Penerapan Aplikasi *Chat* berbasis Python pada MANET (*Mobile Ad-Hoc Network*) menggunakan *Routing Protocol* Babel

Aditya Wijayanto[1], Rifki Adhitama[2]
[1,2] (Rekayasa Perangkat , Fakultas Teknologi Industri dan Informatika, Institut Teknologi Telkom Purwokerto)
[1,2] (Jl. D.I Panjaitan No. 128, Purwokerto, 53147, Indonesia)
[1]aditya.wijayanto@ittelkom-pwt.ac.id, [2]rifki@ittelkom-pwt.ac.id

Infrastruktur komunikasi data nirkabel diperlukan agar perangkat bergerak nirkabel *(wireless mobile device*) dapat berkomunikasi satu dengan yang lain. Pada beberapa kondisi seperti pada pemulihan daerah bencana dan medan perang, dimana infrastruktur komunikasi data rusak atau tidak dapat dimanfaatkan, teknologi *mobile ad-hoc network* dapat digunakna untuk komunikasi data alternatif. Salah satu sistem komunikasi adalah model pengiriman pesan dengan menggunakan aplikasi *Chat* berbasis Python. Aplikasi *Chat* diguanakan pada setiap *node* dalam MANET untuk berkomunikasi dengan *node* MANET lain dengan menggunakan pesan berbasis teks. Dari penelitian yang dilakukan diperoleh hasil bahwa Aplikasi *Chat* dapat berjalan dengan optimal di platform Phyton pada MANET menggunakan *Routing Protocol* Babel.

Kata kunci : Protokol Perutean; Babel; MANET; Chat

A wireless data communications infrastructure is required for wireless mobile devices to communicate with each other. In some conditions such as disaster and battlefield disaster recovery, where data communications infrastructures are damaged or can not be utilized, mobile ad-hoc network technology can be used for alternative data communications. One of the communication systems is a messaging model by using the Python-based Chat app. Chat apps are used on each node in MANET to communicate with other MANET nodes by using text-based messaging. The result of this research indicates that Chatting Applications can run optimally on the platform Phyton on the MANET using Routing Protocol Babel.

Keywords : Routing Protocols; Babel; MANET;Chat

I. PENDAHULUAN

Infrastruktur komunikasi data nirkabel diperlukan agar perangkat bergerak nirkabel (*wireless mobile device*) dapat berkomunikasi satu dengan yang lain. Pada beberapa kondisi seperti pada pemulihan daerah bencana dan medan perang, dimana infrastruktur komunikasi data rusak atau tidak dapat dimanfaatkan [1], Teknologi MANET (*Mobile Ad-Hoc Network*) dapat digunakan untuk komunikasi data alternatif.

Dalam realita penggunaan MANET dapat diasumsikan "hubungan dimana saja dan kapan saja", karena itulah implementasi MANET sangat berguna dalam berbagai bidang aplikasi seperti pemulihan daerah bencana, pencairan dan penyelamatan korban, komunikasi dalam medan perang, kebutuhan pendidikan, *entertainment, robot*, dan *sensor network* [2].Pesatnya pekembangan teknologi nirkabel, semakin kecilnya volume baterai dalam menampung arus listrik dan perkembangan *chip wireless* yang semakin kecil namun berkekuatan sinyal yang baik, *Mobile Ad-Hoc Network* (MANET) muncul sebagai akibat dari perkembangan teknologi di atas.

Secara singkat MANET adalah sistem otonom dimana setiap *node* beroperasi tidak hanya sebagai diri sendiri tetapi juga sebagai *router/bridge* untuk meneruskan paket menuju *node* lain[3]

Agar pemanfaatannya optimal, MANET membutuhkan algoritma routing yang handal sehingga paket – paket data yang dikirmkan simpul sumber dapat diterima dengan utuh oleh simpul tujuan. Babel termasuk dalam *routing protocol* proaktif, Babel adalah salah satu *protocol routing distance vector* yang memiliki performa *fast convergence* dan *fast route repair time* [4].

Salah satu sistem komunikasi yang dapat diimplementasikan dalam MANET adalah sistem komunikasi pengiriman pesan (*Chat*), secara *broadcast* untuk sistem komunikasi pada jaringan tersebut.

II. METODE PENELITIAN

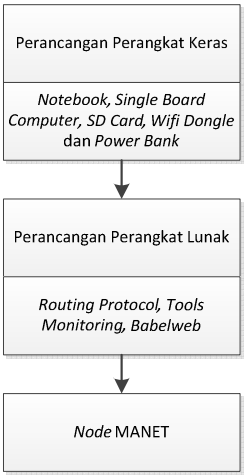
A. Analisis Masalah

Pada penelitian ini, akan dibangun sebuah jaringan model *Mobile Ad-Hoc Network* (MANET)

menggunakan *routing protocol* Babel. Perancangan ini akan diimpelementasikan secara langsung di lapangan, perancangan ini akan digunakan untuk melakukan pengujian penerapan aplikasi *Chat*.

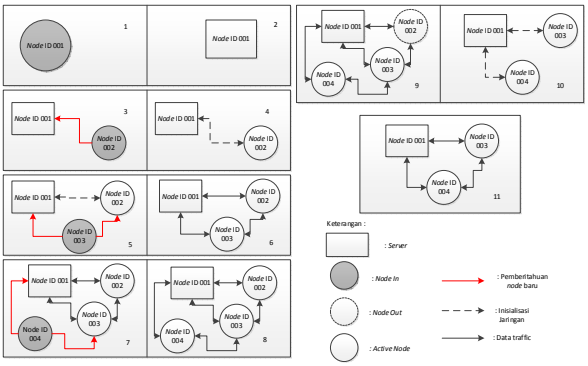
B. *Analisis Kebutuhan Sistem*

Analisis kebutuhan sistem bertujuan untuk membahas bagaimana sistem harus menyelesaikan masalah pada sistem yang akan dibangun serta mengidentifikasi kebutuhan yang terlibat untuk membangun sebuah *Mobile Ad-Hoc Network* (MANET). Analisis kebutuhan tersebut meliputi kebutuhan perangkat keras dan kebutuhan perangkat lunak.



Gambar 1. Alur Perancangan Sistem Secara Umum

Gambar 1 dijelaskan mengenai perancangan sistem secara keseluruhan, dimulai dari merancang perangkat keras sistem, yang terdiri dari *Notebook, Single Board Computer, Micro SDHC Card, Wifi Dongle* dan *Power Bank*. Setelah merancang perangkat keras selesai, kemudian merancang perangkat lunak yang terdiri dari sistem operasi Ubuntu, Raspbian, perancangan *routing protocol* yang menggunakan *routing protocol* Babel, Babelweb yang nanti akan digunakan sebagai visualisasi topologi MANET.

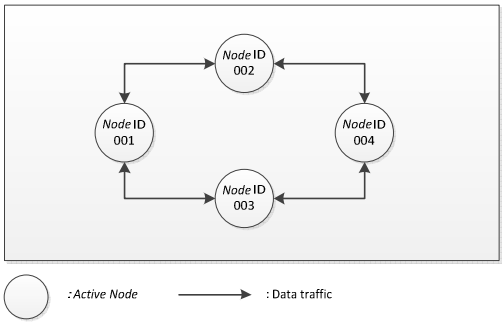


Gambar 2. Diagram Blok Ilustrasi terbentuknya MANET

Pada Gambar 2. terlihat bagaimana *node* membentuk sebuah jaringan *Node ID* 001 saat pertama diaktifkan mencara apakah ada perangkat yang sudah terbentuk. Ketika *Node ID* 004 masuk maka akan membentuk jaringan baru. *Node ID* 002 keluar dari jaringan, *Server* mengulang lagi proses konfigurasi dari awal dengan jumlah *Node* yang baru[5].

Skenario Pengujian

Pada Gambar 3. Pengujian aplikasi *Chat* pada MANET dengan kondisi *multihop* dengan jumlah 4 *node*.



Gambar 3. Diagram blok pengujian *Chat* dengan 4 *node*

C. *Perangkat Keras*

Implementasi perangkat keras terdiri dari implementasi sistem minimum *Notebook* dan SBC (instalasi sistem operasi dan aplikasi pendukung lainnya). Kemudian pemasangan Micro SDHC V-Gen 16 GB, Wifi dongle Ralink RT5370 dan Power Bank HAME T6 10.00 mAh. Implementasi dari keseluruhan perangkat keras ditunjukan pada Gambar 6.

Gambar 4. Implementasi keseluruhan perangkat keras.

D. *Implementasi Sistem*

Pada implementasi sistem, penulis akan melakukan instalasi seluruh perangkat lunak yang dibutuhkan pada setiap *node*.

```
// masuk halaman interfaces
pi@raspberry:~$ sudo nano
/etc/network/interfaces
// setting ad-hoc pada halaman interfaces
Auto wlan0
     Iface wlan0 inet static
address xxx.xxx.xxx.xxx //memberikan no ip pada
wlan0
netmask xxx.xxx.xxx.xxx
wireless-essid mesh //membuat nama essid agar
komputer lain mengenali
wireless-mode ad-hoc //membuat interface wlan0
menjadi mode ad-hoc
wireless-channel 7 //dipilih frekuensi dengan
channel 7
wireless-key off //mematikan kunci essid
```

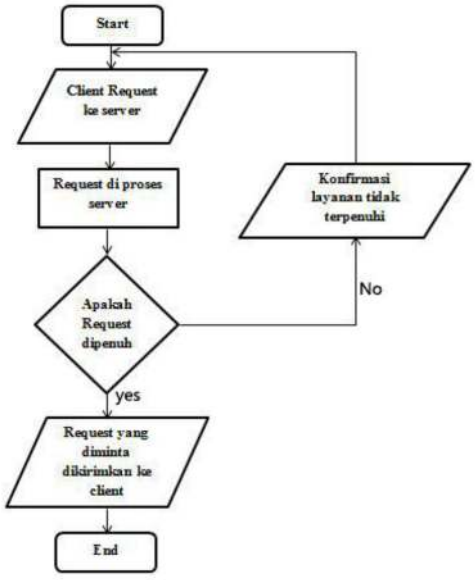
terminal telah ditunjukkan pada Gambar 5 supaya konfigurasi jaringan *ad-hoc* bersifat tetap, penulis melakukan konfigurasi melalui halaman *interfaces*, yang ada di */etc/network/interfaces*. *Interface wlan0* di *setting static* karena lebih mudah dalam melakukan kontrol dari setiap *node*, dimana salah satunya adalah jika ada *node* bermasalah misalnya belum terkoneksi akan mudah untuk melakukan *troubleshoot*.

```
        // instalasi protokol Babel
pi@raspberry:~$ sudo apt-get install babeld
```

Gambar 6. *Install Routing Protocol* Babel

Routing protocol yang akan digunakan adalah *routing protocol* Babel. Gambar 6 *Source code* untuk menginstall *routing protocol* Babel.

E. Aplikasi Chat



Gambar 7. *Flowchart* aplikasi *Chat*

Aktivitas *Client – Server* diperlihatkan pada Gambar 7. Diagram alir komunikasi *Chat* menunjukkan aktivitas antara *Server* dan *Client*. *Client* memulai permintaan layanan (*request*), *Server* menanggapi *request* tersebut (*response*), selanjutnya permintaan layanan *Client* diproses oleh *Server*. *Server* memperhatikan apakah *request Client* dapat dipenuhi atau tidak. Jika terpenuhi, *server* akan kembali mengirimkan kembali mengirimkan kembali hasil permintaan tersebut ke *Client*. Namun, apabia *request Client* tersebut tidak dipenuhi, *Server* akan mengkonfirmasikan *Client* bahwa permintaan layanan saat ini belum bisa dipenuhi. Selanjutnya, proses *Client – Server* kembali ke awal yaitu *Client* memulai permintaan layanan ke *Server*.

III. HASIL PENELITIAN

A. Pengujian Perangkat

Pengujian aplikasi *Chat* berbasis Phyton pada MANET dengan *routing protocol* Babel ini dengan menempatkann 4 *node* secara *multihop*. 2 *node* terhubung langsung dengan *node id* 001yaiu *node id* 002 dan *node id* 003. 1 *node* lagi tidak terhubung langsung dengan *node id* 001, yaitu *node id* 004. Ketika *node id* 004 akan berkomunikasi dengan *node id* 001 harus memilih lewat *node id* 002 atau *node id* 003.

```
aditya@aditya-Aspire-4740:~$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
169.254.38.32   192.168.30.14   255.255.255.255 UGH   0      0        0 wlan0
169.254.227.154 192.168.50.16   255.255.255.255 UGH   0      0        0 wlan0
169.254.230.210 192.168.50.16   255.255.255.255 UGH   0      0        0 wlan0
192.168.10.0    *               255.255.255.0   U     0      0        0 wlan0
192.168.30.14   192.168.30.14   255.255.255.255 UGH   0      0        0 wlan0
192.168.50.16   192.168.50.16   255.255.255.255 UGH   0      0        0 wlan0
192.168.60.17   192.168.50.16   255.255.255.255 UGH   0      0        0 wlan0
aditya@aditya-Aspire-4740:~$
```

Gambar 8. *Routing table* sebelum terjadi kerusakan *node*

Pada Gambar 8 memperlihatkan belum terjadinya kerusakan *route* yang terbentuk untuk komunikasi dari *node IP* 192.168.10.12 menuju *node IP* 192.168.60.17 melalui *node ip* 192.168.50.16. Dari *node ip* 192.168.50.16 berkomunikasi secara langsung tanpa melalui *node* perantara.

```
aditya@aditya-Aspire-4740:~$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
169.254.38.32   192.168.30.14   255.255.255.255 UGH   0      0        0 wlan0
169.254.227.154 -               255.255.255.255 !H    -1     -        0 -
169.254.230.210 192.168.30.14   255.255.255.255 UGH   0      0        0 wlan0
192.168.10.0    *               255.255.255.0   U     0      0        0 wlan0
192.168.30.14   192.168.30.14   255.255.255.255 UGH   0      0        0 wlan0
192.168.50.16   -               255.255.255.255 !H    -1     -        0 -
192.168.60.17   192.168.30.14   255.255.255.255 UGH   0      0        0 wlan0
aditya@aditya-Aspire-4740:~$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
169.254.38.32   192.168.30.14   255.255.255.255 UGH   0      0        0 wlan0
169.254.230.210 192.168.30.14   255.255.255.255 UGH   0      0        0 wlan0
192.168.10.0    *               255.255.255.0   U     0      0        0 wlan0
192.168.30.14   192.168.30.14   255.255.255.255 UGH   0      0        0 wlan0
192.168.60.17   192.168.30.14   255.255.255.255 UGH   0      0        0 wlan0
aditya@aditya-Aspire-4740:~$
```

Gambar 9. *Routing table* setelah terjadi kerusakan

Pada Gambar 9. memperlihatkan *node IP* 192.168.50.16 dimatikan untuk mensimulasikan terjadinya kerusakan pada *node IP* 192.168.50.16. Pada Gambar 11 memperlihatkan rute baru telah terbentuk untuk komunikasi antara *node IP* 192.168.10.12 dan *node IP* 192.168.60.17 yaitu melalui jalur *node IP* 192.168.30.14.

B. Pengujian Aplikasi Chat

Pengujian aplikasi *Chat* berbasis Phyton digunakan untuk menguji komunikasi pengiriman pesan berbasis teks antar *node*, dari hasil pengujian tersebut, diharapkan bisa diterapkan pada salah satu pemanfaatan MANET seperti pemanfaatan MANET pada daerah bencana.

Pengujian aplikasi *Chat* adalah menguji komunikasi antar *node* dengan saling mengirimkan pesan, komunikasi ini dibutuhkan ketika sistem komunikasi lain mengalami kerusakan akibat terkena dampak bencana alam. Pengujian dilakukan pada *Client* dan *Server*. Pengujian *Client* meliputi pengujian komunikasi antar *Client*, sedangkan untuk *Server* pengujian dilakukan untuk mengetahui komunikasi antar *Client* yang terkoneksi dengan *Server*.

```
aditya@aditya-Aspire-4740:~/Desktop$ python server.py
Server running, listening at ('192.168.10.12', 22222)
('192.168.50.16', 35605) connected
('192.168.50.16', 35605) change username to adit5016
('192.168.40.15', 46433) connected
('192.168.40.15', 46433) change username to adit4015
('192.168.30.14', 49608) connected
('192.168.30.14', 49608) change username to adit3014
('192.168.20.13', 54263) connected
('192.168.20.13', 54263) change username to adit2013
('192.168.60.17', 58191) connected
('192.168.60.17', 58191) change username to adit6017
<adit6017> saya adit6017
<adit5016> saya adit5016
<adit4015> saya adit4015
<adit3014> saya adit3014
<adit2013> saya adit2013
```

Gambar 10. *Server*

Server berfungsi untuk memberikan layanan kepada *Client*, *Server* menerima koneksi dari *Client* yang menghubunginya untuk melakukan komunikasi dengan *Client* lainnya. *Server* berjalan pada IP 192.168.10.12 dengan menggunakan *port* 22222. Setiap ada komunikasi pesan dari *Client* yang baru berkoneksi dengan *Server* akan ditampilkan di *Server* karena semua komunikasi antar *Client* melewati *Server*.

```
pi@raspberrypi ~/Desktop $ sudo python client.py
Please input your name: adit5016
-----------------------------------------------
Command list:
<name> to change username, e.g.: "<name> adit"
<quit> to quit from the program
@username to private chat, e.g.: "@adit hello adit"
-----------------------------------------------
> ('192.168.40.15', 46433) connected
> ('192.168.40.15', 46433) change username to adit4015
> ('192.168.30.14', 49608) connected
> ('192.168.30.14', 49608) change username to adit3014
> ('192.168.20.13', 54263) connected
> ('192.168.20.13', 54263) change username to adit2013
> ('192.168.60.17', 58191) connected
> ('192.168.60.17', 58191) change username to adit6017
> <adit6017> saya adit6017
> saya adit5016
> <adit4015> saya adit4015
> <adit3014> saya adit3014
> <adit2013> saya adit2013
>
```

Gambar 9. *Client*

Untuk melakukan pengiriman pesan masing – masing *Client* harus sudah terkoneksi dengan *Server*, setelah terkoneksi dengan *Server*, langkah pertama adalah mengetikkan nama *user*, kemudian ketikkan pesan untuk mengirimkan pesan ke *Client* lainnya, pesan yang dikirim akan melalui server terlebih dahulu kemudian oleh *Server* dikirimkan ke semua *Client* yang terhubung dengan *Server* seperti pada Gambar 9. Kecuali si pengirim Pada Gambar 10. *Client* menggunakan nama @adit4015 dan @adit2013 guna melakukan pengiriman secara *private* dengan mengetikkan perintah @nama isi pesan.

```
pi@raspberrypi ~/Desktop $ sudo python client.py
Please input your name: adit4015
-----------------------------------------------
Command list:
<name> to change username, e.g.: "<name> adit"
<quit> to quit from the program
@username to private chat, e.g.: "@adit hello adit"
-----------------------------------------------
> private: <adit2013> @adit4015 hello adit
>
```

Gambar 10. *Client*

.

IV. PEMBAHASAN

Dari segi *routing protocol* Babel dapat berjalan dengan baik, hal ini dibuktikan ketika ada *node* yang rusak babel dengan cepat membentuk *route* baru. Sesuai dengan karakteristik MANET yang memiliki topologi yang dinamis, juga menyebabkan jarak poin koneksi yang jauh dan memerlukan node lain untuk berkomunikasi atau *multihop* memungkinkan pengiriman aplikasi *Chat* terhambat. Serta dalam karakteristik MANET yaitu otonomi, sehingga memberatkan *node*. Meskipun begitu pengujian pada kondisi keadaan sesungguhnya akan lebih terlihat nyata sesuai dengan manfaat dari MANET itu sendiri.

V. PENUTUP

Berdasarkan hasil pengujian yang dilakukan, konektivitas antar *node* dalam MANET mampu bekerja dengan baik dalam topologi yang berubah – ubah. Telah berhasil membuktikan *routing protocol* Babel berjalan dengan baik pada MANET, dengan terbentuknya *route* baru setelah terjadi kerusakan *route* akibat matinya salah satu *node*. Berdasarkan hasil pengujian yang dilakukan, sistem komunikasi *Chat* berjalan dengan baik pada MANET.

DAFTAR PUSTAKA

[1] Jonathan, R.A., 2011, Analisa Quality Of Service Pada Mobile Ad-hoc Network Dengan Optimized Link-state Routing Protocol Sebagai Jalur Nirkabel Dan Hierarchical Fair Service Curve Sebagai Metode Penjadwalan Paket. Yogyakarta: Universitas Kristen Dutawacana

[2] Marwan, M. A., 2015, Pengembangan Algoritma Multipath Dominator Pada Mobile Ad-Hoc Network, Jakarta: Univsersitas Gunadarma.K. Elissa, "Title of paper if known," unpublished.

[3] Affandes, M., 2013, Analisa Parameter QoS Routing pada Protokol Jaringan Ad-Hoc Bergerak, Riau: Universitas Sultan Syarif Kasim.

[4] Chroboczek, J., 2011, The Babel Routing Protocol, Request for Comments: 6126, University of Paris, France.

[5] Wijayanto, A., 2017, Analisis Routing Protocol Babel pada MANET, Yogyakarta: Universitas Gadjah Mada