, Supeno, 2015. *Kinerja Protokol Routing pada Lingkungan Wireless Mesh Network dengan Combined Scalable Video Coding*. ISSN/e-ISSN: 1412-6389 / 2406-8535. JUTI - Volume 13, Nomer 1. Surabaya. (diakses pada 9 Agustus 2015)

DeCristofaro, Michael A., Lansdowne, Chatwin A. & Schlesinger, Adam M., 2015. *Heterogeneous Wireless Mesh Network Technology Evaluation for Space Proximity and Surface Applications*.Houston.(diakses pada 8 agustus 2015)

Chroboczek, J., 2014. *Diversity Routing for the Babel Routing Protocol*.Paris. (diakses pada 8 agustus 2015)

Neuman, Axel., Aichele, Corinna., Lindner, Marek & Wunderlich, S., 2008. *Better Aproach To Mobile Ad-Hoc Networking (BATMAN.)*. Internet-Draft. (diakses pada 8 agustus 2015)

Margolang, Adam Kurniawan, 2014. Analisis Perbandingan Protokol Better Approach To Mobile Ad-Hoc Network(BATMAN) dengan Protokol Babel untuk Layanan Voice Over Internet Protocol(VOIP) pada Mobile Ad-Hoc Network(MANET). Medan. (diakses pada 15 agustus 2015)

Abolhasan, Mehrana., Hagelstein, Brett & Wang, Jerry Chun-Ping, 2009. *Real-world Performance of Current Proactive Multi-hop Mesh Protocols*. Wolongong: NSW 2522. Shanghai - China. (diakses pada 8 agustus 2015)

Jacquet, P., Muhlethaler, P., Clausen, T., Laouiti, A., Qayyum, A. & Viennot, L., 2003. *Optimized Link State Routing Protocol from Ad Hoc Network*.Prancis. (diakses pada 8 agustus 2015)

Lindner, marek., Wunderlich, simon., Lussing, Linus & Hudeboll, Martin, 2006. *BATMAN Concept.*[online] https://www.open-mesh.org/projects/open-mesh/wiki/BATMANConcept (diakses pada 8 september 2015)

Kassler, Andreas J., 2012. *Introduction to Wireless Mesh Networks*. ICTP-ITU/BDT School on Sustainable Wireless ICT Solutions 2012: Italy(diakses pada 8 september 2015)

Mariana, 2012. Model *Raspberry Pi* dari yang lama hingga terbaru.[online] https://tutorkeren.com/artikel/lengkap-5-model-raspberry-pi-beserta-perbandingan-dan-spesifikasinya.htm (diakses pada 9 september 2015)

Syahrial, Miftah Rahman, 2014. Analisis Quality of service IP Telephony dengan Metode Low Latency Queuing. [online] (diakses pada 18 mei 2016)

Romadhon, Pearl Pratama, 2014. Analisis Kinerja *Wireless LAN* menggunakan Metode *QOS* dan *RMA* pada PT PERTAMINA EP UBEP RAMBA(Persero).Palembang.(diakses pada 18 mei 2016)

Tiphon, 1999. Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON) General aspects of Quality of Service (QoS). DTR/TIPHON-05006 (cb0010cs.PDF).(diakses pada 7 desember 2017)