

Sistem Komunikasi Darurat Bencana Dengan Teknologi Mobile *Ad-Hoc* Network (MANET)

Audy Gerald Palilingan, Meicsy E.I Najoan, Sherwin R.U.A Sompie

Teknik Elektro Universitas Sam Ratulangi Manado, Jl. Kampus Bahu-Unsrat Manado, 95115

E-mail: audygeraldpalilingan@gmail.com, meicsynajoan@unsrat.ac.id, aldo@unsrat.ac.id

Diterima : 4 Juni; direvisi : 15 Juni; disetujui : 18 Juni

Abstract - Natural disasters are natural phenomena that occur directly or indirectly disrupt human life. Indonesia has a high level of potential natural disasters that can cause damage to telecommunications infrastructure. On the one hand, communication during disasters is very important so that disaster management can run well

The solution to deal with the problem above is to use Mobile Ad-Hoc Network (Manet) technology to build a communication system. This communication system is expected to be used when the main infrastructure is damaged so that it can be used as an alternative to communication.

The research will be obtained the results that the technology of manet is suitable for communication systems in disaster areas, because the connectivity test has been carried out that can perform self-configure and self-healing on the wifi system that has been installed at each manet node, even though each node in manet is mobile causing frequent changes in topology.

Keywords – Disaster; Emergency Communication; Mobile Ad-hoc Network; Self Configure; Self Healing.

Abstrak - Bencana alam merupakan suatu fenomena alam yang terjadi secara langsung maupun tidak langsung mengganggu kehidupan manusia. Indonesia memiliki tingkat potensi bencana alam yang tinggi yang dapat menyebabkan kerusakan infrastruktur telekomunikasi. Di satu sisi, komunikasi pada saat terjadi bencana sangat penting agar penanggulangan bencana dapat berjalan dengan baik

Solusi untuk menangani masalah diatas adalah dengan menggunakan teknologi Mobile Ad-Hoc Network (Manet) untuk membangun sistem komunikasi. Sistem komunikasi ini diharapkan dapat digunakan pada saat infrastruktur utama mengalami kerusakan sehingga bisa digunakan sebagai alternatif komunikasi.

Penelitian yang dilakukan akan diperoleh hasil bahwa teknologi manet cocok digunakan untuk sistem komunikasi pada daerah bencana, karena telah dilakukan uji konektifitasnya yang dapat melakukan *self configure* dan *self healing* pada sistem *wifi* yang telah terpasang di tiap *node* manet, meskipun setiap *node* dalam manet bersifat *mobile* yang menyebabkan sering terjadinya perubahan topology.

Kata Kunci : Bencana, Komunikasi Darurat, Mobile Ad-hoc Network, Self Configure, Self Healing.

I. PENDAHULUAN

Bencana alam merupakan suatu fenomena alam yang terjadi secara langsung maupun tidak langsung mengganggu kehidupan manusia. Dalam hal ini, bencana alam dapat menyebabkan kerugian bagi manusia baik secara materi, non materi bahkan jiwa.

Indonesia memiliki tingkat potensi bencana alam yang tinggi yang dapat menyebabkan kerusakan infrastruktur

telekomunikasi. Di satu sisi, komunikasi pada saat terjadi bencana sangat penting agar penanggulangan bencana dapat berjalan dengan baik.

Solusi untuk menangani masalah diatas adalah dengan menggunakan teknologi Mobile Ad-Hoc Network (Manet) untuk membangun sistem komunikasi. Sistem komunikasi diharapkan dapat digunakan pada saat infrastruktur utama mengalami kerusakan sehingga bisa digunakan sebagai alternatif komunikasi. Salah satu sistem komunikasi adalah bisa berupa pengiriman pesan dengan menggunakan aplikasi VoIP yang sudah ada atau membuat sendiri program untuk melakukan komunikasi pada jaringan terdapat di antara *node-node* yang sudah terhubung.

Dari penelitian yang dilakukan diperoleh hasil bahwa teknologi manet cocok digunakan untuk sistem komunikasi pada daerah bencana, karena telah dilakukan uji konektifitasnya yang dapat melakukan *self configure* dan *self healing* pada sistem *wifi* yang telah terpasang di tiap *node* manet, meskipun setiap *node* dalam manet bersifat *mobile* yang menyebabkan sering terjadinya perubahan topology.

A. Mobile Ad-hoc Network (MANET)

Salah satu contoh model jaringan nirkabel yang memiliki kemampuan *multi-hop* dan mampu beroperasi tanpa dukungan infrastruktur apapun adalah jaringan Ad Hoc[1]. Ketidakhadiran infrastruktur atau pusat koordinator komunikasi atau *base station* menjadikan *routing* sangat kompleks dibandingkan jaringan selular (infrastruktur network). Dalam *Ad-Hoc*, setiap *node* bertugas dalam merouting data kepada *node* lain sehingga penentuan *node* mana yang mengirim data dibuat secara dinamis berdasarkan konektivitas dari jaringan itu sendiri. Mobile Ad-Hoc Network (MANET) merupakan sebuah teknologi *wireless* LAN terdiri gabungan dari *node-node* atau perangkat-perangkat bergerak (*mobile*) yang sifatnya dinamis. Mobile Ad-Hoc Network (MANET) bekerja tanpa menggunakan infrastruktur dalam jaringan yang sudah ada seperti *access point* dan lain-lain sehingga membentuk jaringan yang bersifat sementara dan mempermudah *user* dalam *mobile device* nya. *Node* pada jaringan MANET tidak hanya berperan sebagai pengirim atau penerima data saja tapi dapat berperan sebagai menunjang *node* yang lain yang dapat meneruskan paket data kepada perangkat lain. MANET sangat cocok diterapkan di daerah yang kekurangan infrastruktur telekomunikasi seperti solusi telekomunikasi pada saat terjadinya bencana alam yang mengalami kerusakan prasarana jaringan komunikasi fisik, ataupun pembangunan jaringan komunikasi di medan perang[1].

Routing adalah suatu proses memindahkan informasi dari pengirim ke penerima melalui sebuah jaringan[2]. Sedangkan protokol merupakan aturan yang mengatur setiap komputer untuk saling bertukar informasi melalui media jaringan. Sehingga *routing* protokol diperlukan untuk mengatur bagaimana router berkomunikasi antara satu dengan yang lain dalam menyebarkan informasi, yang memungkinkan router untuk memilih rute pada jaringan computer[3]. Pada umumnya *routing* protokol untuk jaringan *ad-hoc* dibagi menjadi dua tipe yaitu *Table Driven Routing Protocol* (Proaktif) dan *On-Demand Routing Protocol* (Reaktif). *Table Driven Routing Protocol* (Proaktif). Protokol *routing* proaktif bersifat *table driven* artinya dimana setiap *node* menyimpan tabel yang berisi informasi rute ke setiap *node* yang diketahuinya, artinya sebuah *node* mengetahui semua route ke *node* lain yang berada dalam jaringan tersebut. Informasi rute diperbaharui secara berkala jika terjadi perubahan link.

B. Optimize Link State Routing (OLSR)

OLSR (*Optimized Link State Protocol*) merupakan salah satu jenis dari *proactive routing* protokol yang biasa digunakan dalam jaringan *ad-hoc*. OLSR merupakan salah satu dari dua standart internet untuk jaringan mesh[4]. OLSR sering dijalankan pada *wireless mesh network*. Protokol ini melakukan pertukaran pesan secara periodik dalam rangka menjaga informasi topologi jaringan yang ada pada setiap *node*. Secara umum langkah-langka kerja dalam OLSR dapat diurutkan sebagai berikut :

1. *Link Sensing* (Pendeteksian hubungan).
2. *Neighbour detection* (pendeteksian *node* tetangga).
3. *MPR selection* (pemilihan MPR).
4. Pengiriman TC (Topology Control) *Messages*.
5. *Route calculation* (penghitungan jalur).

C. Raspberry Pi (Raspi)

Raspberry Pi adalah sebuah mini-komputer atau dikenal sebagai *Single-Board Circuit* (SBC) dimana ukuran dari perangkat ini hanya sebesar kartu kredit. Raspberry Pi pada umumnya mempunyai fungsi yang sama dengan komputer atau laptop, bisa menjalankan perangkat lunak pengolah kata, pengolah angka, permainan, pemutar audio dan pemutar video. Raspberry Pi resmi dirilis pada bulan Februari 2012 oleh yayasan *Raspberry Pi Foundation*. Tujuan awal diciptakan Raspberry Pi adalah untuk memperkenalkan keterampilan komputer tingkat rendah untuk anak-anak di Inggris. Perkembangan teknologi saat ini membuat Raspberry Pi bisa difungsikan sebagai *Server* dan juga bisa menjadi perangkat pendukung dalam *Internet of Things* (IoT).

D. Iperf

Iperf adalah program yang berfungsi untuk mengukur performa dari sebuah jaringan komputer. Iperf adalah program yang gratis (*open-source*) dan dapat dijalankan pada platform yang berbeda-beda termasuk Linux dan Windows. Iperf dapat mengukur aliran data dari TCP (*Transmission Control Protocol*) dan UDP (*User Datagram Protocol*). Pada protokol TCP, iPerf dapat mengukur bandwidth dan downlink dan pada protokol UDP, iPerf dapat mengukur *packet loss*, *delay* dan *jitter*. Iperf bekerja dengan mode *client – server*.

E. Virtual Networking Computer (VNC)

Virtual Networking Computer (VNC) adalah singkatan dari komputasi jaringan virtual. Ini adalah sistem berbagi desktop yang memungkinkan Anda mengontrol komputer lain dari jarak jauh. VNC bekerja dengan mentransmisikan semua pergerakan keyboard dan mouse Anda dari komputer *client* Anda ke komputer *client* besar lainnya.

Sistem komputasi jaringan virtual bergantung pada platform. Ini berarti bahwa *client* yang bekerja pada satu jenis sistem operasi tidak dapat terhubung ke *server* VNC yang beroperasi pada jenis sistem operasi yang berbeda. Ada banyak jenis *client* dan *server* yang tersedia untuk sistem berbasis GUI yang berbeda. Selain itu, sistem komputasi jaringan virtual tersedia untuk Java. Beberapa program VNC hanya berfungsi untuk sistem operasi Windows.

F. Secure Shell (SSH)

Secure Shell (SSH) adalah salah satu utility dalam sistem operasi Linux yang digunakan untuk melakukan remote login pada komputer yang berada pada sebuah jaringan. SSH juga dapat dijalankan pada sistem operasi Windows tetapi dengan bantuan software tertentu seperti Putty. Perbedaan penggunaan utility *remote login* SSH dan TELNET yaitu SSH mengenkripsi data yang di transmisikan dalam jaringan komputer menjadikan SSH lebih aman untuk digunakan.

SSH menggunakan model *Client – Server*, dimana dibutuhkan alamat *server* dan port untuk bisa melakukan SSH, port default untuk SSH dalam TCP yaitu 22. Salah satu fungsi dari SSH adalah kita bisa menggunakan untuk melakukan transfer file secara aman, dengan menggunakan perintah *secure copy* (SCP) atau *Secure File Transfer Protocol* (SFTP). Berikut adalah contoh melakukan SSH.

G. File Transfer Protocol (FTP)

Proftpd merupakan sebuah program dari protokol internet untuk melakukan *File Transfer* seperti upload dan download file yang dilakukan oleh FTP *client* dan FTP *server*. Layanan FTP bisa diatur menjadi FTP public, dimana semua orang bisa mengakses data-data yang ada di *server* FTP dengan mudah. Selain dapat diatur menjadi FTP public, layanan FTP ini juga bisa diatur agar tidak semua orang dapat mengakses data-data yang ada di *server*, jadi hanya pengguna terdaftar saja yang memiliki izin untuk mengakses data-data tersebut. FTP berkerja menggunakan salah satu protokol yang dapat diandalkan untuk urusan komunikasi data antara *client* dan *server*, yaitu protokol TCP (yang menggunakan port nomor 21). Port 21 ini digunakan untuk mengirimkan command (perintah). Oleh karena port 21 dimaksudkan khusus untuk mengirimkan command, maka port ini sering juga disebut dengan nama command port. Dengan adanya protokol ini, antara *client* dan *server* dapat melakukan sesi komunikasi sebelum pengiriman data berlangsung..

H. Topologi Wifi

Terdapat dua buah topologi yang biasa digunakan pada jaringan *Wireless LAN* yaitu Mode Infrastruktur dan mode *Ad-Hoc*.

1). Mode Infrastruktur

Mode ini membutuhkan *access point* sebagai sumber lalu lintas data jaringan guna menghubungkan dan mengatur data

semua perangkat pada jaringan tersebut. *Access point* mengirimkan data ke semua *device* hanya jika semua *device* berada didalam jangkauan *access point* tersebut. Jarak jangkauan Wi-Fi dapat diperluas dengan menambahkan dan mengatur ulang letak *access point*[5].

2). Mode Ad-Hoc

Mode *Ad-Hoc*, untuk mode ini untuk saling terhubung pada suatu jaringan tidak memerlukan *Access Point* untuk saling terhubung dan melakukan komunikasi sesama perangkat, yang diperlukan hanyalah *wireless receiver* dan *transmitter* pada setiap perangkat.

I. Penelitian Terkait

Penelitian yang dilakukan oleh Salem Sati, 2018[6] dengan Manet Testbed Using Raspberry Pis. Dalam Penelitian ini penulis menganalisa performa, hasil, dan stabilitas dari 2 buah protokol *routing vector* jarak yaitu OLSR dan Babel.

Penelitian yang dilakukan oleh Mochammad Luthfi febriadi, Adian Fatchur Rochim, Eko Didik Widiyanto, 2013[7] dengan judul penelitian Perencanaan Dan Implementasi *Wireless Mesh Node* pada Raspberry Pi. Dalam penelitian ini dilakukan pengimplementasian *wireless mesh node* dengan protokol *routing* OLSR. Dimana dalam penelitian ini dilakukan pengujian dan analisis terhadap *wireless mesh node* tersebut dengan beberapa parameter seperti pengujian penggunaan CPU dan memori, pengujian kinerja *self-configure*, *Throughput analysis*, dan Analisa *Latency*.

Penelitian yang dilakukan oleh Ulfa Septilia Permatasari dan Indrastanti Ratna Widiasari, 2016[8] dengan penelitian Analisis *Routing* Protokol Optimized Link State *Routing* (OLSR) Pada Raspberry Pi. Dalam penelitian ini, penulis membahas tentang analisa mengenai topologi jaringan mesh pada raspberry pi menggunakan OLSR. Dimana parameter yang diuji adalah pengujian *self-configure*, *self-healing* dan *bandwidth*.

Penelitian oleh Rudi Kurniawan, 2019[9] dengan penelitian Penerapan Protokol *Routing* Proaktif OLSR Pada Jaringan Mobile *Ad-hoc* Network (MANET) Menggunakan Raspberry Pi Untuk Transfer File. Dalam penelitian ini, peneliti melakukan pengujian pada kemampuan MANET dengan menerapkan *File Transfer Protocol* (FTP) sebagai media untuk melakukan komunikasi data *file transfer* yang diimplementasi pada jaringan MANET. Dari hasil pengujian yang telah dilakukan diperoleh hasil bahwa File Transfer dapat berfungsi dengan baik saat diterapkan pada jaringan MANET.

II. METODE PENELITIAN

A. Jenis Metode Penelitian

Jenis metode penelitian yang digunakan untuk penelitian ini adalah metode penelitian eksperimen. Penelitian eksperimen adalah penelitian yang menyangkut pengaruh dari manipulasi satu atau lebih variabel terhadap variabel lain yang dilakukan secara sistematis[10]. Variabel yang dimanipulasi disebut *experimental treatment* atau *independent variable*. Variabel yang diamati dan terukur disebut *dependent variable*. Dalam penelitian eksperimen, peneliti harus berusaha untuk

mengontrol semua variabel – variabel lain yang bisa mempengaruhi *dependent variable*.

Dari pengertian diatas, dapat dijelaskan bahwa metode penelitian eksperimen adalah metode yang digunakan untuk melihat perubahan dari variabel yang akan diukur (*dependent variabel*) dan dapat dimanipulasi dengan merubah variabel – variabel yang lain (*independent variabel*).

B. Tempat dan Waktu Penelitian

Tempat pengolahan data program dan penelitian tugas akhir ini dilakukan di Fakultas Teknik Universitas Sam Ratulangi Manado tepatnya Laboratorium Sistem Komputer UNSRAT dengan lama waktu penelitian dimulai pada bulan september tahun 2018 sampai bulan Januari 2020.

C. Perangkat Keras dan Perangkat Lunak

Dalam penelitian ini digunakan beberapa perangkat keras dan perangkat lunak. *Raspberry Pi 3* pada penelitian ini bertindak sebagai *server* dan *station*. *Wifi Dongle* sebagai penghubung tiap *node* yang ada pada jaringan *MANET* secara *wireless*. *Power Bank* diunakan sebagai penyuplai bagi *Raspberry Pi* agar dapat menyala tanpa menggunakan suplai *power* dari adapter. Menggunakan *Raspbian Jessie* sebagai sistem operasi dari *Raspberry Pi*

D. Prosedur Penelitian

Dalam melakukan penelitian ini berikut adalah uraian tahapan penelitian yang akan dilakukan. Tahap – tahap yang akan dilakukan adalah:

1) Studi Literatur

Studi Literatur merupakan tahap pertama penelitian yaitu, pengumpulan data melalui jurnal, paper riset dan buku. Mencari referensi-referensi yang berhubungan dengan analisa performa komunikasi jaringan, *routing* protocol pada jaringan mesh dan mengkonfigurasi protokol *routing* tersebut pada raspberry pi.

2) Analisis Kebutuhan

Menganalisis apa yang dibutuhkan dalam penelitian, seperti menambahkan wifi dongle TP-LINK WN725N sebagai tambahan perangkat yang digunakan sebagai penangkap signal bagi raspberry yang bertugas sebagai *server* dan juga menentukan topologi jaringan yang akan digunakan untuk percobaan.

3) Perancangan dan Konfigurasi Perangkat Keras

Melakukan perancangan dan konfigurasi terhadap perangkat keras keseluruhan sistem yang akan digunakan.

4) Perancangan dan Konfigurasi Perangkat Lunak

Selanjutnya melakukan perancangan dan konfigurasi terhadap keseluruhan perangkat lunak yang akan digunakan yang didalamnya terdapat juga perancangan program *routing* OLSR untuk sistem jaringan mesh yang akan dibuat.

5) Pengujian performa self-configure pada raspberry pi

Selanjutnya, melakukan pengukuran performa komunikasi antara sistem komunikasi yang ada di kapal dan sistem komunikasi yang ada di pantai.

6) Uji coba komunikasi antar perangkat

Setelah sistem jaringan mesh telah berhasil dibuat. Selanjutnya, melakukan simulasi komunikasi antar perangkat menggunakan jaringan yang telah dirancang.

7) Analisa Data Performa Komunikasi

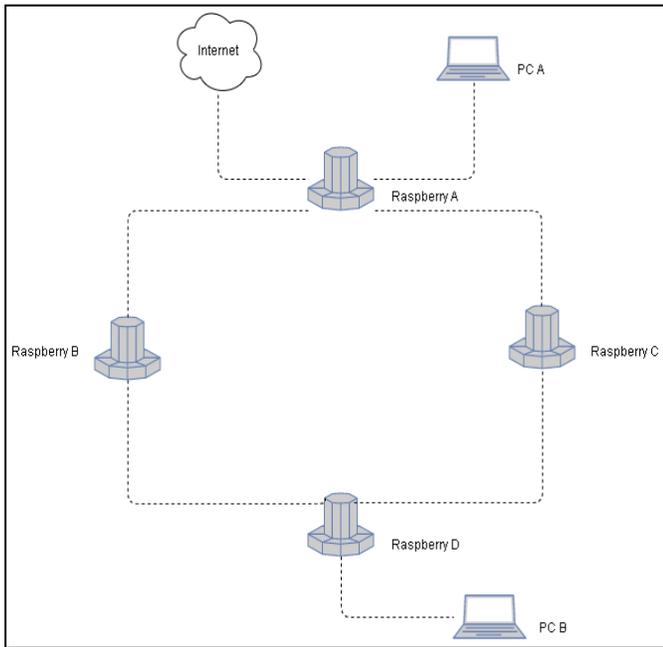
Analisa data performa komunikasi merupakan tahap akhir penelitian dimana data pengukuran sudah berhasil diolah dan memberikan informasi terhadap performa komunikasi.

E. Desain Topologi Jaringan

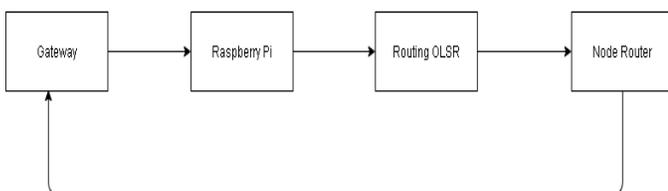
Node router yang digunakan sebanyak empat buah agar dapat melakukan perpindahan jalur *routing* sesuai dengan cara kerja *wireless mesh network*. Salah satu *node router* terhubung dengan internet, dan menjadi *gateway* bagi *node router* lain yang terhubung secara nirkabel agar dapat terkoneksi ke internet. Ada dua *client* yang digunakan, yaitu Pc A dan Pc B. Dalam hal ini pada *client* akan dilakukan tes komunikasi menggunakan jaringan *Ad-Hoc* MANET yang telah di desain. Untuk desain dari topologi dapat dilihat pada gambar 1.

F. Blok diagram Sistem

Pada gambar 2 dapat dilihat cara kerja dari sistem komunikasi darurat bencana alam. *Gateway* berfungsi sebagai gerbang untuk masuknya jaringan internet. Yang diterima oleh raspberry untuk selanjutnya OLSR sebagai *routing* protokol yang digunakan untuk merutekan trafik jaringan internet ke tiap *node router* raspberry lainnya. Lalu *node client*, bisa berkomunikasi dengan *client* lain dengan terhubung ke jaringan yang ada pada tiap *node router*.



Gambar 1. Rancangan Topologi Jaringan



Gambar 2. Blok Diagram Sistem

G. Flowchart Sistem

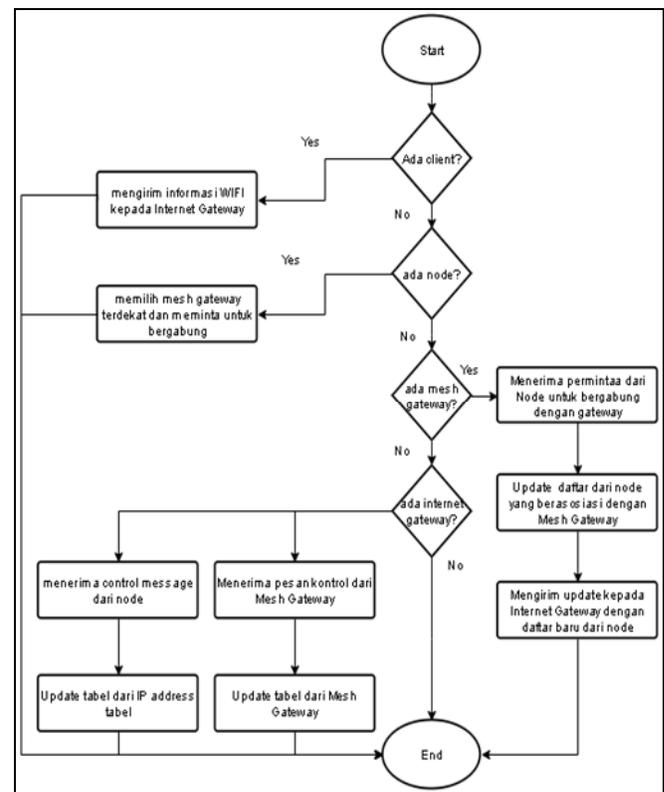
Pada tahap ini akan dijelaskan alur dari sistem yngakan dibuat dalam penelitian ini. Saat dimulai jika terdapat *client*, *client* akan mencari *node* terdekat untuk request masuk ke dalam jaringan *Mesh Gateway* untuk terhubung ke jaringan *Internet Gateway*. Setelah *client* sudah terhubung dengan jaringan Mesh, mereka dapat berinteraksi dengan *client* 1.

III. HASIL DAN PEMBAHASAN

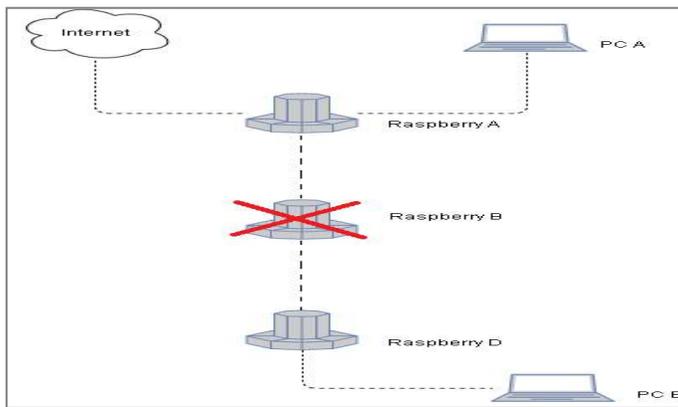
A. Pengujian self-configure OLSR pada Raspberry Pi

Pengujian self-configure OLSR pada tiap *node* Raspberry Pi dilakukan untuk mencari data waktu yang diperlukan bagi suatu *node* untuk bergabung dengan jaringan *wireless mesh network* yang sudah terbentuk. Pengujian ini dilakukan dengan mengubah besarnya variabel *hello message interval* dan *TC message interval* dari paket OLSR yang terdapat pada sudo nano /etc/olsrd/olsrd.conf . Skenario pengujian *self-configure* dapat dilihat pada Gambar 4.

Pengujian dari self-configure *node* Raspberry dilakukan dengan cara *node* Raspberry A melakukan ping terus menerus kepada Raspberry D. Setelah itu, *Node* Raspberry B dimatikan maka *node* Raspberry D yang berada di luar jangkauan dari *node* Raspberry A akan terputus dan menghasilkan *request time out* pada ping. Kemudian, agar *node* A dan D terhubung kembali dihidupkan kembali *node* B. Kemampuan dari *node* B ini untuk terhubung kembali dengan jaringan dinamakan sebagai *self-configure*.



Gambar 3. Flowchart Sistem



Gambar 4. Skenario pengujian self-configure

```
File Edit Tabs Help
GNU nano 2.2.6 File: /etc/olsrd/olsrd.conf
# Ip6MulticastGlobal ff0e::1
# Emission intervals.
# If not defined, RFC proposed values will
# be used in most cases.
# Hello interval in seconds(float)
HelloInterval 6.0
# HELLO validity time
HelloValidityTime 600.0
# TC interval in seconds(float)
TcInterval 0.5
# TC validity time
TcValidityTime 300.0
```

Gambar 5 Variabel hello Message dan TC

TABEL I. TABEL PENGUJIAN WAKTU SELF-CONFIGURE PADA VARIABEL TC

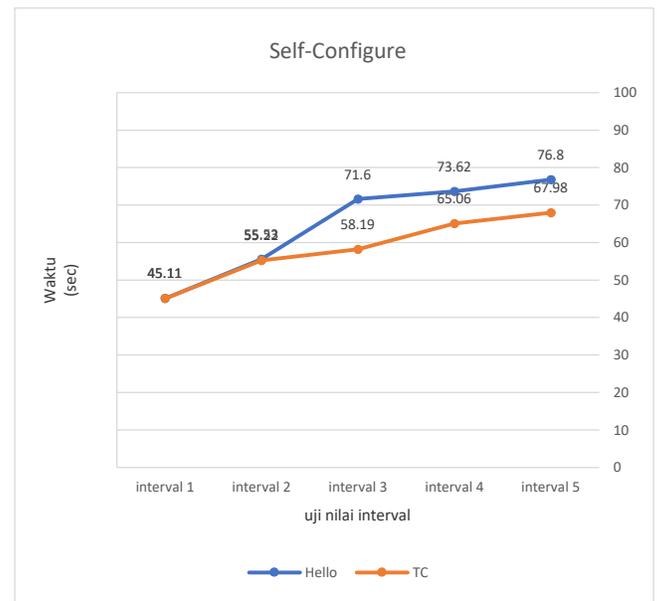
No	Hello Interval (detik)	TC Interval (detik)	Waktu (detik)
1	5	2	45.11
2	5	5	55.22
3	5	10	58.19
4	5	15	65.06
5	5	20	67.98

Pengujian yang diambil adalah data dari waktu pertama respon ping dari node A ke D ketika node B dihidupkan kembali dapat dilihat pada gambar 5.

Untuk hasil yang didapatkan dari pengujian self-configure dari wireless mesh network untuk nilai interval dari variabel Hello message yang berbeda pada Tabel I dan II.

TABEL II. TABEL PENGUJIAN WAKTU SELF-CONFIGURE PADA VARIABEL HELLO MESSAGE

No	Hello Interval (detik)	TC Interval (detik)	Waktu (detik)
1	5	2	45.11
2	10	2	55.53
3	15	2	71.60
4	20	2	73.62
5	25	2	76.80

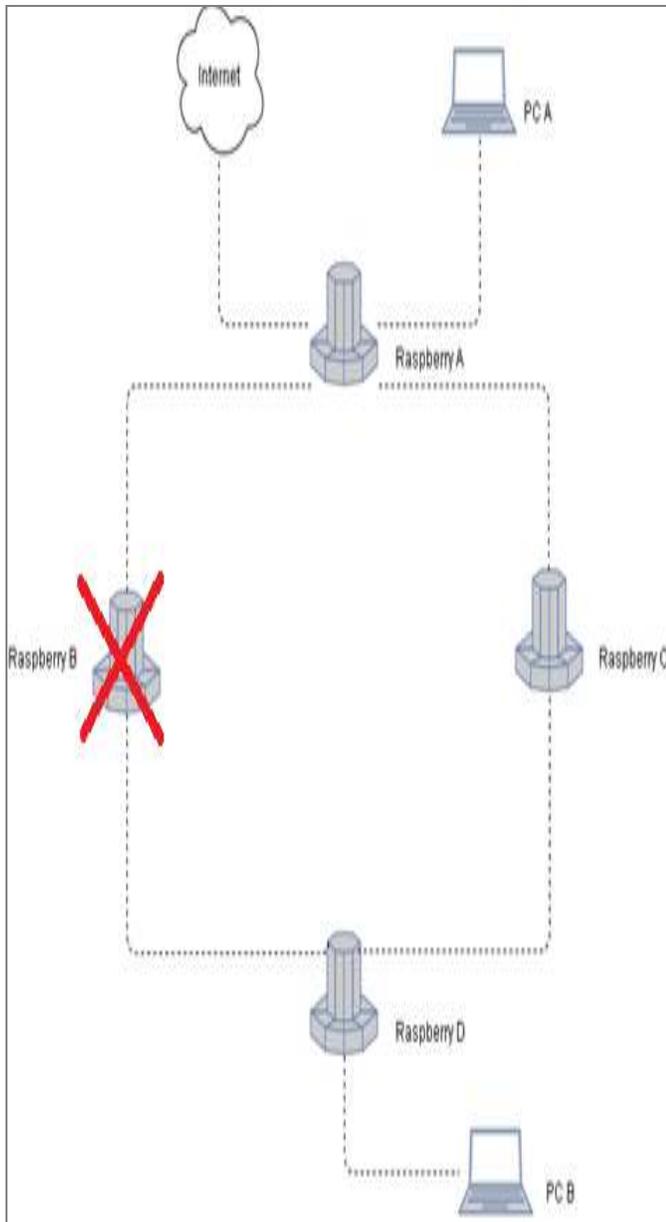


Gambar 6. Grafik perbandingan hello dan TC message untuk self-configure

Dapat dilihat dengan lebih jelas pada gambar 6 melalui grafik bahwa besarnya nilai hello message lebih berpengaruh terhadap waktu self-configure dibandingkan nilai TC message.

B. Pengujian self-healing OLSR pada Raspberry Pi

Skenario pengujian kedua adalah self-healing untuk mencari data waktu yang diperlukan bagi jaringan untuk mencari jalur baru maupun memperbaiki jalur apabila jaringan putus pada suatu node router. Variabel pengujian menggunakan kembali perubahan nilai interval pada hello dan TC message dari paket OLSR yang terdapat pada file /etc/olsrd/olsrd.conf. Berikut adalah skenario pengujian self-healing dapat dilihat pada gambar 7.



Gambar 7 . Skenario pengujian *self-healing*

Untuk pengujian *self-healing* pada raspi A berhubungan dengan raspi D melakukan ping terus-menerus melalui raspi B. Kemudian jalur raspi B dimatikan sehingga raspi C menjadi jalur selanjutnya karena *routing* protokol OLSR akan mencari jalur baru untuk menghubungkan jalur yang terputus ke raspi D, kemampuan inilah yang disebut dengan *self-healing*. Data yang diambil adalah waktu mulai respon pertama ping pada *node* raspi A dengan *node* raspi D melalui jalur C dapat diliha pada gambar 8 dan 9.

Destination	Gateway
0.0.0.0/0	10.10.10.1 (RaspiA.local.mesh)
10.10.10.1	10.10.10.1 (RaspiA.local.mesh)
10.10.10.2	10.10.10.2 (RaspiB.local.mesh)
192.168.137.0/24	10.10.10.1 (RaspiA.local.mesh)

Gambar 8. Jalur koneksi A-B-D

OLSR Routes in Kernel

Destination	Gateway
0.0.0.0/0	10.10.10.1 (RaspiA.local.r)
10.10.10.1	10.10.10.1 (RaspiA.local.r)
10.10.10.3	10.10.10.3 (RaspiC.local.r)
192.168.137.0/24	10.10.10.1 (RaspiA.local.r)

Links

Local IP	Remote IP
10.10.10.4 (RaspiD.local.mesh)	10.10.10.3 (RaspiC.local.mesh)

Gambar 9. Jalur koneksi A-C-D

TABEL III. TABEL PENGUJIAN WAKTU SELF-HEALING PADA VARIABEL HELLO MESSAGE

No	Hello Interval (detik)	TC Interval (detik)	Waktu (detik)
1	5	2	56.11
2	10	2	61.17
3	15	2	74.18
4	20	2	77.33
5	25	2	81.50

TABEL IV. TABEL PENGUJIAN WAKTU SELF-HEALING PADA VARIABEL TC MESSAGE

No	Hello Interval (detik)	TC Interval (detik)	Waktu (detik)
1	5	2	51.47
2	5	5	57.34
3	5	10	64.47
4	5	15	69.31
5	5	20	71.27

Berdasarkan hasil dari pengujian yang terdapat pada tabel III dan IV pada sebuah *wireless mesh network*, sebuah *node router* akan secara otomatis mencari jalur sendiri jika terjadi kerusakan pada jalur yang dilalui sebelumnya. Ini terjadi karena *routing* protokol OLSR akan mendeteksi *node* tetangga (*Neighbour Detection*), apabila saat *hello message* yang tidak dikirimkan tidak ada update informasi lagi tentang suatu *node*. *Hello message* yang tidak diterima lagi updatenya tentang suatu *node* dalam batas waktu tertentu akan menandakan bahwa hubungan terhadap *node* tersebut telah putus. Dan secara otomatis akan mencari jalur baru, dan *wireless mesh network* akan membentuk topologi baru. Dan pada *node raspi* yang terputus akan menghasilkan *ping request time out* pada saat melakukan ping.

Hampir sama dengan self-configure bahwa besarnya nilai interval *hello message* sangat berpengaruh sekali dibandingkan variabel dari nilai *TC message* terhadap kemampuan *self-healing* untuk *node router* mencari jalur baru ketika jalur putus atau terdapat kerusakan. Untuk grafik dapat dilihat pada gambar 10.

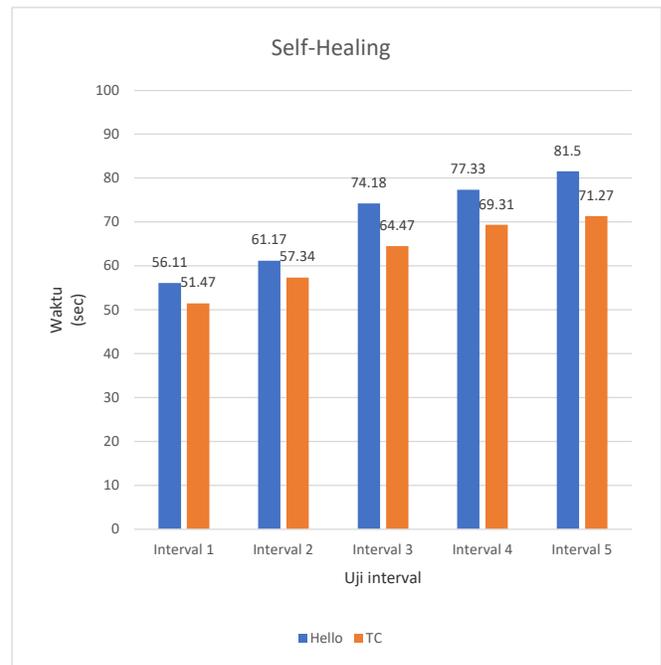
C. Pengujian Bandwidth Dengan Iperf

Dengan menggunakan iPerf maka harus ada perangkat yang berfungsi untuk menjalankan fungsi *server* dan ada perangkat yang berfungsi sebagai *client*. Pada *wireless mesh network* ini digunakan untuk menguji kecepatan *bandwidth* pada masing-masing *node router* dalam sistem *multihop*. Dan skenario

TABEL V. PENGGUNAAN IPERF

No	Nama	Status	Perintah
1	Raspberry Pi (Client)	Client	*iperf -c x.x.x.x -t 60
2	Raspberry Pi (server)	Server	*iperf -s

Keterangan : - iperf=*utility* yang digunakan
 - c=*mode client*
 - X.x.x.x=IP Raspberry *server* sebagai *gateway*
 --t 60=pengambilan data dilakukan dalam 60 detik
 - s=*mode server*

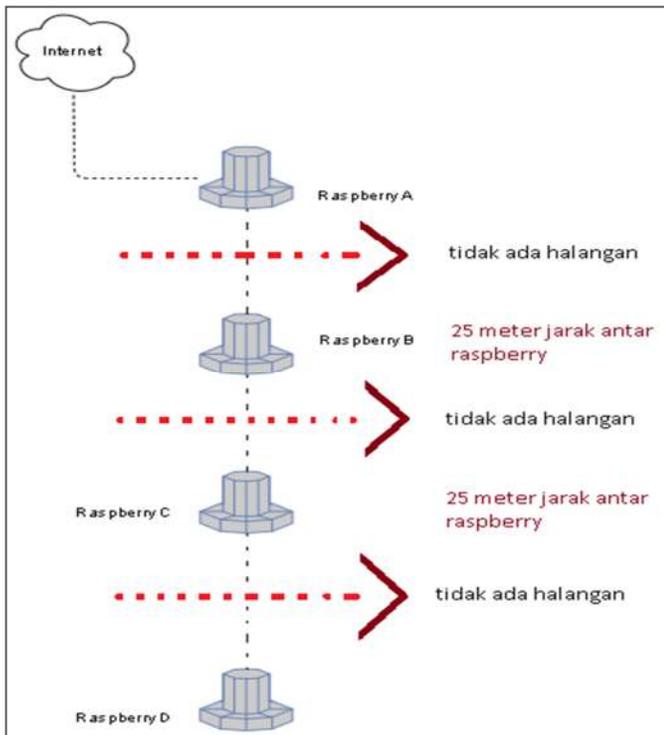


Gambar 10. Grafik perbandingan hello dan TC message untuk self-healing

pengujian dilakukan dengan 3 kali pengujian 1 *hop*, 2 *hop*, dan 3 *hop*. Dengan lima kali pengujian pada setiap skenario. Pada setiap skenario juga terdapat variabel LoS (*Line of Sight*) dan juga Non LoS itu adalah variabel tanpa hambatan dan ada hambatan. Aplikasi iperf diinstal pada ketiga *node router*, salah satu *node* dijadikan *server* dan *node* lain dijadikan *client*. Skenario pengujian dengan menggunakan iPerf dapat dilihat pada tabel V.

1) Pengujian Iperf dengan tanpa hambatan LoS (*Line of Sight*)

Pengujian pertama dengan tanpa ada halangan diuji untuk melihat seberapa besar kecilnya ukuran *bandwidth* yang dihasilkan pada tiap *hop*. diberikan. Skenario pengujian tanpa hambatan dapat dilihat pada gambar 11.



Gambar 11. Skenario pengujian tanpa halangan

```

pi@Raspia:~$ iperf -s
-----
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
-----
[ 4] local 192.168.43.35 port 5001 connected with 10.10.10.3 port 43270
[ ID] Interval      Transfer    Bandwidth
[ 4] 0.0-110.4 sec  256 KBytes  19.0 Kbits/sec
[ 5] local 192.168.43.35 port 5001 connected with 10.10.10.3 port 43272
[ 5] 0.0-215.5 sec  384 KBytes  14.6 Kbits/sec
[ 4] local 192.168.43.35 port 5001 connected with 10.10.10.3 port 43274
[ 4] 0.0-75.0 sec   512 KBytes  55.9 Kbits/sec
[ 5] local 192.168.43.35 port 5001 connected with 10.10.10.3 port 43276
[ 5] 0.0-136.2 sec  1.00 MBytes  61.6 Kbits/sec
[ 4] local 192.168.43.35 port 5001 connected with 10.10.10.3 port 43278
[ 4] 0.0-59.7 sec   256 KBytes  35.1 Kbits/sec
[ 5] local 192.168.43.35 port 5001 connected with 10.10.10.3 port 43280
[ 5] 0.0-44.6 sec   640 KBytes  118 Kbits/sec
[ 4] local 192.168.43.35 port 5001 connected with 10.10.10.3 port 43282
[ 4] 0.0-85.5 sec   256 KBytes  24.5 Kbits/sec
    
```

Gambar 12. hasil uji coba tanpa hambatan dengan 1 hop

```

pi@Raspia:~$ iperf -s
-----
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
-----
[ 4] local 192.168.43.35 port 5001 connected with 10.10.10.2 port 57310
[ ID] Interval      Transfer    Bandwidth
[ 4] 0.0-66.1 sec  4.12 MBytes  524 Kbits/sec
[ 5] local 192.168.43.35 port 5001 connected with 10.10.10.2 port 57312
[ 4] local 192.168.43.35 port 5001 connected with 10.10.10.2 port 57314
[ 4] 0.0-361.2 sec  474 KBytes  10.7 Kbits/sec
[ 6] local 192.168.43.35 port 5001 connected with 10.10.10.2 port 57316
[ 4] local 192.168.43.35 port 5001 connected with 10.10.10.2 port 57318
[ 4] 0.0-670.4 sec  36.8 KBytes  449 bits/sec
[ 7] local 192.168.43.35 port 5001 connected with 10.10.10.2 port 57320
[ 7] 0.0-717.4 sec  5.12 MBytes  59.9 Kbits/sec
[ 4] local 192.168.43.35 port 5001 connected with 10.10.10.2 port 57322
[ 4] 0.0-864.1 sec  1.25 MBytes  12.1 Kbits/sec
[ 7] local 192.168.43.35 port 5001 connected with 10.10.10.2 port 57324
[ 4] local 192.168.43.35 port 5001 connected with 10.10.10.2 port 57326
[ 4] 0.0-1217.0 sec  512 KBytes  3.45 Kbits/sec
[ 8] local 192.168.43.35 port 5001 connected with 10.10.10.2 port 57328
[ 8] 0.0-1279.7 sec  1.50 MBytes  9.83 Kbits/sec
[ 4] local 192.168.43.35 port 5001 connected with 10.10.10.2 port 57330
[ 4] 0.0-1326.7 sec  640 KBytes  3.95 Kbits/sec
[ 8] local 192.168.43.35 port 5001 connected with 10.10.10.2 port 57332
[ 8] 0.0-1450.2 sec  1.00 MBytes  5.78 Kbits/sec
[ 4] local 192.168.43.35 port 5001 connected with 10.10.10.2 port 57334
[ 4] 0.0-1530.9 sec  1.00 MBytes  5.48 Kbits/sec
    
```

Gambar13. hasil uji coba tanpa hambatan dengan 2 hop

```

pi@Raspia:~$ iperf -s
-----
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
-----
[ 4] local 192.168.43.35 port 5001 connected with 10.10.10.4 port 34240
[ ID] Interval      Transfer    Bandwidth
[ 4] 0.0-98.8 sec  256 KBytes  21.2 Kbits/sec
[ 5] local 192.168.43.35 port 5001 connected with 10.10.10.4 port 34242
[ 5] 0.0-236.8 sec  512 KBytes  17.7 Kbits/sec
[ 4] local 192.168.43.35 port 5001 connected with 10.10.10.4 port 34244
[ 4] 0.0-53.7 sec   384 KBytes  58.6 Kbits/sec
[ 5] local 192.168.43.35 port 5001 connected with 10.10.10.4 port 34246
[ 5] 0.0-36.7 sec   768 KBytes  171 Kbits/sec
[ 4] local 192.168.43.35 port 5001 connected with 10.10.10.4 port 34248
[ 4] 0.0-69.2 sec   384 KBytes  45.5 Kbits/sec
    
```

Gambar 14. Hasil uji coba tambatan dengan 3 hop

TABEL VI. HASIL PENEGUJIAN IPERF 1 HOP TANPA HAMBATAN

No	Transfer (Mbytes)	Bandwidth (Mbits/detik)
1	4.15	0.52
2	5.12	0.05
3	1.25	0.01
4	1.5	0.09
5	1.0	0.05
Rata-rata	2.6	0.14

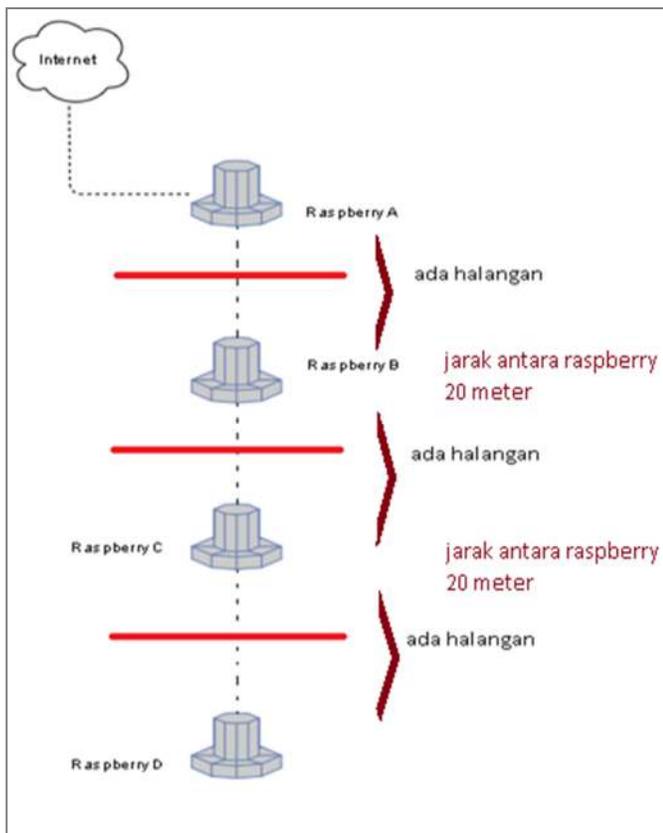
TABEL VII. HASIL PENEGUJIAN IPERF 2 HOP TANPA HAMBATAN

No	Transfer (Mbytes)	Bandwidth (Mbits/detik)
1	0.25	0.02
2	0.38	0.01
3	0.51	0.05
4	1.00	0.06
5	0.25	0.03
Rata-rata	0.47	0.03

Pengujian yang akan dilakukan yaitu tanpa hambatan dengan 1 hop, 2 hop dan 3 hop dapat dilihat pada gambar 12, 13, dan 14. Untuk asil yang didapatkan dapat dilihat juga pada tabel VI, VII, dan VIII.

TABEL VIII. HASIL PENEGUJIAN IPERF 3 HOP TANPA HAMBATAN

No	Transfer (Mbytes)	Bandwidth (Mbits/detik)
1	0.25	0.02
2	0.51	0.01
3	0.38	0.05
4	0.76	0.10
5	0.38	0.04
Rata-rata	0.45	0.04



Gambar 15. Skenario pengujian dengan halangan

2) Pengujian iperf dengan hambatan Non LoS

Pengujian kedua dengan adanya hambatan atau penghalang pada setiap *hop*. Dengan hambatan yang digunakan adalah pintu, dinding, dan juga pohon yang terdapat pada setiap *hop* dengan jarak antar *hop* sebesar lebih kurang 20 meter. Berikut adalah gambar skenario dari untuk pengujian menggunakan hambatan. Untuk scenario dapat dilihat pada gambar 15.

Untuk pengujian 1 *hop* dengan hambatan dengan jarak antar *node* sebesar lebih kurang 20 meter pada waktu sore hari dengan cuaca berawan. Untuk pengujian dapat dilihat pada gambar 16, 17, 18 dan untuk hasil dapat dilia pada tabel IX, X, X.

```

pi@Rasp1A: ~ $ iperf -s
-----
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
-----
[ 4] local 192.168.43.35 port 5001 connected with 10.10.10.2 port 53560
[ ID] Interval      Transfer    Bandwidth
[ 4] 0.0-38.5 sec  512 KBytes  109 Kbits/sec
[ 5] local 192.168.43.35 port 5001 connected with 10.10.10.2 port 53562
[ 5] 0.0-87.3 sec  1.00 MBytes  96.1 Kbits/sec
[ 4] local 192.168.43.35 port 5001 connected with 10.10.10.2 port 53566
[ 4] 0.0-42.1 sec  768 KBytes  149 Kbits/sec
[ 5] local 192.168.43.35 port 5001 connected with 10.10.10.2 port 53568
[ 5] 0.0-63.6 sec  256 KBytes  33.0 Kbits/sec
[ 4] local 192.168.43.35 port 5001 connected with 10.10.10.2 port 53570
[ 4] 0.0-37.7 sec  640 KBytes  139 Kbits/sec

```

Gambar 16. Hasil uji coba dengan hambatan 1 hop

```

pi@Rasp1A: ~ $ iperf -s
-----
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
-----
[ 4] local 192.168.43.35 port 5001 connected with 10.10.10.3 port 54992
[ ID] Interval      Transfer    Bandwidth
[ 4] 0.0-103.5 sec  256 KBytes  20.3 Kbits/sec
[ 4] local 192.168.43.35 port 5001 connected with 10.10.10.3 port 56774
[ 4] 0.0-47.4 sec  384 KBytes  66.4 Kbits/sec
[ ID] Interval      Transfer    Bandwidth
[ 4] 0.0-38.3 sec  384 KBytes  82.2 Kbits/sec
[ 4] local 192.168.43.35 port 5001 connected with 10.10.10.3 port 56770
[ 4] 0.0-46.5 sec  256 KBytes  45.1 Kbits/sec
[ 5] local 192.168.43.35 port 5001 connected with 10.10.10.3 port 38204
[ ID] Interval      Transfer    Bandwidth
[ 4] 0.0-50.4 sec  640 KBytes  104 Kbits/sec

```

Gambar 17. Hasil uji coba dengan hambatan 2 hop

```

pi@Rasp1D: ~ $ iperf -c 192.168.43.35 -t 30
-----
Client connecting to 192.168.43.35, TCP port 5001
TCP window size: 43.8 KByte (default)
-----
[ 3] local 10.10.10.4 port 56786 connected with 192.168.43.35 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 3] 0.0-40.9 sec  384 KBytes  76.9 Kbits/sec
[ 3] local 10.10.10.4 port 56770 connected with 192.168.43.35 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 3] 0.0-45.3 sec  256 KBytes  46.3 Kbits/sec
[ 3] local 10.10.10.4 port 56774 connected with 192.168.43.35 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 3] 0.0-43.8 sec  384 KBytes  71.8 Kbits/sec
[ 3] local 10.10.10.4 port 56780 connected with 192.168.43.35 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 3] 0.0-241.7 sec  512 KBytes  17.4 Kbits/sec
[ 3] local 10.10.10.4 port 56786 connected with 192.168.43.35 port 5001
[ ID] Interval      Transfer    Bandwidth
[ 3] 0.0-40.9 sec  384 KBytes  76.9 Kbits/sec

```

Gambar 18. Hasil uji coba dengan hambatan 3 hop

TABEL IX. HASIL PENEGUJIAN IPERF 1 HOP DENGAN HAMBATAN

No	Transfer (Mbytes)	Bandwidth (Mbits/detik)
1	0.51	0.10
2	1.00	0.09
3	0.76	0.10
4	0.25	0.03
5	0.64	0.10
Rata-rata	0.63	0.08

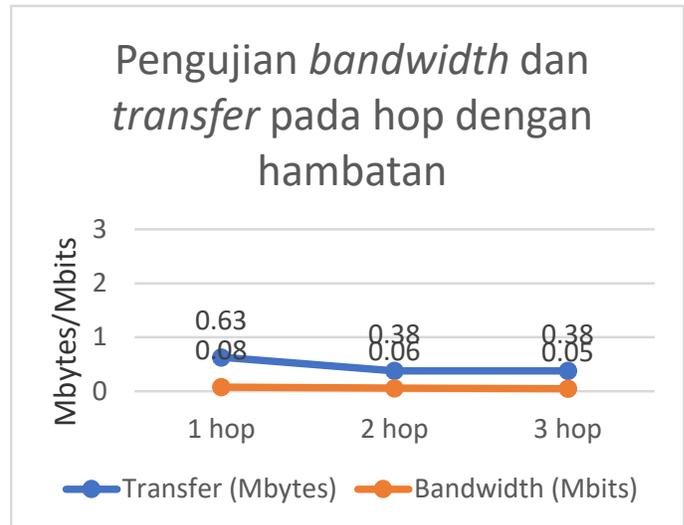
TABEL X. HASIL PENEGUJIAN IPERF 2 HOP DENGAN HAMBATAN

No	Transfer (Mbytes)	Bandwidth (Mbits/detik)
1	0.25	0.02
2	0.38	0.06
3	0.38	0.08
4	0.25	0.04
5	0.64	0.10
Rata-rata	0.38	0.06

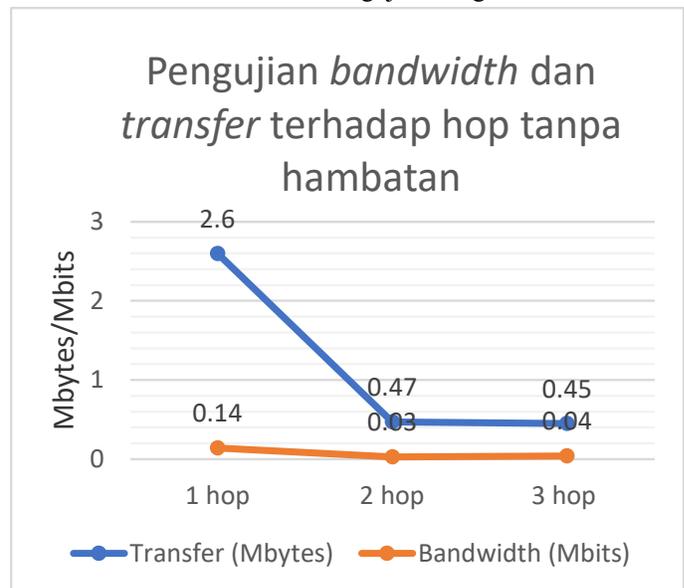
TABEL XI. HASIL PENEGUJIAN IPERF 3 HOP DENGAN HAMBATAN

No	Transfer (Mbytes)	Bandwidth (Mbits/detik)
1	0.38	0.07
2	0.25	0.04
3	0.38	0.07
4	0.51	0.01
5	0.38	0.07
Rata-rata	0.38	0.05

Grafik perbandingan *bandwidth* dan transfer antar *hop* dengan hambatan dan tanpa hambatan. Dapat dilihat dari kedua grafik diatas bahwa naik turunnya transfer atau *throughput* dengan *bandwidth*, selain dipengaruhi dengan banyaknya *hop* juga dipengaruhi oleh ada tidaknya hambatan. Perbandingannya dapat dilihat pada gambar 19 dan 20.



Gambar 19. Grafik Pengujian dengan hambatan



Gambar 20. Grafik Pengujian tanpa hambatan

3) Uji File Transfer Manet 2 node

Pada pengujian ini *File Transfer* diterapkan pada skenario pengujian MANET dengan dua *node*. Pengujian ini merupakan yang paling sederhana dikarenakan dua *node* di dalamnya yang masing-masing berperan sebagai *client* dan *server* saling terhubung tanpa adanya jalur dari *node gateway*. Pengujian dimulai dengan membangun infrastruktur MANET yang terdiri dari 2 *node* yaitu *node A* dan *node B*, kemudian *node B* menjalankan *FTP server* yang telah dikonfigurasi dan disesuaikan dengan seting ip dari manet yaitu 10.10.10.2. dapat dilihat pada gambar 21 sampai gambar 27.

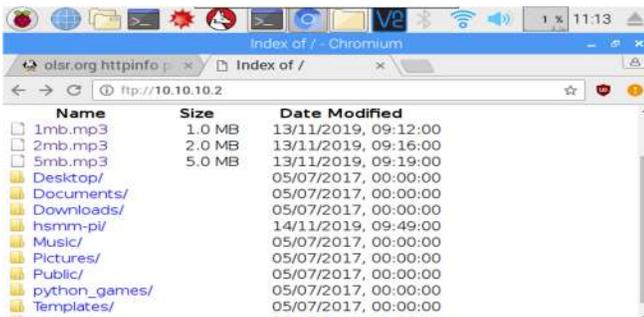
Proses selanjutnya *node A* yang berperan sebagai *client* akan mengakses *ftp server node B* menggunakan *browser* dengan memasukkan alamat `ftp://10.10.10.2`. Percobaan ini dapat dilihat pada gambar 21 sampai 25 dan untuk hasilnya dapat dilihat pada tabel XII.



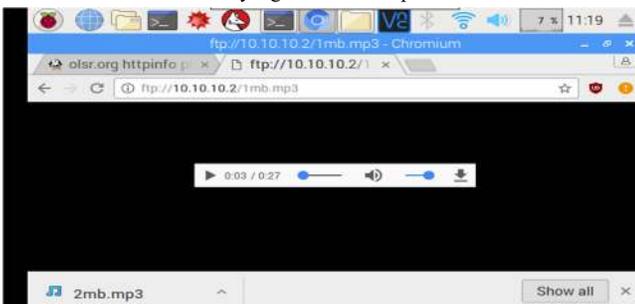
Gambar 21. uji transfer file 2 node



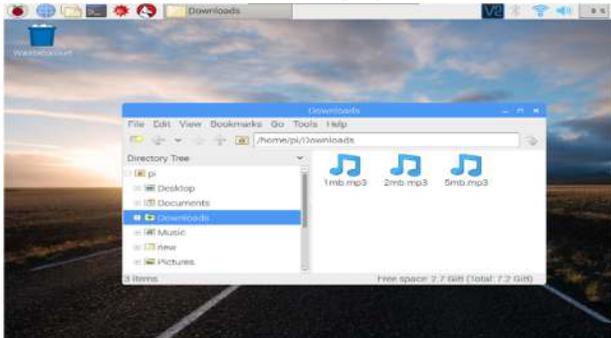
Gambar 22. Proses autentifikasi proftpd dengan *username* dan *password*



Gambar 23. File yang bisa diunduh pada direktori *node B*



Gambar 24. Proses pengunduhan file



Gambar 25. File yang telah diunduh dari *node B* pada *node A*

TABEL XII. HASIL PENGUJIAN WAKTU TRANSFER FILE 2 *NODE*

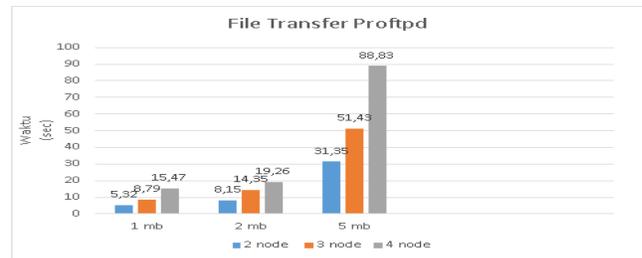
No	File (mb)	Waktu (sec)
1	1	5.32
2	2	8.15
3	5	31.35

TABEL XIII. HASIL PENGUJIAN WAKTU TRANSFER FILE 3 *NODE*

No	File (mb)	Waktu (sec)
1	1	8.79
2	2	14.35
3	5	51.43

TABEL XIV. HASIL PENGUJIAN WAKTU TRANSFER FILE 4 *NODE*

No	File (mb)	Waktu (sec)
1	1	15.47
2	2	19.26
3	5	88.83



Gambar 26. Grafik Pengujian FTP Proftpd

4) Uji File Transfer Manet 3 node

Pada tahap ini ditambahkan satu *node* lagi yaitu *node C* setelah *node B*. Jadi terdapat 1 *hop* yang akan dilewati dalam proses *File Transfer* dari A-B-C.

Proses selanjutnya *node A* yang berperan sebagai *client* akan mengakses ftp *server node C* menggunakan *browser* dengan memasukkan alamat ftp://10.10.10.3. Jika setelah memasukkan alamat tersebut terdapat *authentication* maka perlu dilakukan *login* terlebih dahulu dengan memasukkan *username* dan *password* yang telah ditentukan sebelumnya. Untuk hasilnya dapat dilihat pada tabel XIII.

5) Uji File Transfer Manet 4 node

Pada tahap terakhir ini ditambahkan satu *node* lagi yaitu *node D* setelah *node C*. Jadi terdapat 2 *hop* yang akan dilewati dalam proses *File Transfer* dari A-B-C-D. Proses selanjutnya *node A* yang berperan sebagai *client* akan mengakses ftp *server node D* menggunakan *browser* dengan memasukkan alamat ftp://10.10.10.4. Jika setelah memasukkan alamat tersebut terdapat *authentication* maka perlu dilakukan *login* terlebih dahulu dengan memasukkan *username* dan *password* yang telah ditentukan sebelumnya. Untuk melihat hasil dari transfer ini dapat dilihat pada tabel XIV.

IV. PENUTUP

A. Kesimpulan

Berdasarkan hasil dari pengujian dan eksperimen dari penelitian ini maka dapat disimpulkan mengacu pada hasil pengujian yang telah dilakukan, bahwa sistem komunikasi darurat menggunakan Raspberry Pi bisa dilakukan menggunakan teknologi *Mobile Ad-hoc Network* dalam jaringan yang memiliki topologi dinamis.

Berdasarkan hasil yang didapat bahwa kinerja *routing protocol* OLSR berguna untuk menggantikan fungsi router yaitu untuk merutekan trafik dari sumber ke tujuan

Komunikasi dapat dilakukan pada jaringan MANET menggunakan laptop dengan OS windows 7 atau bisa dikatakan hanya dengan perangkat yang mendukung konektivitas pada jaringan *Ad-Hoc*

Routing Protokol OLSR juga terbukti mampu memperbaiki sistemnya sendiri karena kemampuan *self-healing* dan *self-configure*, serta variabel yang mempengaruhi performansi adalah besarnya nilai interval Hello *message* dibandingkan TC *message*.

Besarnya nilai *Bandwidth* dan *Transfer* dipengaruhi oleh banyaknya *hop* yang dilalui juga adanya hambatan juga dapat memberikan dampak bagi menurunnya *Bandwidth* dan *Transfer*.

B. Saran

Saran yang dapat diberikan untuk penelitian dan pengembangan selanjutnya agar dapat membuat *security* pada jaringan *wireless mesh network* juga untuk bisa mengembangkan aplikasi atau perangkat yang bisa digunakan untuk berkomunikasi secara langsung pada jaringan *Ad-hoc* selain laptop windows 7.

TENTANG PENULIS



Audy Gerald Palilingan, yang biasa dipanggil dengan nama Ody lahir di Bogor pada tanggal 9 Juli 1993. Alamat tempat tinggal penulis di Kelurahan Bahu Lingkungan 9, Kota Manado. Penulis menempuh Pendidikan formal secara berturut-turut di SD Shalom II BK-3 Tangerang (1999-2005), SMP Strada St. Maria 2 Tangerang (2005-2008), SMA Negeri 6 Tangerang (2008-2011). Pada tahun 2014, penulis melanjutkan Pendidikan S1 di Program Studi Teknik Informatika, Jurusan Teknik Elektro, Fakultas Teknik, Universitas Sam Ratulangi. Selama kuliah penulis juga tergabung dalam organisasi mahasiswa yaitu, Himpunan Mahasiswa Elektro (HME), menjadi bagian dari POSITIVISME, dan berada dalam komunitas UNSRAT IT Community (UNITY). Hingga akhirnya pada bulan Januari 2020 saya dapat menyelesaikan studi S1 dengan hasil yang baik.

V. KUTIPAN

- [1] Y. Sidharta, "PERBANDINGAN UNJUK KERJA PROTOKOL ROUTING AD HOC ON-DEMAND DISTANCE VECTOR (AODV) DAN DYNAMIC SOURCE ROUTING (DSR) PADA JARINGAN MANET," *J. Teknol.*, vol. 6, no. 1, pp. 83–89, 2013.
- [2] Cisco, "Creating a PDF of the Internetworking Technology Handbook Internetworking Basics," pp. 1–7, 2004.
- [3] C. Kopp, "Adhoc networking," pp. 33–40, 2002.
- [4] P. Onno, "Optimized Link State Protocol(OLSR)," [Online]. Available: <http://cyberlearning.web.id/wiki/index.php/OLSR> [Online].
- [5] R. Munadi, "Pengujian Polarisasi Linier Terhadap Kualitas SNR Pada Sistem Komunikasi Nirkabel IEEE 802.11g," *Penguji. Polarisasi Linier Terhadap Kualitas SNR Pada Sist. Komun. Nirkabel IEEE 802.11g*, vol. 9, no. 3, pp. 132–136, 2011, doi: 10.17529/jre.v9i3.162.
- [6] S. Salem, "MANET Testbed Using Raspberry Pis," *IJWMT (International Journal of Wireless and Microwave Technologies)*, *Int. J. Wirel. Microw. Technol.*, vol. 8, no. 2, pp. 52–63, 2018.
- [7] M. L. Febriadi, A. F. Rochim, and E. D. Widiyanto, "Perencanaan dan Implementasi *Wireless Mesh Node* pada Raspberry Pi," *J. Teknol. dan Sist. Komput.*, vol. 1, no. 4, pp. 145–154, 2013, doi: 10.14710/JTSISKOM.1.4.2013.145-154.
- [8] U. S. Permatasari, "Analisis *Routing* Protokol Optimized Link State *Routing* (OLSR) Pada Raspberry Pi," 2016.
- [9] and L. M. F. A. R.D. Kurniawan, M. Yamin, "Penerapan Protokol *Routing* Proaktif OLSR Pada Jaringan *Mobile Ad-Hoc Network* (MANET) Menggunakan Raspberry Pi Untuk Transfer File," 2019.
- [10] D. Ary, "Introduction to Research in Education," 2010.