**PROPOSAL PROYEK AKHIR**

HALAMAN SAMPUL  
**IMPLEMENTASI DAN ANALISIS PERFORMA *AVAILABILITY* PROTOKOL *ROUTING* *AD HOC ON-DEMAND DISTANCE VECTOR* PADA *MOBILE ADHOC NETWORK* DENGAN PENGARUH *DATA* *FLOODING* *ATTACK* BERBASIS XBEE**

***IMPLEMENTATION AND AVAILABILITY PERFORMANCE ANALYSIS OF AD HOC ON-DEMAND DISTANCE VECTOR ROUTING PROTOCOL ON MOBILE AD HOC NETWORK UNDER DATA FLOODING ATTACK BASED ON XBEE***

****

**Diajukan oleh :**  
**ADI GUNAWAN**  
14/368530/SV/06940

**PROGRAM SARJANA TERAPAN TEKNOLOGI REKAYASA INTERNET**  
**SEKOLAH VOKASI**  
**UNIVERSITAS GADJAH MADA**  
**YOGYAKARTA**  
**2018**



**USULAN TUGAS AKHIR YANG DIAJUKAN KEPADA**

**PROGRAM SARJANA TERAPAN TEKNOLOGI REKAYASA INTERNET SEKOLAH VOKASI UNIVERSITAS GADJAH MADA**

1. JUDUL TUGAS AKHIR : Implementasi dan Analisis *Availability* Protokol *Routing* *Ad hoc On-demand Distance Vector* pada *Mobile Ad hoc* *Network* dengan Pengaruh *Data* *Flooding* *Attack* Berbasis XBee.

*Implementation and Availability Analysis of Ad hoc On-demand Distance Vector Routing Protokol on Mobile Ad hoc Network Under Data Flooding Attack based on XBee.*

1. PENYUSUN : Adi Gunawan
2. DOSEN PEMBIMBING I
3. Nama Lengkap : Alif Subardono, S.T, M.Eng.
4. NIP : 19740210200212211001
5. TEMPAT PENELITIAN : Ruang Layanan Internet Teknologi Rekayasa Internet UGM.

Yogyakarta, 10 April 2018

# HALAMAN PENGESAHAN

DiSetujui Oleh :

|  |  |
| --- | --- |
| Dosen Pembimbing  Alif Subardono, S.T, M.Eng.  NIP. 197402102002121001 | Penyusun  Adi Gunawan  NIM. 14/368530/SV/06940 |
| Mengetahui,  Ketua Program Studi Teknologi Jaringan  Muhammad Arrofiq, S.T, M.T, Ph.D.  NIP. 197311271999031001 | |

# INTISARI

**USULAN TUGAS AKHIR**

**IMPLEMENTASI DAN ANALISIS PERFORMA *AVAILABILITY* PROTOKOL *ROUTING* *ADHOC ON-DEMAND DISTANCE VECTOR* PADA *MOBILE ADHOC NETWORK* DENGAN PENGARUH SERANGAN *DATA* *FLOODING* *ATTACK* BERBASIS XBEE**

Jaringan *wireless ad hoc* dengan tidak adanya dukungan infrastruktur yang tetap, mengharuskan jaringan ini saling bergantung antar *node-node* yang saling terhubung dalam jaringan tersebut. Jaringan *mobile ad hoc* adalah jaringan yang termasuk kedalam jaringan *wireless ad hoc* dengan karakteristik yang sedikit berbeda, yaitu *node* yang sering berpindah-pindah tempat sehingga mengakibatkan topologi jaringan menjadi sangat dinamis dan susah untuk diprediksi sehingga mengharuskan proses pertukaran informasi pada jaringan *mobile ad hoc* memerlukan sebuah protokol *routing* khusus. XBee melalui protokol ZigBee dapat mengimplmentasikan *Ad hoc On-demand Distance Vector* (AODV) sebagai cara untuk menemukan rute dari sumber ke tujuan dimana protokol tersebut termasuk kedalam kategori reaktif *routing* yang bekerja hanya berdasarkan jika terdapat permintaan dari sebuah *node*. Pada proses penerapan *mobile ad hoc network* itu sendiri masih beberapa celah, salah satunya adalah pada keamanan jaringan *mobile ad hoc* belum sepenuhnya terjamin karena banyak sekali pengaruh yang bisa dilakukan untuk merusak jaringan *mobile ad hoc* tersebut. Maka dari itu pada tugas akhir ini dilakukan implementasi dan analisis kinerja dari protokol *routing* AODV pada protokol ZigBee *mesh networ*k dengan pangaruh *Data* *Flooding Attack*. Parameter yang digunakan untuk mengukur *availability* dan kinerja dari protokol antara lain: *packet delivery ratio* (pdr), *delay*, *throughput*, dan *packet loss*.

Kata Kunci : *Wireless, Ad hoc*, *Mobile Ad hoc Network* (MANET), AODV, Data *Flooding.*

# DAFTAR ISI

[HALAMAN SAMPUL i](#_Toc503989741)

[HALAMAN PENGESAHAN ii](#_Toc503989742)

[INTISARI iii](#_Toc503989743)

[DAFTAR ISI iv](#_Toc503989744)

[DAFTAR GAMBAR v](#_Toc503989745)

[DAFTAR TABEL vi](#_Toc503989746)

[BAB I PENDAHULUAN 1](#_Toc503989747)

[1.1 Latar Belakang 1](#_Toc503989748)

[1.2 Rumusan Masalah 2](#_Toc503989749)

[1.3 Batasan Masalah 2](#_Toc503989750)

[BAB II TUJUAN PROYEK AKHIR 4](#_Toc503989751)

[BAB III TINJAUAN PUSTAKA 5](#_Toc503989752)

[BAB IV LANDASAN TEORI DAN HIPOTESIS 13](#_Toc503989753)

[4.1 Landasan Teori 13](#_Toc503989754)

[4.1.1 Jaringan Nirkabel atau *Wireless Network* 13](#_Toc503989755)

[4.1.2 Jaringan *Mobile Ad hoc Network* (MANET) 17](#_Toc503989756)

[4.1.3 *Routing* pada jaringan *Mobile Ad hoc Network* (MANET) 21](#_Toc503989757)

[4.1.4 Mikrokontroler dan Arduino 31](#_Toc503989758)

[4.1.5 Modul XBee 34](#_Toc503989759)

[4.1.6 Kalilinux 38](#_Toc503989760)

[4.1.7 Parameter Kinerja 38](#_Toc503989761)

[4.2 Hipotesis 41](#_Toc503989762)

[BAB V METODE PENELITIAN 42](#_Toc503989763)

[5.1 Alat dan Bahan 42](#_Toc503989764)

[5.2 Prosedur Penelitian 43](#_Toc503989765)

[5.3 Analisis Hasil 47](#_Toc503989766)

[5.4 Jadwal Penelitian 47](#_Toc503989767)

[DAFTAR PUSTAKA 48](#_Toc503989768)

# DAFTAR GAMBAR

[Gambar 4.1 *Mobile Ad hoc Network* (MANETs) 17](#_Toc511035013)

[Gambar 4.2 Tantangan pada Jaringan MANETs berdasarkan layer 21](#_Toc511035014)

[Gambar 4.3 Format Pesan *Route Request* 25](#_Toc511035015)

[Gambar 4.4 Format Pesan *Route Reply* 25](#_Toc511035016)

[Gambar 4.5 Format Pesan *Route Error* 26](#_Toc511035017)

[Gambar 4.6 Format Pesan *Route Reply Acknowledgement* 26](#_Toc511035018)

[Gambar 4.7 Ilustrasi *Route Discovery* 31](#_Toc511035019)

[Gambar 4.8 Ilustrasi *Route Maintence* 31](#_Toc511035020)

[Gambar 4.9 Arduino Uno R3 32](#_Toc511035021)

[Gambar 4.10 Tampilan Arduino IDE 33](#_Toc511035022)

[Gambar 4.11 Topologi Jaringan XBee 37](#_Toc511035023)

[Gambar 5.1 Rancangan Topologi Jaringan MANET Skenario 1 43](#_Toc511035024)

[Gambar 5.2 Rancangan Topologi Jaringan MANET Skenario 2 43](#_Toc511035025)

[Gambar 5.3 Rancangan Topologi Jaringan MANET Skenario 3 44](#_Toc511035026)

[Gambar 5.4 Gambaran *Node* 45](#_Toc511035027)

[Gambar 5.5 *Flowchart* Penelitian 46](#_Toc511035028)

DAFTAR TABE

[Tabel 3.1 Ringkasan Sumber Jurnal Penelitian 11](#_Toc511035085)

[Tabel 4.1 Standarisasi Delay versi *TIPHON* 39](#_Toc511035086)

[Tabel 4.2 Data *Througput* XBee 40](#_Toc511035087)

[Tabel 4.3 Standarisasi *Packet Loss* versi *TIPHON* 41](#_Toc511035088)

[Tabel 5.1 Ringkasan data dan waktu penyerangan 45](#_Toc511035089)

[Tabel 5.2 Jadwal Penelitian 47](#_Toc511035090)

# BAB I

**PENDAHULUAN**

## Latar Belakang

Teknologi jaringan w*ireless* terus berkembang setiap harinya yang diakibatkan oleh aktivitas seseorang dengan kebutuhan mobilitas yang tinggi. Jaringan *wireless* dapat dijumpai dimana saja seperti telepon yang biasa digunakan untuk berkomunikasi dan *Access Point*. Pada dasarnya jaringan *wireless* terbagi menjadi 2 yaitu jaringan *wireless* dengan insfrastruktur dan jaringan *wireless* tanpa infrastruktur yang kemudian dikenal dengan istilah *wireless* *ad hoc*. Dalam perkembangan jaringan *wireless ad hoc* itu sendiri kemudian memunculkan berbagai macam bentuk model jaringan baru seperti *Mobile Ad hoc Networks* (MANET) dan *Vehicular Ad hoc Networks* (VANETs).

Jaringan *Mobile Ad hoc* adalah jaringan yang tidak memiliki *gateway* atau jalur keluar yang tetap dengan demikian setiap perangkat yang saling terhubung akan saling bertukar informasi satu sama lain menggunakan bantuan protokol *routing* khusus untuk jaringan MANETs seperti *Ad hoc On Demand Distance Vector Routing* (AODV), *Dynamic Source Routing Protokol* (DSR), *Destination Sequence Distance Vector Routing* (DSDV), *Optimized Link State Routing* (OLSR) dan protokol *routing* yang lain. Ketika berbagi informasi antar perangkat pada *Mobile Ad hoc Network* terdapat beberapa hal yang harus diperhatikan terkait dengan keamanan *Mobile Ad hoc Network*. Hal-hal yang harus diperhatikan adalah *Authenticity* yaitu keaslian dari sebuah pesan yang dikirimkan, *Integrity* yaitu pesan yang dikirim atau diinformasikan kepada perangkat lain tidak dimodifikasi ketika dijalan atau dalam proses pengiriman, *Confidentiality* yaitu pesan harus bersifat rahasia karena tidak boleh ada orang yang tahu selain pengirim dan penerima pesan tersebut, *Availability* berhubungan dengan ketersediaan jaringan apakah jaringan tersebut dapat mengirimkan pesan atau tidak, dan *Non-Repudiation* (Thilak 2016).

Dari lima (5) bagian yang perlu diamankan yang paling penting dan sangat mendasar adalah tingkat ketersediaan jaringan itu sendiri atau *availability*, karena adanya ketersediaan jaringan dapat memfasilitasi aktivitas pada bagian lain seperti *Integrity* dan *Authenticity.* Maka dari itu pada tugas akhir ini akan dilakukan implementasi dan analisis kinerja protokol *routing Ad hoc On-demand Distance Vector* (AODV) yang dipengaruhi oleh serangan *data flooding attack* untuk menguji tingkat ketersediaan pada protokol *routing* yang digunakan tersebut.

## Rumusan Masalah

Beberapa rumusan masalah yang mengacu pada tujuan penulisan tugas akhir ini adalah sebagai berikut :

1. Apakah protokol *routing* *Ad hoc On-demand Distance Vector* (AODV) dapat diimplementasikan pada jaringan *Mobile Ad hoc* menggunakan modul komunikasi XBee ?
2. Bagaimana penerapan protokol *routing* *Ad hoc On-demand Distance Vector* (AODV) pada jaringan *Mobile Ad hoc* tersebut ?
3. Bagaimana tingkat *availability* kinerja protokol *routing* *Ad hoc On-demand Distance Vector* (AODV) pada saat dilakukan *flooding* data ?

## Batasan Masalah

1. Implementasi dilakukan hanya untuk serangan data *flooding attack* pada *node-node* yang telah dibangun dengan protokol *routing* *Ad hoc On-demand Distance Vector* (AODV).
2. Protokol *routing Ad hoc On-demand Distance Vector* (AODV) sudah terbangun ketika menggunakan Protokol ZigBee dan bisa dilihat pada tabel *routing*.
3. *Flooding data* dilakukan dengan bantuah aplikasi bernama hping3 pada Sistem Operasi Kali Linux.
4. Hal-hal lain yang menyangkut interferensi dan keamanan seperti enkripsi dan yang lain tidak diperhitungkan.
5. Jarak antar *node* dalam hal ini XBee diatur sesuai dengan jarak yang sesuai berdasarkan kemampuan XBee.
6. Serangan dilakukan pada *node* yang bertugas sebagai koordinator, *router* maupun *end device* dan kemudian dilakukan analisa terhadap *availability* jaringan berdasarkan QoS.
7. *Node* yang digunakan berjumlah 5 sampai 6 *node*, dengan spesifikasi 1 *node* sebagai *coordinator*, 2 *node* sebagai *router* dan sisanya digunakan sebagai *end* *device*.
8. *Node* yang berpartisipasi kedalam proses pembuatan rute pada Protokol ZigBee adalah *Coordinator* dan *Router* dalam mode API, sedangkan *End Device* tidak berpartisipasi dalam proses pembuatan rute melainkan hanya menerima dan mengirim data. Oleh karena itu, *node* yang berperan sebagai *Coordinator* dan *Router* di atur kedalam mode API sedangkan *End Device* lebih fleksibel bisa menggunakan API maupun AT (Transparan).

### 1

# BAB II

**TUJUAN PROYEK AKHIR**

### **3**

### Bagi Mahasiswa D4 Teknologi Rekayasa Internet :

### Menambah wawasan mahasiswa mengenai penerapan protokol MQTT dan CoAP dalam jaringan IoT.

### Mampu mengetahui protokol mana yang lebih dibutuhkan dalam kondisi jaringan tertentu

Bagi Program Studi D4 Teknologi Jaringan :

* Penelitian ini dapat digunakan sebagai bahan ajar tambahan dalam mata kuliah Praktikum Rancangan dan Aplikasi IoT

# BAB III

**TINJAUAN PUSTAKA**

### 3

Paradigma *wireless networking* atau jaringan nirkabel dapat dikategorikan menjadi dua golongan yaitu *wireless Ad hoc* dan *cellular networking* atau jaringan yang terinfrastruktur. Adanya infrastruktur menjadi perbedaan utama antara dua golongan jaringan *wireless* tersebut. Pada *Ad hoc networking paradigm* atau paradigma jaringan *Ad hoc* tidak ada infastruktur yang tetap dan paket data terkirim ke tujuan melewati jaringan *wireless* dengan beberapa lompatan (*multihop*), dan juga *node* seringkali tidak hanya berfungsi sebagai host melainkan juga berfungsi sebagai *router* dengan memberikan informasi keadaan lalu lintas pada *node* yang lain (*neighboor node*). Topologi pada jaringan *Ad hoc* dapat berubah setiap saat karena *node* selalu bergerak (*mobile*) atau keadaan lain yang mungkin terjadi adalah *node*-*node* tersebut gagal untuk membangun sebuah topologinya. Pada jaringan yang terinsfrastruktur atau *cellular networking* *node* dapat mencapai *access point* atau *gatewaynya* hanya dalam satu lompatan jaringan *wireless*, gambar jaringan wireless *Ad hoc* dan *cellular networking* (Çayırcı and Rong 2009).

Fungsi *node* pada jaringan *Ad hoc* tidak hanya sebagai host, fungsi yang lain adalah sebagai *router*, ketika berfungsi sebagai *router* maka *node* tersebut harus mencari dan memelihara informasi dari jalur yang digunakan untuk sampai pada *node* yang lain. Dalam melakukan tugas tersebut *node* tidak bisa menjalankan pekerjaannya seorang diri melainkan dibutuhkan sebuah protokol *routing* pada jaringan *Ad hoc*. Secara garis besar protokol *routing* pada *Mobile Ad hoc Network* (MANET) dapat dikategorikan menjadi 2 yaitu *Table-driven* atau *Proactive* dan *On-demand* atau *Reactive*. Protokol *routing* yang bersifat proaktif bekerja dengan cara memelihara informasi rute setiap saat kepada *node* tetangganya. Contoh yang termasuk dalam golongan *routing* proaktif ini adalah *Destination Sequence Distance Vector* (DSDV). Kemudian protokol *routing* yang bersifat reaktif, *node* hanya melakukan pencarian rute ketika ada *node* sumber akan melakukan pengiriman paket data ke *node* tujuan, maka kemudian *node* mencari jalur menuju *node* tujuan tersebut dan pembaruan informasi dilakukan secara berkala tidak setiap saat. Contoh protokol *routing* yang termasuk dalam kategori ini adalah *Ad hoc On-demand Distance Vector* (AODV) dan *Dynamic Source Routing* (DSR) (Deng, Li and Agrawal 2002).

Dalam penelitian ini, ada beberapa hal penting yang dapat dikaji dalam manet maupun protokol Zigbee pada jaringan nirkabel yaitu dengan cara mengukur *throughput*, *delay* *packet delivery ratio* (pdr) dan *packet loss* sebagai parameter tolak ukur performa jaringan manet tersebut. Penelitian pertama yang digunakan sebagai acuan penelitian yang berkaitan dengan protokol *routing Ad hoc On-demand Distance Vector* (AODV) adalah pada penelitian simulasi menggunakan perangkat lunak NS-3 tentang kinerja protokol *routing* AODV yang ditimpa dengan *flooding attack* (Bandyopadhyay and Vuppala 2011)*.* Penelitian tersebut membagi *flooding attack* kedalam dua tipe serangan yaitu *Route Request* (RREQs) *Flooding Attack* dan *Data Flooding Attack*. *Flooding Attack* sendiri merupakan jenis serangan dari *Denial of Service Attack* dimana *node* jahat melakukan serangan dengan cara membanjiri *node* yang lain dengan paket data dalam jumlah yang sangat banyak. Dua jenis serangan *flooding attack* yang akan digunakan juga memiliki mekanisme yang berbeda saat melakukan serangan, RREQs *Flooding Attack* melakukan serangan dengan cara mengambil keuntungan ketika proses pencarian rute pada protokol *routing* yang bersifat reaktif seperti DSR dan AODV. Dalam serangan ini *node* jahat melakukan pesan siaran dijaringan dimana tidak terdapat alamat valid yang dituju, sehingga ketika tidak ada tujuan yang valid jaringan menjadi penuh dengan paket RREQs dari *node* jahat, sedangkan untuk serangan *Data Flooding Attack* pembajiran data dilakukan ketika rute antar *node* sudah terbangun dan siap digunakan. *node* jahat mengalirkan data dalam jumlah yang besar dan tidak berguna sama sekali didalam jaringan sehingga membuat jalur menjadi tersumbat dan mengurangi *bandwidth* jaringan yang ada. Pada simulasi ini digunakan 15 *node* yang ditempatkan secara acak dalam area berukuran (500x500)m dengan waktu simulasi untuk melakukan serangan adalah selama 60 detik. Parameter yang digunakan untuk pengukuran antara lain : *routing overhead,* presentase dari *data packet loss*, serta *bandwidth usagenya.* Simulasi yang dilakukan menghasilkan data seperti penjelasan berikut, ketika dilakukan RREQs *flooding attack* terdapat dua parameter yang digunakan yaitu *routing overhead* dan *data packet loss* yang sama-sama mengalami peningkatan dan membuat jaringan menjadi lemah, hal yang sama ditunjukkan pada jaringan *Ad hoc* yang dilakukan serangan dengan *data flooding attack,* dilihat dari parameter *routing overhead, bandwidth usage* dan *data packet loss* mengalami peningkatan yang signifikan.

Simulasi terhadap *Mobile Ad hoc Network* (MANET) yang ditimpa dengan berbagai macam serangan juga dilakukan peneltian dengan beberapa variabel yang dianalisis seperti dampak dari ukuran jaringan, dampak dari mobilitas *node*, dampak *traffic load* (HoudaMoudni, et al. 2016)*.* Pada penelitian tersebut dititik beratkan pada kinerja protokol *routing* AODV pada jaringan MANET dan menggunakan perangkat lunak NS-2. Simulasi sendiri menggunakan beberapa parameter untuk menentukan batasan masalah terhadap penelitian tersebut, parameter-parameter yang digunakan untuk simulasi antara lain luas area berukuran (500 x 500)m, ukuran paket sebesar 512 bytes dan durasi atau lama waktu melakukan simulasi terhadap sebuah skenario adalah 200 detik. Kinerja protokol *routing* AODV ketika ditimpa oleh serangan *balckhole attack*, *flooding attack*, dan *rushing attack* diamati melalui parameter pada jaringan seperti *packet delivery ratio* (pdr), *end-to-end delay*, *throughpu*t, dan *packet loss* dengan hasil simulasi yang menunjukkan hampir sebagian variabel yang diamati mengalami penurunan kinerja ketika ditimpa oleh serangan-serangan tersebut.

Variabel yang pertama yaitu pengaruh ukuran jaringan yang ditandai dengan banyaknya *node* yang berhubungan satu sama lain ketika ditimpa. *Packet delivery ratio* menunjukkan penurunan yang signifikan saat ditimpa oleh *black hole attack* sampai dibawah 25%, berbeda halnya saat diserang oleh *flooding attack* dan *rushing attack* yang masih memiliki packet delivery ratio diatas 75 %, selanjutnya pada pengamatan dengan parameter *end-to-end delay* menunjukkan penurunan kinerja ketika ditimpa oleh tiga serangan tersebut, sedangkan *throughput* mengalami penurunan secara signifikan dialami oleh jaringan ketika ditimpa dengan *black hole attack*. Variabel yang kedua pada pengamatan yaitu pengaruh mobilitas atau pergerakan *node* terhadap kinerja protokol. Pada pengamatan ini *node* yang digunakan sebanyak 5, dengan kecepatan diatur sampai dengan 30 m/s dan hasil menunjukkan ketika ditimpa oleh *black hole attack*, kinerja protokol *routing* mengalami penurunan yang sangat signifikan yaitu dibawah angka 20% pada parameter pdr dan *throughput*. Berbeda dengan *flooding attack* dan *rushing attack* yang tidak terlalu menurunkan kinerja protokol *routing* AODV tersebut. Sedangkan pada parameter *end-to-end delay* serangan *black hole* juga meningkatkan waktu secara signifikan dibandingan kedua serangan lainnya.

Sebuah penelitian juga dilakukan pada jaringan *Ad hoc* yang ditimpa dengan DoS *attack*. Penelitian dilakukan dengan cara simulasi menggunakan perangkat lunak *open source* NS-3 dengan *platform* sistem operasi Ubuntu. Pada simulasi tersebut digunakan 10 *node* dengan protokol *routing* AODV, pada area (500x500)m, dan waktu simulasi dalam satu kali tempaan adalah 10 detik. Selanjutnya serangan yang dilakukan dalam jaringan pada penelitian ini adalah dua jenis serangan yang termasuk dalam kategori serangan DoS yaitu *hello floods attack* dan *flooding attack.* Serangan *flooding attack* dilakukan dengan cara mengirim paket *route request* (rreq) secara terus menerus dalam jaringan yang mengakibatkan kemacetan sementara untuk *hello floods attack* dilakukan dengan cara pembanjiran paket data pesan hello kepada para tetangganya untuk membuat *node* tetangga menjadi kehilangan daya sehingga tidak memberikan pesan balasan terhadap pesan rreq yang asli. Dalam pengambilan data penelitian ini menggunakan lima skenario yaitu tiga skenario untuk *node* dengan pergerakan (*mobile nodes*) dan dua skenario untuk *node* tanpa pergerakan tetap (*fixed nodes*). Hasil dari penelitian terhadap lima skenario tersebut adalah seperti berikut : simulasi yang pertama yaitu *node* bergerak dengan tidak ditimpa oleh serangan apapun didalamnya yang menunjukkan presentasi plr 0% dan rtt sebesar 4ms, akan tetapi ketika *node-node* dilakukan perubahan posisi atau digerakkan (*mobile),* parameter plr berubah menjadi 65% dan rtt mendekati 1000ms. Skenario yang kedua dilakukan penyerangan pada *node* tanpa pergerakan dengan *hello floods attack*, hasilnya adalah kehadiran satu *node* jahat dengan serangan tersebut membuat *round trip time* (rtt) dari sebuah *node* normal menjadi 120ms. Skenario yang ketiga yaitu dilakukan penyerangan *flooding attack* pada *node* tanpa pergerakan dengan hasil kehadiran satu atau dua *node* dengan serangan tersebut membuat *node* normal memiliki rtt 10 kali lipat lebih besar dari pada tidak dilakukan serangan, sedangkan untuk plrnya naik menjadi 40%. Kemudian untuk skenario yang keempat dilakukan serangan *hello flood attack* pada jaringan dengan *node* dilakukan pergerakan (*mobile nodes*) yang menunjukkan hasil satu *node* jahat yang melakukan serangan mengakibatkan seluruh *node* kehilangan paketnya, dan untuk skenario yang terakhir adalah penyerangan pada *node* dengan pergerakan oleh serangan *flooding attack*. Dampak dari serangan tersebut membuat plr menjadi 80% (Lotfy and Azer 2013).

Beberapa penelitian diatas baru dilakukan secara simulasi artinya belum sepenuhnya yang ada pada simulasi tersebut bisa dilakukan secara nyata dengan menggunakan sebuah alat tertentu. Penelitian secara nyata kini mulai dilakukan, salah satunya adalah pada jurnal yang melakukan penelitian tentang perbandingan *Quality of Service* (QoS) dijaringan *Wireless Sensor Network* (WSN) menggunakan XBee pada topologi *star* dan topologi *multihop* (Pramono, et al. 2017)*.* *Quality of Service* (QoS yang digunakan untuk menjadi acuan dalam penelitian tersebut berupa *delay*, *throughput* dan *packet loss*. Peneltian dilakukan menggunakan sebuah modul komunikasi nirkabel berupa XBee dengan tipe S2C dengan setiap pengambilan data QoSnya, perangkat Xbee tersebut ditempatkan pada jarak yang berbeda-beda. Sehingga selain membandingkan topologi *star* dan *Multihop* penelitian tersebut juga membandingkan dari faktor jarak dan jarak yang digunakan adalah 10m, 20m, dan 30m. Dari penelitian tersebut didapatkan kesimpulan bahwa 2 topologi yang dibandingkan didapatkan nilai *throughput* tertinggi pada topologi *star* dengan nilai 4109 bps dan 3019 untuk topologi multihop. Pada parameter *packet loss* didapatkan paket yang hilang sebesar 4% pada topologi *star* dan 5,5% pada topologi *multihop*, sedangkan untuk parameter *delay* didapatkan nilai 296 ms pada topologi *star* dan 554 ms pada topologi *multihop* serta jarak terjauh yang dapat tercapai pada penelitian tersebut adalah 660 m dengan penerapan *Line-of-Sight* (LOS). Penelitian yang hamper sama juga dilakukan yaitu tantang analisis performa jaringan ZigBee pada *Wireless Sensor Network* (WSN) dengan parameter *Quality of Service* (Qos) (Fitriawan, et al. 2017). Dalam penelitian ini dilakukan uji performa QoS berupa *throughput*, *delay* dan *packet loss* dengan model topologi menerapkan *Line-of-Sight* (LOS) dan *Non Line-of Sight* (NLOS) dan percobaan dilakukan dengan kecepatan *baudrate* yang berbeda-beda. Modul komunikasi yang digunakan pada penelitian ini adalah modul RF XBee S2 dan menggunakan perangkat lunak docklight v 2.0 untuk menganalisan *Quality of Service* (Qos) tersebut. Hasil pada penelitian tersebut membuktikan bahwa setiap kecepatan *baudrate* yang diterapkan mempengaruhi terhadap parameter uji *Quality Os Service.* Pada *throughput* semakin bertambahnya kecepetan *baudrate* maka akan meningkatkan nilai *throughput*, dan pada *delay* semakin rendah kecepatan *baudrate* akan membuat nilai *delay* semakin besar.

Berdasarkan penelitian pada uraian sebelumnya dapat diketahui bahwa pada jaringan *Ad hoc* yang ditimpa berbagai macam jenis serangan dari *Denial of Service* (DoS) Attack pada protokol *routing* yang tidak diberikan protokol keamanan seperti ARAN atau SAODV akan mengkibatkan kemacetan pada jalur, yang jika dibiarkan terus menerus akan membuat jaringan akan mati total dan tidak bisa digunakan untuk komunikasi antar *node*. pendekatan yang mungkin bisa dilakukan untuk melakukan implentasi jaringan *ad hoc* network kedalam dunia nyata adalah menggunakan modul komunikasi nirkabel XBee seperti pada 2 paparan jurnal terkahir. Dalam penelitian yang diuraikan tersebut jaringan *Ad hoc* diimplementasikan hanya dalam sebatas simulasi dan baru beberapa penelitian yang melakukan implementasi menggunakan alat yang sesungguhnya dan sedikit penelitian yang berusaha untuk merealisasikan *mobile ad hoc network* (manet) secara langsung, oleh karena itu berdasar pada penelitian yang sudah ada, penelitian ini akan mengimplementasikan jaringan *Ad hoc* menggunakan alat berupa modul komunikasi nirkabel XBee dengan protokol ZigBee dimana didalam protokol ZigBee sudah menerapkan proses *routing* *Ad hoc On demand Distance Vector* (AODV). Setelah topologi terbangun akan dilakukan ujicoba ketersediaan jaringan dengan pengaruh serangan *data* *flooding attack* menggunakan salah satu *tools* pada Kali Linux yaitu hping3. Adapun ringkasan berdasarkan uraian penelitian diatas dapat dilihat pada Tabel 3.1 dibawah ini :

Tabel 3.1 Ringkasan Sumber Jurnal Penelitian

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Judul Penelitian** | **Pokok Bahasan** | **Tipe Penelitian** | **Skenario** | **Penulis** |
| 1 | *A Simulation Analysis of Flooding Attack in MANET using NS-3* | Melakukan serangan pada jaringan *Adhoc* dengan jenis serangan *RREQ Flooding Attack* dan *Data Flooding Attack* | Simulasi | NS-3, AODV tanpa keamanan, 15 *node* | Alokparna Bandyopadhyay, Satyanarayana Vuppala |
| 2 | *Performance Analysis of AODV Routing Protocol in MANET* *under the influence of Routing Attack* | Melakukan serangan pada jaringan *Mobile Adhoc Network* (MANET) dengan jenis serangan *black hole, flooding* dan *rushing attack* | Simulasi | NS-2, AODV, tanpa keamanan, 20 s/d 80 *node* | HoudaMoudni, Mohamed Er-rouidi, HichamMouncif, Benachir El Hadadi |
| 3 | *Performance Evaluation of AODV Under Dos Attack* | Melakukan serangan pada jaringan *Adhoc* dengan jenis serangan *Hello Floods Attack* dan *Flooding Attack* | Simulasi | NS-3, AODV tanpa keamanan, 10 *node* | Poussy A. Lotfy, Marianne A. Azer |
| 4 | *Comparative Analysis of Star Topology and Multihop Outdoor Propagation Basen on Quality of Service (*QoS*) of Wireless Sensor Network (*WSN*)* | Melakukan analisis terhadap nilai QoS berupa *delay*, *throughput* dan *packet* *loss* pada jaringan WSN dengan topologi Star dan *Multihop* | Implementasi | XBee S2C, Sensor MQ-9, Star dan Multihop (Mesh) | Subuh Pramono, Arisa Olivia Putri, Endro Warsito, Budi Basuki |
| 5 | ZigBee *Based Wireless Sensor Network and Performance Analysis in Various Environment* | Melakukan uji performa jaringan Zigbee dengan dua keadaan yaitu LOS dan NLOS | Implementasi | XBee S2, ZigBee Protocol, AODV, *Line-of-Sight* (LOS), *Non Line-of-Sight* (NLOS) | Helmy Fitriawan, Misfa Susanto, Ahmad Surya Arifin, Danny Mausa, Agus Trisanto |

# BAB IV

**LANDASAN TEORI DAN HIPOTESIS**

### 3

Pada bab landasan teori dan hipotesis dijelaskan berbagai macam teori mengenai jaringan *wireless* secara umum beserta pembagiannya, khususnya untuk teori yang mendalam tentang *Mobile Ad hoc Networks* (MANETs). Selain itu juga dijelaskan teori pendukung mengenai komponen-komponen yang digunakan dalam proses implementasi jaringan *Ad hoc* menggunakan XBee ini. Penjelasan teori dimulai dari tinjauan singkat mengenai bagian-bagian penting dan mendasar dari sistem jaringan *wireless* yang merupakan dasar dalam pembuatan jaringan *Mobile Ad hoc Network* (MANET) itu sendiri. Pustaka yang dipilih disesuaikan berdasarkan keterkaitan isu dari pustaka dengan ruang lingkup topik yang dibahas secara lengkap dan menyeluruh sehingga dapat mendukung pemecahan masalah dalam penelitian ini.

Pada sub bab berikutnya dijelaskan hipotesisyang merupakan jawaban sementara terhadap permasalahan yang sedang diteliti dimana kebenarannya harus diuji secara empiris. Hipotesis dikatakan sementara karena jawaban yang diberikan baru didasarkan pada teori yang relevan dan belum didasarkan pada fakta-fakta yang empiris yang diperoleh melalui pengumpulan data (Hasibuan 2007).

#### Landasan Teori

##### Jaringan Nirkabel atau *Wireless Network*

*Wireless Network* atau Jaringan Nirkabel adalah jaringan komputer yang menggunakan frekuensi radio sebagai perantara komunikasinya. Setiap *node* dalam jaringan menyiarkan informasi yang dapat diterima oleh semua *node* dalam jangkauan transmisinya. Sistem komunikasi radio nirkabel pertama kali ditemukan oleh Guglielmo Marconi pada tahun 1897. Pada tahun 1901, beliau sukses mendemonstrasikan sistem telegraph nirkabelnya dengan mentransmisikan sinyal radio menyeberangi Samudra Atlantik dari Inggris ke Amerika, menjangkau lebih dari 1.700 mil (Murthy and Manoj 2004). Banyak jenis perangkat nirkabel yang tersedia saat ini memjadikan perbedaan sering ditemukan di lapisan fisik sistem ini. Perangkat nirkabel dan jaringan yang lain seperti jaringan dengan kabel menunjukkan karakteristik yang berbeda, khususnya dalam hal :

* Interferensi yang tinggi menghasilkan tingkat keandalan yang lebih rendah.
* Ketersediaan *bandwidth* dan tingkat transmisi yang jauh lebih rendah membuat kecepatan menjadi lebih lambar dari pada jaringan kabel, serta menyebabkan ­*Qos* atau kualitas layanan menjadi turun yang ditandai dengan adanya *jitter, delay,* dan pembangunan koneksi yang lebih lama.
* Kondisi jaringan yang bervariasi dengan tingkat kehilangan data lebih tinggi akibat gangguan, pergerakan perangkat yang menyebabkan terputusnya koneksi, dan perubahan *channel.*
* Cakupan layanan terbatas karena keterbatasan perangkat, jarak, dan kondisi jaringan, implementasi layanan untuk perangkat nirkabel dan jaringan menghadapi banyak kendala dan lebih menantang dibandingkan dengan jaringan kabel.
* Sumber daya yang terbatas seperti media untuk berbagi, frekuensi yang terbatas dengan peraturan yang ketat.
* Keamanan lemah karena antarmuka radio dapat diakses oleh semua orang yang membuat keamanan jaringan sulit diterapkan karena penyerang dapat berinteraksi dengan lebih mudah.

1. Tipe Jaringan Nirkabel / *Wireless*

Terdapat beberapa jenis jaringan nirkabel yang ada, (Murthy and Manoj 2004) membagi menjadi 4 kategori jaringan nirkabel dengan berbagai cara seperti yang tercantum dalam sub bagian berikut.

1. Berdasarkan Arsitektur Jaringan

Jaringan nirkabel dapat dibagi menjadi dua kategori berdasarkan bagaimana jaringan dibangun dan arsitektur jaringan yaitu :

1. Jaringan Berbasis Infrastruktur

Jaringan dengan infrastruktur terdiri dari jaringan kabel tetap, biasanya layanan jaringan yang dikirimkan melalui infrastruktur yang telah dikonfigurasi sebelumnya. Misalnya, jaringan seluler adalah jaringan berbasis infrastruktur yang dibangun dari *switch* backbone PSTN, MSC, *base station*, dan *host mobile*. Setiap *node* memiliki tanggung jawab khusus dalam jaringan, dan pembentukan koneksi mengikuti urutan sinyal yang ketat di antara *node* (Lin and Chlamtac 2000).

1. Jaringan Tanpa Infrastruktur (*Ad hoc*)

Dalam hal ini sebuah jaringan terbentuk secara dinamis melalui kerja sama sekumpulan *node* independen. Tidak ada pengaturan awal mengenai peran spesifik setiap *node* yang harus diasumsikan. Sebagai gantinya, setiap *node* membuat keputusannya secara independen, berdasarkan situasi jaringan, tanpa menggunakan infrastruktur jaringan yang sudah ada sebelumnya. Sebagai contoh, dua PC yang dilengkapi dengan kartu adaptor nirkabel dapat mengatur jaringan independen setiap kali berada dalam jangkauan satu sama lain. Dalam jaringan *Ad hoc mobile*, *node* diharapkan dapat berperilaku sebagai *router* dan mengambil bagian dalam penemuan dan pemeliharaan rute ke *node* lain.

1. Berdasarkan Cakupan Area Komunikasi

Seperti jaringan kabel, jaringan nirkabel dapat diklasifikasikan ke dalam jenis yang berbeda berdasarkan jarak data yang ditransmisikan :

1. *Wireless Wide Area Networks* (Wireless WANs)

Wireless WAN adalah jaringan nirkabel berbasis infrastruktur yang mengandalkan infrastruktur jaringan seperti MSC dan *base station* untuk memungkinkan pengguna ponsel membuat koneksi nirkabel melalui jaringan public . Koneksi ini dapat dilakukan di wilayah geografis yang luas, di seluruh kota atau bahkan negara, melalui penggunaan beberapa antena atau sistem satelit yang dikelola oleh penyedia layanan nirkabel. Jaringan seluler (seperti jaringan GSM atau jaringan CDMA) dan jaringan satelit adalah contoh jaringan WAN nirkabel yang baik.

1. *Wireless Metropolitan Area Networks* (Wireless MANs)

Jaringan Wireless MAN kadang disebut sebagai *fixed wireless* yang juga merupakan jaringan berbasis infrastruktur untuk memungkinkan pengguna membangun koneksi nirkabel *broadband* di antara beberapa lokasi di wilayah metropolitan, misalnya di antara beberapa gedung perkantoran di kota atau di kampus universitas, tanpa biaya pemasangan kabel serat atau tembaga.

1. *Wireless Local Area Networks* (Wireless LANs)

Jaringan lokal nirkabel memungkinkan pengguna membuat koneksi nirkabel di dalam area lokal, biasanya di dalam gedung perusahaan atau kampus, atau di tempat umum, seperti bandara, biasanya berada dalam jarak 100 m. WLAN menyediakan sistem komunikasi data yang fleksibel yang dapat digunakan di kantor atau ruang lain dimana pemasangan kabel yang luas akan menjadi penghalang, atau untuk melengkapi LAN yang ada sehingga pengguna dapat bekerja di lokasi yang berbeda dalam bangunan pada waktu yang berbeda (Chlamtac and El-Zarki 1994).

1. *Wireless Personal Area Networks* (Wireless PANs)

Teknologi *Wireless* PAN memungkinkan pengguna untuk membangun jaringan *ad hoc*, komunikasi nirkabel antara perangkat nirkabel pribadi seperti PDA, telepon seluler, atau laptop yang digunakan di dalam ruang operasi pribadi, biasanya mencapai jarak 10 meter. Dua teknologi Wireless PAN adalah *Bluetooth* dan cahaya inframerah. *Bluetooth* adalah teknologi penggantian kabel yang menggunakan gelombang radio untuk mentransmisikan data ke jarak hingga 9-10 m, sedangkan inframerah dapat menghubungkan perangkat dalam jarak 1 m. PAN nirkabel mendapatkan momentum karena kompleksitasnya rendah, konsumsi daya rendah, dan interoperabilitas dengan jaringan 802.11.

1. Berdasarkan Teknologi Akses

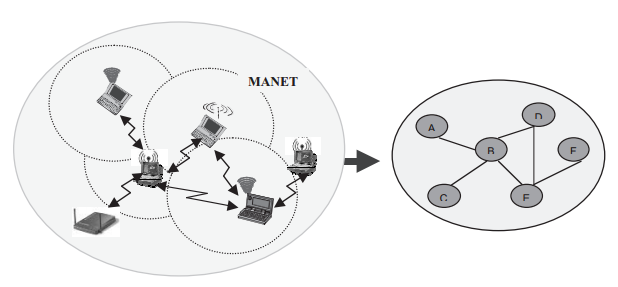
Bergantung pada penggunaan standar, frekuensi, dan spektrum tertentu, jaringan nirkabel dapat dikategorikan berdasarkan teknologi akses yang digunakan, diantaranya :

1. Jaringan GSM
2. Jaringan TDMA
3. Jaringan CDMA
4. Jaringan Satelit
5. Jaringan Wi-fi (802.11)
6. Jaringan Hiperlan2
7. Jaringan *Bluetooth*
8. Jaringan Inframerah
9. Berdasarkan Pengaplikasian Jaringan

Jaringan nirkabel juga dapat dikategorikan berdasarkan penggunaan dan aplikasi spesifik yang mereka dukung. Sebagai contoh Jaringan *Enterprise*, Jaringan Rumah, *Tactical Network*, *Sensor Network*, *Pervaise Network*, *Wearable Network*, dan *Automatic Vehicle Network*.

##### Jaringan *Mobile Ad hoc Network* (MANET)

Jaringan *mobile ad hoc* atau MANET merupakan jaringan yang dibentuk secara dinamis oleh *node* yang terhubung melalui *wireless* tanpa menggunakan infrastruktur jaringan atau administrasi yang terpusat. *Node* bebas bergerak secara acak dan mengatur diri mereka seseuai kehendak mereka masing-masing, dengan demikian topologi *wireless* jaringan bisa berubah dengan cepat dan tak terduga (Basagni, et al. 2004).



Gambar 4.1 *Mobile Ad hoc Network* (MANETs)

Pada gambar 4.1 ditunjukkan bahwa jaringan *ad hoc* mungkin bisa terdiri dari beberapa perangkat rumah termasuk *notebook*, *Personal Computer*, dan sebagainya. Setiap *node* dapat berkomunikasi langsung dengan *node* lain yang berada dalam jangkauan transmisinya. Untuk berkomunikasi dengan *node* yang berada di luar jangkauan, *node* perlu menggunakan *intermediate node* atau *node* atau perantara untuk menyampaikan pesan (Basagni, et al. 2004).

1. Karakteristik Jaringan MANETs

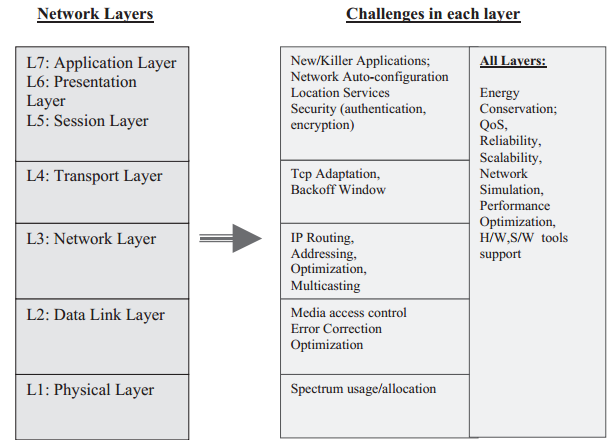
Jaringan MANETs tentu memiliki karakteristik yang sedikit berbeda dari jaringan wireless yang ada, akan tetapi jaringan *Ad hoc* tetap memiliki karakteristik yang dimiliki oleh teknologi *wireless* secara umum. Karakteristik tersebut meliputi :

1. *Node* berkomunikasi secara nirkabel / tanpa kabel dan berbagi media yang sama.
2. Jaringan *Ad hoc* merupakan sebuah jaringan yang bersifat sementara karena terbentuk secara acak dan dinamis oleh suatu kumpulan *node*.
3. Jaringan MANET tidak bergantung pada infrastruktur yang tetap atau administrasi terpusat. Setiap *node* beroperasi dalam mode *peer to peer* yang terdistribusi, bertindak sebagai *router* independen, dan menghasilkan data independen.
4. Tidak ada *router* khusus yang dibutuhkan dalam jaringan ini ; setiap *node* dapat bertindak sebagai *router* dan meneruskan masing-masing paket untuk memungkinkan berbagi informasi antar host.
5. Setiap *node* dapat bergerak secara bebas saat berkomunikasi dengan *node* lain. Topologi jaringan *Ad hoc* semacam itu bersifat dinamis karena pergerakan *nodes* yang terus berlanjut sehingga menyebabkan pola interkomunikasi antar *node* berubah secara terus menerus.
6. Kendala dalam Jaringan MANETs

Pada bagian sebelumnya telah dijelaskan berbagai macam keuntungan dari MANET itu sendiri seperti *self-reconfiguration,* dan lain – lain, namun dengan adanya kemudahan dan keuntungan tersebut tidak menyebabkan jaringan MANET bebas dari segala ancaman. Ancaman yang terjadi dikarenakan jaringan MANET memiliki karakterisktik seperti mobilitas, *multihop,*  dan kurangnya infrastruktur. Kendala atau kerumitan yang mungkin terjadi dalam jaringan MANET adalah sebagai berikut :

1. Jaringan MANETs tidak memiliki infrastruktur, jaringan *mobile ad hoc* adalah jaringan nirkabel *multihop* tanpa infrastruktur. Kurangnya infrastruktur tetap menghasilkan masalah dalam desain jaringan dibandingkan dengan jaringan yang mempunyai infrastruktur tetap. Selain itu, kurangnya entitas terpusat membuat manajemen jaringan harus didistribusikan ke semua *node*, yang membawa masalah baru dalam deteksi kesalahan dan manajemen.
2. Perubahan topologi yang sangat dinamis, karena *node* dapat bergerak secara bebas menyebabkan perubahan topologi jaringan dengan cepat dan tidak dapat diprediksi yang mengakibatkan perubahan rute dan kemungkinan *packet loss*.
3. *Bandwidth* dan kualitas *link* yang terbatas, karena *node* berkomunikasi satu sama lain melalui lebar pita yang terbatas, kapasitas variabel, rawan kesalahan, dan saluran nirkabel yang tidak aman, maka *wireless link* akan terus memiliki kapasitas yang jauh lebih rendah daripada tautan kabel dan kemacetan semakin menjadi masalah.
4. Kapabilitas *node* dan *link* yang bervariasi, setiap node dapat dilengkapi dengan satu atau lebih antarmuka radio yang memiliki kemampuan transmisi atau penerimaan yang bervariasi dan beroperasi di berbagai *bandwidth.* Keberagaman kemampuan dalam radio komunikasi dapat menyebabkan kemungkinan hubungan yang asimetris, dan juga setiap *mobile node* mungkin memiliki konfigurasi perangkat keras maupun perangkat lunak yang berbeda, sehingga menghasilkan berbagai macam dalam kemampuan pemrosesannya.
5. Operasi yang terbatasi oleh sumber daya *node*, karena baterai yang dibawa oleh masing-masing *mobile node* memiliki daya yang terbatas, kekuatan pemrosesan terbatas, sehingga membatasi layanan dan aplikasi yang dapat didukung oleh masing-masing *node*. Ini menjadi masalah yang lebih besar dalam jaringan *mobile ad hoc* karena masing-masing *node* berfungsi sebagai *end device* maupun *router* pada saat yang bersamaan, dan energi tambahan diperlukan untuk meneruskan paket dari node lain tersebut.
6. Kerangka dan keandalan jaringan, *node* jahat dan *link* yang tidak dapat diandalkan dalam jaringan MANETs dapat berdampak buruk pada kinerja jaringan secara keseluruhan. Kurangnya pemantauan dan manajemen terpusat menjadikan beberapa jenis kesalahan perilaku *node* tidak dapat dideteksi dan diisolasi dengan cepat dan mudah, sehingga menambah kompleksitas yang signifikan pada desain protokol.
7. Keamanan jaringan, jaringan nirkabel pada umumnya lebih rentan terhadap informasi dan ancaman keamanan fisik daripada jaringan kabel. Karena pada nirkabel menggunakan pesan siaran terbuka dan digunakan secara bersama berarti *node* dengan proteksi fisik yang tidak memadai rentan terhadap ancaman keamanan. Beberapa persyaratan keamanan utama dalam jaringan *ad hoc* meliputi : kerahasiaan, kontrol akses, integritas data, dan serangan *denial of service* dari *node* jahat.
8. Quality *of Service* (QoS), Jaminan QoS sangat penting untuk keberhasilan pengiriman lalu lintas jaringan multimedia. Persyaratan QoS biasanya mengacu pada serangkaian metrik yang luas termasuk *throughput*, *packet loss*, *delay*, *jitter*, *error rate*, dan seterusnya. Karakteristik jaringan *mobile ad hoc* yang sudah dijelaskan di atas seperti topologi jaringan yang berubah secara dinamis, *bandwidth* dan kualitas *link* yang terbatas, variasi kemampuan *link* menimbulkan kesulitan yang lebih dalam mencapai jaminan QoS yang diperlukan dalam jaringan *mobile ad hoc*.
9. Tantangan dalam Jaringan MANETs

Masalah dan kendala spesifik MANET yang sudah dijelaskan pada bagian sebelumnya menampilkan serangkaian tantangan dalam desain jaringan *ad hoc*. Pada gambar 4.2 dijelaskan permasalah pada jaringan *mobile ad hoc* berdasarkan layer yang ada.



Gambar 4.2 Tantangan pada Jaringan MANETs berdasarkan layer

Selain permasalahan yang ada pada setiap layer, berikut ini akan dipaparkan beberapa tantangan yang dihadapi dalam jaringan *mobile ad hoc* :

1. Kontrol Akses Media dan Optimisasi
2. *Routing* Jaringan *Ad hoc*
3. *Multicasting* dan *Broadcasting*
4. TCP *issues*

##### *Routing* pada jaringan *Mobile Ad hoc Network* (MANET)

*Routing* adalah sebuah aksi memindahkan informasi dari *node* sumber ke *node* tujuan melalui sebuah jaringan. Selama proses ini, minimal harus ada sebuah *node* penengah antara *node* sumber dan *node* tujuan yang biasa disebut dengan *intermediate node*. Konsep *routing* pada dasarnya melibatkan 2 aktifitas: Pertama, menentukan jalur *routing* yang paling optimal dan yang kedua, pengiriman sekumpulan informasi (biasa disebut paket) melalui suatu jaringan. Dalam jaringan MANETs terdapat sebuah protokol *routing* yang berfungsi uttuk melakukan tugas-tugas tersebut, namun sebelum membahas protokol *routing* terdapat beberapa hal terkait dengan tujuan dibuatnya sebuah protokol *routing*. Menurut Basagni dalam buku yang berjudul *Mobile Ad hoc Networking* setidaknya terdapat 5 alasan diantaranya :

* *Minimal Control Overhead*, dalam melakukan pengontrolan sebuah pesan diperlukan *bandwidth*, sumber daya pemrosesan, dan daya baterai untuk mentransmisikan dan menerima pesan. Protokol *routing* seharusnya tidak mengirimkan pesan yang lebih dari jumlah minimum pesan kontrol yang mereka butuhkan untuk melakukan operasi, dan harus dirancang sedemikian rupa sehingga menggunakan jumlah yang relatif kecil. Oleh karena itu, mengurangi pesan kontrol juga membantu menghemat daya baterai.
* *Minimal Processing Overhead*, penggunaan protokol *routing* yang ringan dan memerlukan proses yang minimal dari daya baterai dapat memperpanjang masa pakai baterai secara keseluruhan.
* *Multihop* *routing* capability, karena jangkauan transmisi nirkabel dari *mobile node* seringkali terbatas, maka sumber dan tujuan biasanya tidak berada dalam jangkauan transmisi secara langsung antara satu dengan yang lain. Oleh karena itu, protokol *routing* harus dapat menemukan rute *multihop* antara sumber dan tujuan sehingga komunikasi antar *node* tersebut dapat terjadi.
* *Dynamic topology maintenance*, begitu rute terbentuk kemungkinan beberapa *link* dalam rute akan rusak karena pergerakan *node*. Agar sumber dan tujuan dapat terus menerus berkomunikasi, jalur *routing* yang layak harus dijaga, bahkan saat *node intermediate* atau bahkan *node* sumber atau tujuan melakukan pergerakan. Selanjutnya, karena terdapat jeda pada jaringan *ad hoc*, maka jeda tautan harus ditangani dengan cepat dengan menggunakan minimum *overhead* yang terkait.
* *Loop prevention*, perulangan pada *routing* terjadi ketika beberapa *node* di sepanjang jalur memilih *hop* yang sama dengan *node* awalnya. Bila ada *loop* *routing*, paket data dapat melintasi jalur beberapa kali sampai salah satu jalur diperbaiki dan *loop* dihilangkan atau sampai waktu dihabiskan (TTL) paket mencapai nol. Karena *bandwidth* yang sedikit dan pemrosesan paket yang besar serta penerusan yang mahal, proses perulangan *routing* sangat boros dalam mengonsumsi sumber daya dan merugikan jaringan, Oleh karena itu, perulangan dalam *routing* harus dihindari.

1. Protokol *Routing* pada Jaringan MANETs

Dalam perkembangannya setidaknya terdapat tiga kategori umum yang sering digunakan dalam pengelompokkan protokol *routing* dalam jaringan MANETs. Protokol-protokol *routing* tersebut dikategorikan berdasarkan pendekatan proaktif *routing*, reaktif *routing* dan *hybrid* *routing*, berikut penjelasan untuk masing-masing pendekatan protokol *routing* tersebut :

1. Proaktif *Routing*

Pendekatan *routing* proaktif ini dirancang untuk jaringan *ad hoc* ini berasal dari protokol tradisional yaitu *distance vector* dan *link state* yang dikembangkan untuk internet kabel. Karakteristik utama dari pendekatan proaktif adalah bahwa setiap *node* dalam jaringan memelihara rute ke setiap node lain di jaringan setiap saat. Pembuatan dan pemeliharaan rute dilakukan melalui beberapa kombinasi pembaharuan rute secara periodic. Pembaruan berkala terdiri dari pertukaran informasi antar *node* pada interval waktu yang ditentukan. Keuntungan menggunakan protokol *routing* bertipe proaktif seperti *Destination-Sequence Distance Vector* (DSDV) maupun *Optimized Link State Routing* (OLSR) adalah rute selalu tersedia ketika dibutuhkan setiap saat karena setiap node selalu memelihara jalur yang berada pada tabel *routing* mereka, akan tetapi protokol bertipe ini juga memiliki kelemahan utama yaitu kontrol *overhead* yang berlebih dan sumber daya yang dikonsumsi sebuah *node* menjadi cepat habis karena selain digunakan sebagai mobilitas perpindahan juga digunakan sebagai pembaharuan tabel *routing* setiap saat dalam jangka waktu yang sudah ditentukan.

1. Reaktif *Routing*

Teknik protokol *routing* yang bertipe reaktif atau biasa disebut *on-demand routing* mengambil pendekatan *routing* yang sangat berbeda dari pada protokol proaktif. Sebagian besar *overhead* pada sebuah *node* berasal dari aktivitas *node* itu sendiri dalam rangka menjaga rute agar tetap tersedia dan dalam rangka mobilitas *node*. Pendekatan protokol *routing* reaktif ini didasarkan pada internet tradisional dengan cara tidak menjaga ketersediaan rute secara terus menerus pada *node* yang saling terhubung dalam jaringan. Sebagai gantinya, reaktif *routing* bekerja ketika rute yang ada pada sebuah *node* tidak ditemukan dan protokol ini aru akan melakukan prosedur pencarian rute. Sehingga jika tidak terjadi komunikasi antar *node* karena ketiadaan transmisi paket data akan membuat sumber daya masing-masing *node* bertahan lebih lama. Namun protokol bertipa reaktif seperti *Ad hoc On-demand Distance Vector* (AODV) dan *Dynamic Source Routing* (DSR) juga mempunyai kekurangan yaitu ketika tidak terdapat rute dalam tabel *routing* sebuah *node*, maka diperlukan tambahan waktu untuk *node* melakukan prosedur pencarian rute.

1. *Hybrid* *Routing*

Protokol *routing* bertipe hibrid adalah protokol yang mengintegrasikan antara protokol *routing* bertipe proaktif dan reaktif dalam berbagai cara. Protokol *hybrid* pada waktu tertentu menunjukkan perilaku sebagai prokokol yang bersifat proaktif dan pada waktu yang lain menunjukkan perilaku sebagai protokol reaktif. Protokol ini memungkinkan fleksibilitas berdasarkan karakteristik jaringan. Yang termasuk dalam protokol *routing* dengan pendekatan *hybrid* ini adalah *Zone Routing Protocol* (ZRP) dan *Algoritma Distance Routing Effect for Mobility*.

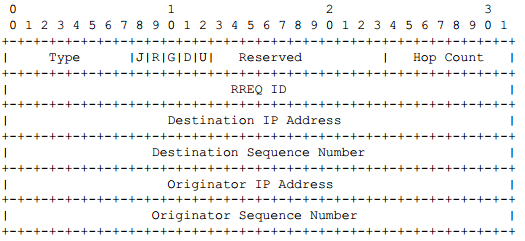
1. Protokol *Routing Ad hoc Ondemand Distance Vector* (AODV)

Pada penelitian jaringan MANETs yang dijalankan nantinya akan menggunakan protokol *routing* *Ad hoc Ondemand Distance Vector* (AODV) yang termasuk kedalam jenis protokol reaktif, oleh karena itu berikut ini akan dijelaskan penjelasan secara mendetail protokol *routing* tersebut.

Algoritma *Ad hoc On-demand Distance Vector* memungkinkan *routing* dinamis, *multihop* antara *mobile node* yang berpartisipasi dan memelihara jaringan *ad hoc*. AODV memungkinkan *mobile node* untuk mendapatkan rute tujuan baru dengan cepat, dan sebuah *node* tidak perlu mempertahankan rute ke *node* tujuan yang tidak berpartisipasi dalam jaringan *ad hoc*. AODV juga memungkinkan *mobile node* untuk merespon kerusakan *link* dan perubahan topologi jaringan dengan waktu yang tepat. Salah satu ciri khas AODV adalah penggunaan nomor urut tujuan untuk setiap rute yang ada didalam tabel *routing*. Nomor urut tujuan dibuat oleh *node* tujuan untuk disertakan bersama dengan informasi rute yang dikirim *node* yang meminta.

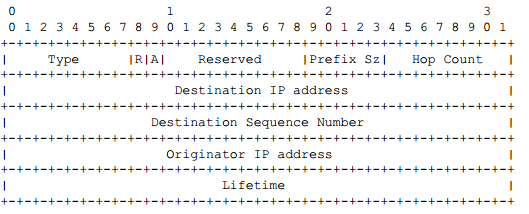
Protokol AODV menggunakan UDP dengan port 654 sebagai protokol untuk melakukan transport antar *node,* secara umum terdapat 4 pesan yang digunakan oleh protokol AODV dalam menginformasikan rute dan mentransmisikan paket data, di antaranya :

1. *Ruote Request* (RREQs)



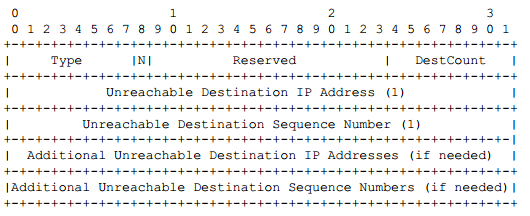
Gambar 4.3 Format Pesan *Route Request*

1. *Route Reply* (RRep)



Gambar 4.4 Format Pesan *Route Reply*

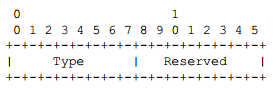
1. *Route Error* (RErr)



Gambar 4.5 Format Pesan *Route Error*

Pesan RERR dikirim setiap kali terjadi *link* yang berhenti atau terputus sehingga beberapa *node* tujuan tidak dapat dijangkau oleh *node* tetangga.

1. *Route Reply Acknowledgement* (RRep-Ack)



Gambar 4.6 Format Pesan *Route Reply Acknowledgement*

Pesan ini harus dikirim sebagai tanggapan atas pesan RREP dengan *flag* 'A'.

Adapun operasi pesan-pesan tersebut dalam protokol *routing* AODV adalah sebagai berikut :

1. *Maintaining Sequence Number*

Setiap masukan rute baru pada tabel *routing* di setiap *node* harus memasukkan informasi terbaru tentang nomor urut untuk alamat IP *node* tujuan. Nomor urut ini diperbarui setiap kali sebuah *node* menerima informasi baru tentang nomor urut yang berasal dari pesan RREQ, RREP, atau RERR. Nomor urut tujuan atau *destination sequence number* dapat bertambah nilainya ketika mengalami 2 kondisi yaitu sebelum *node* sumber melakukan pencarian rute dan sebelum *node* tujuan memulai membuat RRepyang dilakukan sebagai respon terhadap RReq. Penomoran pada *sequence number* protokol *routing* AODV menggunakan bilangan integer 32 bit dan tidak menggunakan bilangan negatif, karena nantinya akan digunakan sabagai perbandingan untuk menentukan sebuah rute yang dimasukkan kedalam tabel *routing*. Jika nomor urut kurang dari 0 maka akan dianggap sebagai rute atau informasi yang sudah basi dan AODV harus menghilangkan rute dengan nomor urut tersebut. Sebuah *node* dapat mengubah nomor urut dalam data tabel *routing* ketika *node* itu sendiri merupakan *node* tujuan dan mencoba membuat rute yang baru untuk dirinya sendiri, atau *node* menerima pesan RReq yang baru, dan ketika jalur menuju *node* tujuan rusak.

1. *Route Table Entries and Precursor Lists*

Ketika sebuah *node* menerima paket kontrol AODV dari *node* tetangga, atau membuat atau memperbarui rute untuk tujuan atau subnet tertentu, ia akan memeriksa data tabel *routing* untuk untuk tujuan tersebut. Jika tidak ada data yang sesuai untuk tujuan, maka data akan dibuat. Nomor urut ditentukan dari informasi yang terkandung dalam paket kontrol, atau kolom nomor urut yang valid. Rute hanya diperbarui jika nomor urut *node* tujuan yang baru lebih tinggi dari nomor urut tujuan di tabel rute, atau nomor urut sama, tapi jumlah *hop* route yang baru lebih kecil dari jumlah *hop* yang ada di tabel *routing*, atau nomor urut tidak diketahui.

1. *Generating Route Request* (RReqs)

Sebuah *node* menyebarluaskan pesan RReq saat ia memerlukan rute ke tujuan dan tidak memiliki satu rutepun yang tersedia. Hal ini dapat terjadi jika tujuan sebelumnya tidak diketahui oleh *node*, atau jika rute yang sebelumnya valid akan tetap sudah kadaluarsa. Kolom nomor urut tujuan dalam pesan RReq adalah nomor urut tujuan terakhir yang diketahui untuk tujuan. Jika tidak ada nomor urut yang diketahui, flag nomor urut yang tidak diketahui harus diisi. Nomor urutan *node* asal dalam pesan RReq adalah nomor urut *node* itu sendiri, yang bertambah sebelum dimasukkan ke dalam RREQ. Selanjutnya kolom ID RREQ bertambah dari satu dari RREQ ID terakhir yang digunakan oleh *node*. Setiap *node* hanya menyimpan satu ID RREQ dan kolom *HopCount* diatur ke nol.

Sebelum menyiarkan pesan RReq, *node* yang berasal akan menahan ID RREQ dan alamat IPnya sendiri dari RREQ untuk PATH\_DISCOVERY\_TIME. Dengan cara ini, ketika *node* menerima paket lagi dari tetangganya, ia tidak akan memproses ulang dan meneruskan paket. Jika pencarian rute telah diupayakan sebanyak RREQ\_RETRIES kali dan nilai pada kolom TTL maksimum tanpa menerima RRep, maka semua paket data yang ditujukan untuk tujuan harus dikeluarkan dari buffer dan pesan *Destination Unreachable* harus segera dikirim ke aplikasi. Jika RRep tidak diterima dalam waktu tertentu, *node* sumber mengirimkan RReq baru. Saat menghitung waktu untuk menunggu RREP setelah mengirim RREQ kedua, *node* sumber kemuadian menggunakan *binary exponential backoff*. Oleh karena itu jika RRep tidak diterima dalam jangka waktu 2 \* NET\_TRAVERSAL\_TIME, maka RReq lain dapat dikirim sampai RREQ\_RETRIES tambahan setelah RReq pertama. Untuk setiap usaha tambahan, waktu tunggu untuk RRep dikalikan dengan 2.

1. *Processing and Forwarding Route Request* (RReqs)

Ketika sebuah *node* menerima RReq, pertama kali langsung melakukan pembaruan rute ke *hop* sebelumnya tanpa nomor urut yang valid, kemudian memeriksa apakah telah menerima RReq dengan ALamat IP dan RReq ID yang sama dalam PATH\_DISCOVERY\_TIME terakhir. Jika RReq dengan RReq ID yang sama pernah diterima maka *node* tersebut dibuang. Selanjutnya jika ternyata belum terdapat RReq ID yang sama maka *node* tersebut harus menambahkan nilai *hop* pada pesan RReq. Kapan pun ketika sebuah *node* menerima pesan RReq, maka nilai *life time* untuk alamat IP Originator ditetapkan maksimal (ExistingLifetime, MinimalLifetime), di mana MinimalLifetime = (waktu sekarang + 2 \* NET\_TRAVERSAL\_TIME - 2 \* *HopCount* \* NODE\_TRAVERSAL\_TIME). Jika sebuah *node* tidak menghasilkan RRep dan jika *header* IP yang masuk memiliki TTL lebih besar dari 1, *node* tersebut akan memperbarui dan menyiarkan RReq ke alamat 255.255.255.255 pada masing-masing antarmuka yang dikonfigurasi. Untuk memperbaharui RREQ, kolom TTL atau *hop* pada *header* IP akan dikurangi satu, dan kolom *HopCount* dalam pesan RReq akan ditambah satu untuk memperhitungkan *hop* baru melalui *node* perantara. Terakhir, nomor urutan *node* tujuan diatur dengan nilai maksimum berdasarkan nilai yang diterima dalam pesan RReq, dan nilai urutan tujuan tersebut dipertahankan sampai dengan *node* tujuan yang diminta. Namun, sebuah *node* *forwarding* harus memodifikasi nilai nomor urut tujuan, walaupun nilai dalam RReq yang masuk lebih besar daripada nilai yang ada saat ini, dan jika tidak dimodifikasi sebuah *node* akan menghasilkan RRep sehingga *node* tersebut akan membuang pesan RReq.

1. *Generating Route Replies*

Sebuah *node* akan menghasilkan sebuah pesan RRep ketika *node* itu sendiri merupakan tujuannya atau *node* tersebut memiliki rute tujuan untuk dilewatinya. Saat membuat pesan RRep sebuah *node* menyalin Alamat IP Tujuan dan nomor urut *node* sumber dari pesan RReq ke kolom yang sesuai dalam pesan RRep. Pada prosesnya pengolahan RRep sedikit berbeda bergantung pada *node* tersebut, apakah *node* tujuan yang diminta atau hanya *node* perantara saja. Setelah dibuat *node* kemudian melakukan *unicast* ke *hop* berikutnya menuju *node* yang melakukan RReq pertama kali. Pesan RRep juga melakukan penambahan pada kolim *HopCount* dengan menambah satu ketika melewati *node* yang bukan tujuannya. Pesan RRep bisa dibuat dalam dua jenis *node* yang berbeda yaitu ketika *node* sebagai perantara dan *node* sebagai tujuan. Jika *node* yang mebuat pesan RRep adalah *node* tujuan itu sendiri, maka *node* harus menambah satu pada nomor urut satu jika nomor urut pada paket RReq sama dengan nilai yang bertambah itu. Jika tidak sama maka *node* tujuan tidak perlu mengubah nomor urutannya sebelum membuat pesan RRep. Kemudian *node* tujuan menempatkan nomor urut pada kolom nomor urut tujuan RRep dan mengatur nilai kolom *HopCount* menjadi nol kembali. Jika *node* yang menghasilkan RRep bukan *node* tujuan melainkan *node* perantara maka ia kemudian menyalin nomor urut yang diketahui ke dalam kolom *Destination Sequence Number* dalam pesan RRep. *Node* perantara memperbarui data rute dengan menempatkan *node* *hop* terakhir ke dalam daftar prekursor untuk diteruskan.

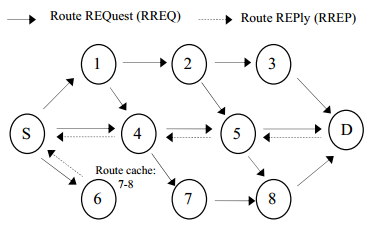
1. *Receiving and Forwarding Route Replies*

Ketika sebuah *node* menerima pesan RRep, ia kemudian mencari untuk rute ke *hop* sebelumnya. Selanjutnya, *node* menambahkan satu pada nilai *hop* di RRep. Penamhahan nilai pada RRep kemudian disebut *New Hop Count.* Kemudian *node* membuat rute untuk tujuan selanjutnya, jika tidak ada *node* membandingkan nomor urut pada pesan RRep dengan nomor urut yang ada. Setelah perbandingan, data dapat masuk kedalam tabel *routing* ketika nomor urut pada tabel *routing* ditandai sebagai tidak valid dalam entri tabel rute, nomor urutnya sama, tapi rutenya ditandai sebagai rute yang tidak aktif, atau nomor urutnya sama, dan nilai *New Hop Count* lebih kecil dari jumlah *hop* dalam daftar tabel *routing*.

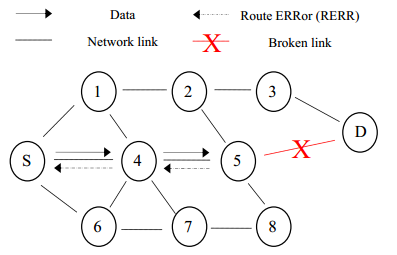
1. *Route Error (RErr) Messages, Route Expiry, and Route Deletion*

Pada umumnya, kesalahan sebuah rute dan hancurnya sebuah *link* memerlukan langkah-langkah berikut seperti memvalidasi rute yang ada, mencantumkan *node* tujuan yang terpengaruh dan jika ada kemudian mengiriman RErr ke *node* tetangga tersebut. Pesan *Route Error* (RErr) boleh di *broadcast* jika terdapat banyak prekursor, namun jika hanya terdapat 1 prekursor saja maka dilakukan unicast atau secara iteratif unicast ke semua prekursor. Bahkan ketika pesan RERR secara iteratif tidak diketahui beberapa prekursor, ini dianggap sebagai pesan kontrol tunggal untuk tujuan deskripsi.

Dari penjelasan diatas setidaknya dapat disimpulkan bahwa dalam AODV terdapat dua operasi yang mendasar yaitu *route discovery* dan *route maintenance* dan operasi tersebut di ilustrasikan seperti pada gambar 4.7 dan gambar 4.8 berikut.



Gambar 4.7 Ilustrasi *Route Discovery*

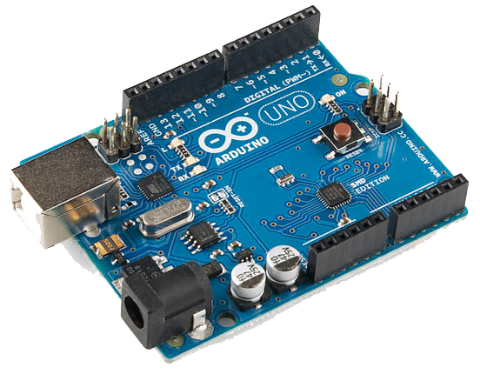


Gambar 4.8 Ilustrasi *Route Maintence*

##### Mikrokontroler dan Arduino

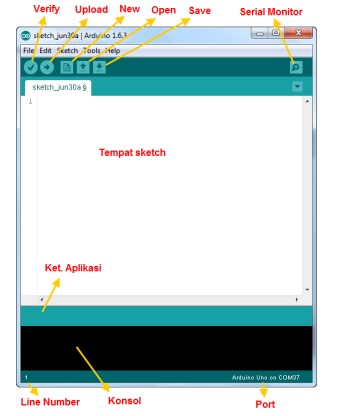
*Microcontroller* adalah komputer yang berukuran mikro dalam satu chip IC (*integrated circuit*) yang terdiri dari *processor*, *memory*, dan antarmuka yang bisa diprogram. Jadi disebut komputer mikro karena dalam IC atau chip  
mikrokontroler terdiri dari CPU, *memory*, dan I/O (*Input/Output*) yang bisa kita kontrol dengan memprogramnya. I/O juga sering disebut dengan GPIO (*General Purpose Input Output Pins*) yang berarti : pin yang bisa kita program sebagai input atau output sesuai kebutuhan.

*Arduino* dikatakan sebagai sebuah *platform* dari *physical computing* yang bersifat *open source*. *Physical computing* adalah membuat sebuah sistem atau  
perangkat fisik dengan menggunakan *software* dan *hardware* yang sifatnya interaktif yaitu dapat menerima rangsangan dari lingkungan dan merespon balik. *Physical computing* adalah sebuah konsep untuk memahami hubungan yang manusiawi antara lingkungan yang sifat alaminya adalah analog dengan dunia digital. Arduino tidak hanya sekedar sebuah alat pengembangan, tetapi juga kombinasi dari *hardware*, bahasa pemrograman dan *Integrated Development Environment* (IDE) yang canggih. IDE adalah sebuah *software* yang sangat berperan untuk menulis program, meng-*compile* menjadi kode biner dan mengunggah ke dalam memory mikrokontroler.



Gambar 4.9 Arduino Uno R3

Bahasa pemrograman yang digunakan adalah C/C++. Dalam sebuah  
mikrokontroler Arduino dapat pula ditanamkan berbagai macam *library* maupun metode selama kapasitas memori dari sebuah mikrokontroler mencukupi.



Gambar 4.10 Tampilan Arduino IDE

Bagian-bagian dalam IDE antara lain :

1. ***Verify*** : pada versi sebelumnya dikenal dengan istilah *Compile*. Sebelum aplikasi diupload ke *board* Arduino dilakukan verifikasi terlebih dahulu pada *sketch* yang dibuat. Jika ada kesalahan pada *sketch* muncul pesan *error*. Proses *Verify* / *Compile* mengubah *sketch* ke *binary code* untuk diupload ke mikrokontroller.
2. ***Upload*** : tombol ini berfungsi untuk mengupload *sketch* ke *board* Arduino. Walaupun kita tidak mengklik tombol *verify*, maka *sketch* akan di-*compile*, kemudian langsung diupload ke *board*. Berbeda dengan tombol *verify* yang hanya berfungsi untuk memverifikasi *source code* saja.
3. ***New Sketch*** : Membuka window dan membuat *sketch* baru.
4. ***Open Sketch*** : Membuka *sketch* yang sudah pernah dibuat. *Sketch* yang dibuat dengan IDE Arduino akan disimpan dengan ekstensi file **.ino.**
5. ***Save Sketch*** : Menyimpan *sketch*, tapi tidak meng-*compile*.
6. ***Serial Monitor*** : Membuka *interface* untuk komunikasi serial.
7. **Keterangan Aplikasi** : pesan-pesan yang dilakukan aplikasi akan muncul di sini, misal "*Compiling*" dan "*Done Uploading*" ketika kita mengcompile dan mengupload *sketch* ke *board* Arduino.
8. **Konsol** : Pesan-pesan yang dikerjakan aplikasi dan pesan-pesan tentang *sketch* akan muncul pada bagian ini. Misal, ketika aplikasi mengcompile atau ketika ada kesalahan pada *sketch* yang kita buat, maka informasi *error* dan baris akan diinformasikan di bagian ini.
9. **Baris *Sketch*** : bagian ini akan menunjukkan posisi baris kursor yang sedang aktif pada *sketch*.
10. **Informasi Port** : bagian ini menginformasikan *port* yang dipakah oleh *board* Arduino.

##### Modul XBee

1. Pengertian XBee

Modul RF XBee merupakan alat pengiriman data secara nirkabel yang dikeluarkan oleh Perusahaan International Digi. XBee sepenuhnya menggunakan protokol komunikasi yang diterapkan untuk komunikasi data yang disediakan untuk kebutuhan komunikasi seperti *wireless sensor network* (WSN). Fitur yang ada didalam modul XBee seperti media akses yang bisa digunakan untuk komunikasi dua arah, pengalamatan,pendeteksian error, *acknowledgement*, dan juga fitur tam bahan untuk keperluan diluar komunikasi yang digunakan dalam pemantauan dan penendalian perangkat jarak jauh. XBee menggunakan protokol IEEE 802.15.4 yang dikenal dengan *Low-Rate Wireless Personal Area Network* (LR-WPAN) dengan dukungan pengiriman data mancapai 250kbps. Walaupun tidak ditujukan untuk data dalam jumlah yang besar seperti file gambar, namun XBee menyediakan sarana untuk memindahkan data dengan cepat diantara *node-node* pada jaringan. *Data Rate* pada XBee bisa mencapai kecepatan 250.000 bps (bits per second) dan throughputnya mencapai 35000 bps.

Dalam berkomunikasi XBee dapat digunakan dalam dua mode yaitu mode AT dan API (*Application Programming Interface*) untuk mengirim dan menerima data. dalam mode AT atau *Transparant Mode*, hanya data pesan yang berasal dari modul yang dikirim dan diterima oleh kontroler karena pada mode ini biasa digunakan untuk membangun komunikasi yang sederhana. Sementara dalam mode API *programmer* bisa memasukkan data sesuai yang diinginkan sehingga dapat membuat pengguna lebih fleksibel dan dapat meningkatkan kehandalan XBee. Perlu diperhatikan ketika menggunakan XBee adalah komunikasi tidak harus menggunakan mode yang sama baik AT maupun API, karena kedua mode tersebut dapat digunakan untuk saling berkomunikasi satu sama lain dengan protokol IEEE 802.15.4 LR-WPAN.

Modul XBee dapat memperluas jangkauan jaringan dengan *routing*, mereka membentuk jaringan sendiri (*Self-establish*), dan *Self-healing* untuk memindahkan data dalam jaringan. Terdapat dua protokol jaringan yang dapat digunakan untuk modul XBee yaitu ZigBee dan DigiMesh, sehingga setiap perangkat XBee dapat diatur dengan salah satu diantara dua protokol yang tersedia baik dalam mode AT maupun API. Pada jaringan Mesh termasuk protokol ZigBee sebuah perangkat dapat di atur satu dari salah satu tugas yang tersedia, yaitu :

* *Coordinator*, *node* yang bertugas sebagai koordinator bertugas untuk membangun dan memelihara jaringan dengan menetapkan alamat dalam pembuatan sebuah rute.
* *Routers*, tugas *router* adalah memindahkan atau hanya sebagai perantara antar *node* satu dengan yang lain karena jarak yang terlalu jauh atau bisa dikatakan dengan jumlah lompatan (*hop*) yang banyak.
* *Endpoints*, sedangkan untuk ep atau *endpoint* hanya berfungsi sebagai tujuan terakhir dari sebuah paket data.

1. Pengalamatan XBee

Pengalamatan pada XBee Terbagi menjadi dua buah alamat yang berbeda yaitu alamat 16 bit dan alamat 64 bit dengan masing-masing karakteristik yang berbeda. Pada pengalamatan 64 bit atau biasa disebut juga dengan *extends address* dimana alamat tersebut menjadi alamat unik untuk setiap perangkat XBee. Sedangkan untuk alamat 16 bit tersedia ketika sebuah perangkat XBee sudah tergabung dalam jaringan ZigBee, dan disebut juga dengan *network address*. Sebuah perangkat XeBe yang diatur sebagai *end device* akan mendapatkan *network address* dari sebuah *router* atau koordinator dan kemudian alamat tersebut adalah alamat yang digunakan untuk pengiriman data.

1. Pengiriman Data pada XBee

Pada XBee, data dapat dikirimkan dengan cara *broadcast* dan *unicast*, ketika mengirimkan data dengan metode *broadcast* maka pengiriman ditujukan untuk semua perangkat yang saling terhubung dengan catatan bahwa *node* yang berfungsi sebagai *router* dan koordinator yang menerima paket *broadcast* akan mengirimkan kembali paket data yang di *broadcast* tersebut sebanyak 3 kali, sedangkan untuk metode *unicast* pengiriman hanya dilakukan pada satu tujuan baik ke *neighbour* *node* maupun melewati beberapa lompatan atau *hop*.

1. *Routing* pada XBee

Secara umum terdapat tiga protokol *routing* yang digunakan pada XBee yaitu *Ad hoc On-demand Distance Vector* (AODV) *mesh Routing*, *Many-to-One Routing*, dan *Source Routing*. Setiap protokol *routing* tersebut pada intinya untuk mencari dan memelihara rute untuk masing-masing tujuan. Penggunaaan dan pemilihan protokol *routing* diatas juga dapat disesuaikan dengan kebutuhan, AODV baik digunakan pada jaringan yang tidak terlalu besar yakni dengan tujuan atau *end device sekitar* 40, sedangkan untuk tujuan lebih dari itu atau untuk jaringan yang berskala besar bisa digunakan protokol *Source Routing* yang lebih mampu menangani pencarian dan pemeliharaan rute dalam cakupan yang besar.

1. Topologi Jaringan XBee

Pada XBee terdapat beberapa topologi jaringan yang dapat diimplementasikan, topologi tersebut diantaranya :

* Topologi Star

Pada topologi star komunikasi dilakukan antara perangkat dengan sebuah pusat pengontrol tunggal yang kemudian disebut sebagai koordinator PAN (*Personal Area Network*). Setiap jaringan star akan memilih sebuah pengenal PAN yang tidak sedang digunakan oleh jaringan lain didalam jangkauan radionya. Ini akan mengijinkan setiap jaringan star untuk bekerja secara tersendiri.

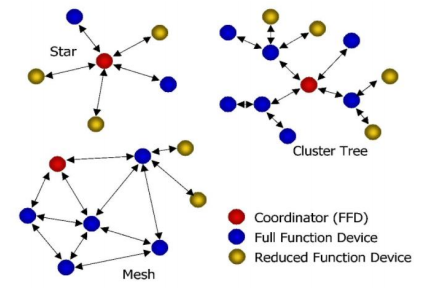
* Topologi Mesh

Dalam topologi *peer to peer* juga hanya ada satu koordinator PAN. Berbeda dengan topologi *star*, setiap perangkat dapat berkomunikasi satu sama lain sepanjang ada dalam jarak jangkauannya. *Peer to peer* dapat berupa *ad hoc, Self-organizing* dan *self healing*.

* Topologi Cluster Tree

*Cluster tree* seperti merupakan sebuah model khusus dari jaringan *peer*  
*to peer* dimana sebagian besar perangkatnya adalah FFD dan sebuah RFD yang terhubung ke jaringan *cluster tree* sebagai *node* tersendiri di akhir percabangan. Salah satu dari FFD dapat berlaku sebagai coordinator (router) dan memberikan layanan sinkronisasi ke perangkat lain dan coordinator lain. Hanya satu dari koordinator ini adalah koordinator PAN yang mengontrol seluruh *node*.

Ilustrasi untuk ketiga penjelasan diatas dapat dilihat pada gambar berikut.



Gambar 4.11 Topologi Jaringan XBee

1. *Tool* Konfigurasi XBee

Untuk melakukan konfigurasi pada modul XBee diperlukan sebuah *utilities,* dan Digi mengeluarkan sebuah perangkat lunak untuk memenuhi kebutuhan tersebut dan diberi nama X-CTU. Program ini dirancang dengan tujuan untuk memudahkan interaksi dengan file *firmware* yang terdapat pada produk RF Digi dan menyediakan antarmuka grafis yang mudah digunakan. Operasi pada X-CTU dikelompokkan menjadi tiga mode kerja yaitu mode konfigurasi, mode konsol, dan mode jaringan. Mode kerja yang dipilih menentukan operasi spesifik yang dapat di lakukan dengan modul radio dan pada satu modul radio hanya bisa dimasukkan satu mode kerja tidak bisa tiga sekaligus. Secara default X-CTU berada dalam mode konfigurasi, mode tersebut digunakan untuk mengkonfigurasi perangkat modul radio, sedangkan mode konsol digunakan untuk berinteraksi atau berkomunikasi dengan modul radio yang dipilih, dan mode jaringan digunakan untuk menemukan dan memvisualisasikan topologi dan interkoneksi jaringan.

##### Kalilinux

Kalilinux merupakan sebuah aplikasi *open source* yang digunakan untuk melakukan *penetration test* untuk menguji keamanan pada sebuah jaringan. Kalilinux adalah versi terbaru dari distribusi Backtrack. Terdapat banyak sekali fitur yang bisa digunakan dengan kalilinux salah satunya adalah aplikasi hping3 yang berfungsi untuk melakukan serangan *Denial of Service* (DoS) *attack* dengan salah satu jenisnya adalah *data* *flooding attack*. Aplikasi ini nantinya yang akan digunakan sebagai alat untuk melakukan *data* *flooding attack* pada jaringan *mobile ad hoc*.

##### Parameter Kinerja

Dalam melakukan proses analisis kinerja jaringan berdasarkan *Quality Of Service* (QoS) (Leon-Garcia and Widjaja 2003). Parameter yang digunakan untuk melakukan pengukuran kinerja adalah packet *delivery ratio* (pdr), *delay*, *throughput* dan *packet loss ratio* (plr). Penjelasan dari setiap parameter dijelaskan pada uraian berikut.

1. *Packet Delivery Ratio* (PDR)

*Packet Delivery Ratio* atau PDR merupakan perbandingan banyaknya jumlah paket yang diterima oleh *node* penerima dengan total paket yang dikirimkan dalam suatu periode waktu tertentu. Atau juga bisa dihitung dengan cara mengurangi jumlah paket keseluruhan yang dikirim dengan paket yang *loss* atau hilang. Secara matematis *Packet Delivery Ratio* (PDR) dapat dihitung dengan persamaan berikut :

1. *Delay*

*Delay* atau waktu tunda merupakan jumlah total waktu pengiriman paket dalam satu kali pengamatan. Dalam hal ini satu kali pengiriman paket data dibagi dengan jumlah usaha pengiriman yang berhasil dalam satu kali pengamatan tersebut. Secara umum *delay* dapat dinyatakan dengan pernyataan berikut :

Adapun menurut versi TIPHON (Telecommunications and Internet Protocol Harmonization Over Network) standarisari nilai delay sebagai  
berikut.

Tabel 4.2 Standarisasi Delay versi *TIPHON*

|  |  |
| --- | --- |
| **Kategori *Latency*** | **Besar *Delay*** |
| Sangat Bagus | < 150 ms |
| Bagus | 150 s/d 450 ms |
| Sedang | 300 s/d 450 ms |
| Buruk | < 450 ms |

(Sumber *TIPHON*)

1. *Throughput*

*Throuhgput* merupakan suatu istilah yang mendefinisikan banyaknya bit yang diterima dalam selang waktu tertentu dengan satuan bit per *second* yang merupakan kondisi dari data *rate* yang sebenarnya. Secara umum *Throughput* dinyatakan dalam persamaan berikut :

Pada Xbee sendiri throughput mempunyai nilai yang berbeda-beda tergantung fungsi *node* tersebut. Nilai tersebut dipengaruhi oleh jumlah lompatan (*hop*), enkripsi yang *enable* maupun *disable,* mode *sleeping* pada *end devce node,* dan rute yang rusak, seperti yang tertera pada tabel berikut.

Tabel 4.3 Data *Througput* XBee

|  |  |
| --- | --- |
| **Konfigurasi *Node*** | **Data *Throughput*** |
| **1 *hop*, RR, SD** | 35 kbps |
| **1 *hop*, RR, SE** | 19 kbps |
| **1 *hop*, RE, SD** | 25 kbps |
| **1 *hop*, RE, SE** | 16 kbps |
| **1 *hop*, ER, SD** | 21 kbps |
| **1 *hop*, ER, SE** | 16 kbps |
| **4 *hops*, RR, SD** | 10 kbps |
| **4 *hops*, RR, SD** | 5 kbps |

Keterangan :

RR = *router to router*,

RE = *router to end device* (*non-sleeping*),

ER = *end device* (*non-sleeping*) *to router*,

SD = *security disabled*,

SE = *security enabled*.

4 *hops* = 5 *nodes*, 3 *intermediate router nodes*.

1. *Packet Loss Ratio* (PLR)

Yaitu perbandingan seluruh paket data yang hilang dengan seluruh paket data yang dikirimkan antara pada sumber dan tujuan.

Di dalam implementasi jaringan, nilai packet loss ini diharapkan  
mempunyai nilai yang minimum. Secara umum terdapat empat ketegori  
penurunan kualitas jaringan berdasarkan nilai packet loss sesuai dengan versi  
TIPHON (Telecommunications and Internet Protocol Harmonization OverNetwork) standarisari nilai packet loss sebagai berikut :

Tabel 4.4 Standarisasi *Packet Loss* versi *TIPHON*

|  |  |
| --- | --- |
| **Kategori Degradasi** | ***Packet Loss*** |
| Sangat Bagus | (0 – 3)% |
| Bagus | (3 – 15)% |
| Sedang | (15 – 25)% |
| Buruk | > 25% |

(Sumber : *TIPHON*)

#### Hipotesis

Berdasarkan uraian yang sudah dijelaskan bada bab sebelumnya seperti latar belakang, perumusan masalah, tinjauan pustaka, dan landasan teori, maka dapat dikemukakan hipotesis seperti berikut :

1. Pada jaringan *mobile ad hoc* yang dibuat menggunakan XBee dapat diimplementasikan protokol *routing Ad hoc Ondemand Distance Vector* (AODV).
2. Kinerja protokol *routing Ad hoc Ondemand Distance Vector* (AODV) mengalami penurunan pada jaringan *mobile ad hoc* ketika diserang dengan *flooding attack*.
3. Dampak dari serangan *flooding attack* berpengaruh pada tingkat *availability* pada jaringan *mobile ad hoc* yang dibangun.

# BAB V

**METODE PENELITIAN**

### 3

###### Alat dan Bahan

Dalam melaksanakan penelitian ini, terdapat beberapa perangkat keras dan perangkat lunak yang di butuhkan guna menunjang penelitian yang di lakukan. Adapun beberapa perangkat tersebut di antaranya :

**Perangkat Keras**

1. Lima (5) PC/LAPTOP
2. Lima (5) Modul RF XBee Shield
3. Satu (1) Papan Arduino Uno
4. Satu (1) Xbee Shield Adapter
5. Empat (4) USB Shield XBee Adapter
6. Satu (1) Ethernet Shield Adapter

**Perangkat Lunak**

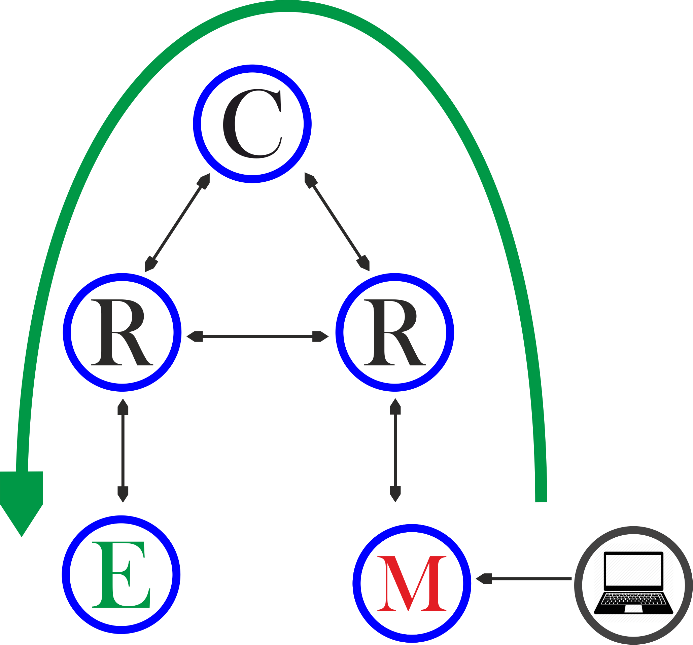
1. Sistem Operasi Windows 10
2. VMware® Workstation 12 Pro
3. Sistem Operasi Kali Linux 2017.3 AMD 64bit
4. Arduino IDE 1.8.2
5. X-CTU
6. Wireshark

###### Prosedur Penelitian

1. **Perancangan Topologi**

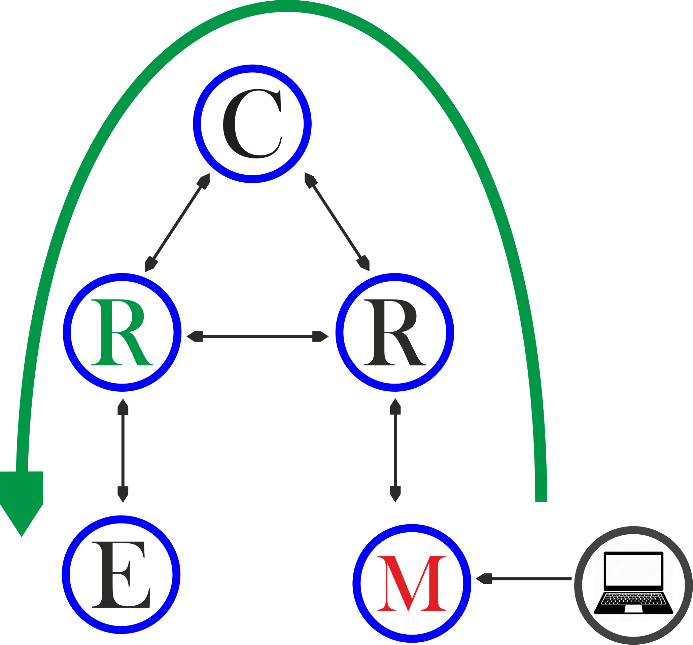
Untuk melakukan serangan *data flooding attack*, pada penelitian ini menggunakan 3 buah skenario dengan topologi mesh dengan setiap skenario terdapat perbedaan pada *node* yang akan dilakukan *flooding data*. Topologi mesh memungkinkan setiap *node* untuk mendapatkan jalur yang terbaik. Skenario yang akan digunakan sebagai acuan dalam melakukan implementasi dan analisis protokol *routing* AODV adalah sebagai berikut :

* Skenario 1 – Target *End Device*



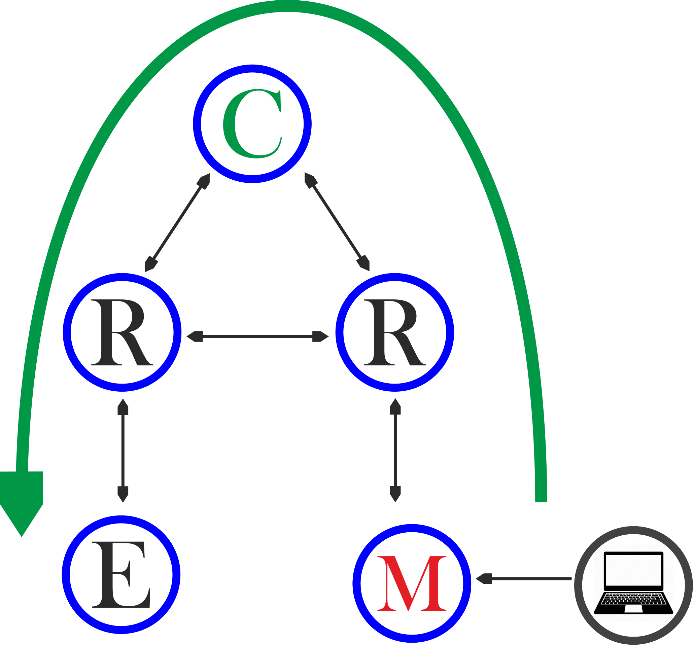
Gambar 5.12 Rancangan Topologi Jaringan MANET Skenario 1

* Skenario 2 – Target *Router*



Gambar 5.13 Rancangan Topologi Jaringan MANET Skenario 2

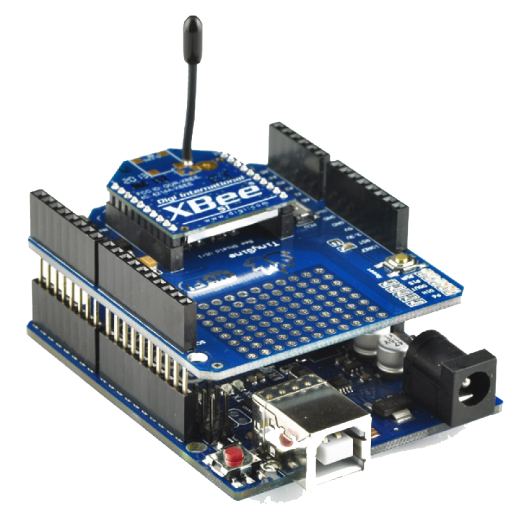
* Skenario 3 – Target Koordinator



Gambar 5.14 Rancangan Topologi Jaringan MANET Skenario 3

Dari 3 skenario diatas penjelasannya adalah seperti berikut. C, R, dan E merupakan simbol masing – masing untuk *node* yang berfungsi sebagai koordinator, *router* dan *end device*. Untuk setiap *node* akan digunakan perangkat XBee dan XBee adapter dengan catatan *node* yang diserang ditambahkan dengan perangkat arduino dan Wi-Fi *shield* adapter yang berfungsi untuk menerima paket data dari hping3 untuk diteruskan pada *node* yang bersangkutan, sedangkan untuk *node* yang disimbolkan dengan huruf M merupakan *node* jahat yang difungsikan untuk melakukan serangan terhadap *node* yang berperan sebagai koordinator, *router*, maupun *end device*. *Node* merupakan simbol untuk penyerang dengan menggunakan *tools* hping3 pada kalilinux.

Pada setiap simbol yang tertera pada topologi diatas baik simbol C, R, dan E merupakan gabungan komponen alat elektronik. Komponen-komponen elektronik tersebut berupa Arduino, XBee Shield Adapter, dan XBee yang dihubungkan melalui pin – pin yang bersesuaian sehingga jika ketiga komponen tersebut digabungkan hasilmya akan seperti pada gambar 5.4 berikut.



Gambar 5.15 Gambaran *Node*

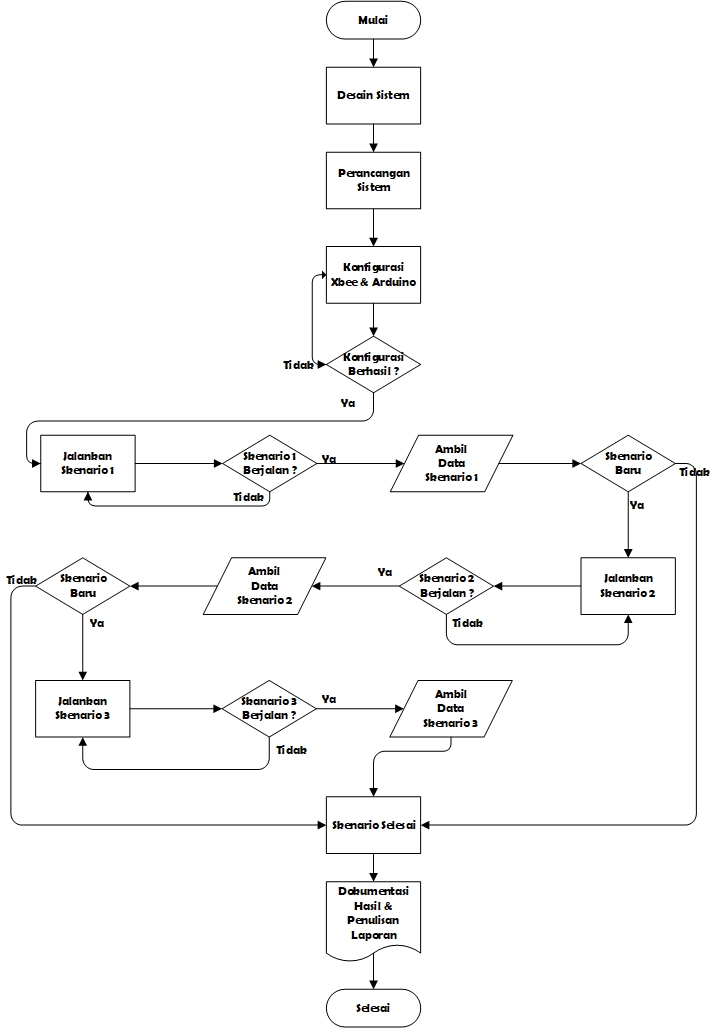
Pada saat melakukan melakukan serangan, terdapat berbagai macam ukuran data yang digunakan untuk membajiri *node* maupun jaringan dan dengan waktu yang sudah ditentukan. Tabel 5.1 menunjukkan ringkasan data dan waktu yang digunakan sebagai acuan dalam melakukan serangan.

Tabel 5.5 Ringkasan data dan waktu penyerangan

|  |  |  |  |
| --- | --- | --- | --- |
| No | *Node* yang Dipengaruhi *Data* *Flooding* | Besar Paket Data (Byte) | Waktu Penyerangan (detik) |
| 1 | *Coordinator, Router* dan *End Device Node* | 30 | 50 |
| 30 | 100 |
| 30 | 150 |
| 30 | 200 |
| 60 | 50 |
| 60 | 100 |
| 60 | 150 |
| 60 | 200 |
| 90 | 50 |
| 90 | 100 |
| 90 | 150 |
| 90 | 200 |

1. **Diagram Alir Metode Penelitian**

Metode penelitian yang dilakukan penulis dalam penelitian ini dapat dilihat pada *flow chart* berikut :



Gambar 5.16 *Flowchart* Penelitian

###### Analisis Hasil

Hasil dari penelitian yang dilakukan adalah berupa sekumpulan data yang diambil berdasarkan parameter yang telah ditentukan sebelumnya. Nilai parameter diambil pada setiap skenario yang diujicobakan, dimana masing-masing skenario akan menghasilkan data yang bervariasi. Selanjutnya data yang sudah didapatkan dilakukan pengolahan dan ditampilkan dalam bentuk grafik serta diberikan penjelasan mengenai grafik tersebut.

###### Jadwal Penelitian

Penelitian ini dirancang untuk dilaksanakan dalam waktu 4 bulan, atau sesuai dengan tabel dibawah ini.

Tabel 5.6 Jadwal Penelitian

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Tahap Kegiatan | Bulan Ke : | | | | | | | | | | | | | | | | | | |
| 1 | | | | 2 | | | | | 3 | | | | | 4 | | | | |
| 1 | 2 | 3 | 4 | | 1 | 2 | 3 | 4 | | 1 | 2 | 3 | 4 | | 1 | 2 | 3 | 4 |
| Persiapan |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  |
| Identifikasi Masalah |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  |
| Perncangan Topologi |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  |
| Pelaksanaan |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  |
| Install Kali Linux |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  |
| Konfigurasi Xbee dan Arduino |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  |
| Konfigurasi *Routing* AODV |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  |
| Percobaan dan Perbaikan |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  |
| Penyelesaian |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  |
| Pengambilan Data |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  |
| Analisis Hasil |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  |
| Penyusunan Laporan |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  |

# DAFTAR PUSTAKA

Abdelshafy, Mohamed A., and Peter J. B. King. 2015. “Dynamic Source Routing under Attacks.” *IEEE* 7.

Asriyadi, and Rahmadi Kurnia. 2016. “Unjuk Kerja Protokol ZIGBEE pada Jaringan WSN.” *Jurnal Teknik Elektro ITP* 3.

Ayrton P, Muhammad Rihga. 2016. “Implementasi dan Analisis Kinerja Protokol Routing Ad hoc On-demand Distance Vector (AODV) pada Pemantauan Suhu dan Kelembaban Ruang Berbasis XBEE.”

Bandyopadhyay, Alokparna , and Satyanarayana Vuppala. 2011. “A Simulation Analysis of Flooding Attack in MANET using NS-3.” *IEEE.*

Basagni, Stefani , Marco Conti, Silvia Giordano, and Ivan Stojmenovic. 2004. *Mobile Ad hoc Networking.* New Jersey: IEEE Press.

Bundela, Avinash S., Prashant Panse, Gaurav Sharma, and Sachin Solanki. 2016. “A Secure Routing in Ad-Hoc Network.” *IEEE.*

Çayırcı, Erdal , and Chunming Rong. 2009. *Security in Wireless Ad Hoc and Sensor Networks.* United Kingdom.

Chlamtac, I. , and M. El-Zarki. 1994. *"Introduction to Computer Networks" in Encyclopedia of Telecommunications.*

Deng, Hongmei , Wei Li, and Dharma P. Agrawal. 2002. “Routing Security in Wireless Ad Hoc Networks.” *IEEE.*

Digi. n.d. *XCTU Configuration and Test Utility Software.*

Faludi, Robert. 2011. *Building Wireless Sensor Network.* 1st. California: O’Reilly Media, Inc.

Fitriawan, Helmy, Misfa Susanto, Ahmad Surya Arifin, Danny Mausa, and Agus Trisanto. 2017. “ZigBee Based Wireless Sensor Networks and Performance Analysis in various Environments.” Bali: IEEE.

Hasibuan, Zainal A. 2007. *Metode Penelitian Pada Bidang Ilmu Komputer dan Teknologi Informasi.*

Hebel, Martin, George Bricker, and Daniel Harris. 2010. *Getting Started with XBee RF Modules : A Tutorial for BASIC Stamp and Propeller Microcontrollers.* California: Parallax.

HoudaMoudni, Mohamed Er-rouidi, HichamMounci, and Benachir El Hadadi. 2016. “Performance Analysis of AODV Routing Protocol in MANET under the Influence of Routing Attacks.” *International Conference on Electrical and Information Technologies ICEIT.* Morroco: IEEE.

Ilyas, Mohammad . 2003. *THE HANDBOOK OF ADHOC WIRELESS NETWORK.* Florida: CRC Press LLC.

Leon-Garcia, Alberto , and Indra Widjaja. 2003. *Communication Networks Fundamental Concepts and Key Architectures.* The McGraw Companies.

Lin, Y. Bing , and I. Chlamtac. 2000. *Wireless and Mobile Network Architectures.* Wiley, .

Lotfy, Poussy A., and Marianne A. Azer. 2013. “Performance Evaluation of AODV Under DoS Attack.” *IEEE.*

Murthy, C. Siva Ram , and B. S. Manoj. 2004. *Ad Hoc Wireless Networks Architectures and Protocols.* New Jersey: Prentice Hall Professional Technical Reference.

Pramono, Subuh , Arisa Olivia, Endro Warsito, and Budi Basuki. 2017. “Comparative Analysis of Star Topology and Multihop Outdoor Propagation Based on Quality of Service (QoS) of Wireless Sensor Network (WSN).” Semarang: IEEE.

Thilak, K.Deepa. 2016. “DoS Attack on VANET Routing and possible defending solutions-A Survey.” Puducherry.