**ANALISA SERANGAN *BLACKHOLE*   
PADA JARINGAN VANET**

SKRIPSI



Disusun oleh :

**Indra Setyo Wibowo**

**20190140077**

**PROGRAM STUDI TEKNOLOGI INFORMASI**

**FAKULTAS TEKNIK**

**UNIVERSITAS MUHAMMADIYAH YOGYAKARTA**

**2024**

# HALAMAN PENGESAHAN I

**SKRIPSI**

**ANALISA SERANGAN** *BLACKHOLE*  **PADA JARINGAN VANET** 

Disusun oleh :

Indra Setyo Wibowo

20190140077

Telah disetujui oleh :

|  |  |
| --- | --- |
| **Dosen Pembimbing 1** | **Dosen Pembimbing 2** |
|  |
|  |  |  |
|  |  |  |
| **Ir. Asroni, S.T., M.Eng.** | **Dr. Ronald Adrian, S.T., M.Eng.** |  |
| **NIDN : 0516047401** | **NIDN : -** |  |
|  |  |  |

# HALAMAN PENGESAHAN II

**SKRIPSI**

**ANALISA SERANGAN** *BLACKHOLE*  **PADA JARINGAN VANET **

Disusun oleh :

Indra Setyo Wibowo

20190140077

Telah dipertahankan di Depan Tim Penguji

Pada Tanggal

Selasa, 20 Agustus 2024

Susunan Tim Penguji :

|  |  |
| --- | --- |
| **Dosen Pembimbing 1** | **Dosen Pembimbing 2** |
|  | |
|  |  |  | |
|  |  |  | |
| **Ir. Asroni, S.T., M.Eng.** | **Dr. Ronald Adrian, S.T., M.Eng.** |  | |
| **NIDN : 0516047401** | **NIDN :** |  | |
|  |  |  | |
| **Dosen Penguji** | | |  | |
|  | |
|  |  |  | |
|  |  |  | |
| **-** | | |  | |
| **NIDN : -** | | |  | |

Skripsi ini telah dinyatakan sah sebagai salah satu persyaratan

Untuk memperoleh gelar Sarjana Komputer

Tanggal, 20 Agustus 2024

|  |
| --- |
| Mengesahkan, |
| Kepala Program Studi Teknologi Informasi |
|  |
|  |
| **Dr. Reza Giga Isnanda, S.T., M.Sc.** |
| **NIDN : 0503068601** |

# DAFTAR ISI

[HALAMAN PENGESAHAN I i](#_Toc175163813)

[HALAMAN PENGESAHAN II ii](#_Toc175163814)

[DAFTAR ISI iii](#_Toc175163815)

[DAFTAR TABEL v](#_Toc175163816)

[DAFTAR GAMBAR vi](#_Toc175163817)

[DAFTAR SINGKATAN vii](#_Toc175163818)

[ABSTRAK viii](#_Toc175163819)

[ABSTRACT ix](#_Toc175163820)

[BAB I. PENDAHULUAN 1](#_Toc175163821)

[1.1. Latar Belakang 1](#_Toc175163822)

[1.2. Rumusan Masalah 4](#_Toc175163823)

[1.3. Batasan Masalah 4](#_Toc175163824)

[1.4. Tujuan Tugas Akhir 4](#_Toc175163825)

[1.4. Manfaat Tugas Akhir 5](#_Toc175163826)

[1.5. Sistematika Penulisan 5](#_Toc175163827)

[BAB II TINJAUAN PUSTAKA DAN DASAR TEORI 7](#_Toc175163835)

[2. 1. Tinjauan Pustaka 7](#_Toc175163839)

[2. 2. Dasar Teori 11](#_Toc175163840)

[2. 3. Analisis perbandingan metode 22](#_Toc175163841)

[BAB III METODE TUGAS AKHIR 25](#_Toc175163842)

[3. 1. Metode Penelitian 25](#_Toc175163848)

[3. 2. Alat dan Bahan Tugas akhir 25](#_Toc175163849)

[3.2.1. Alat Tugas akhir 25](#_Toc175163854)

[3.2.2. Bahan 26](#_Toc175163855)

[3. 3. Alur Tugas akhir 28](#_Toc175163859)

[3. 4. Alur Program Tugas akhir 34](#_Toc175163860)

[BAB IV HASIL DAN PEMBAHASAN 41](#_Toc175163861)

[4.1. Hasil 41](#_Toc175163866)

[4.2. Hasil Modified *Sequence Number* (MSN) 44](#_Toc175163867)

[4.3. Normal VANET 46](#_Toc175163872)

[4.4. *Blackhole Attack* VANET 46](#_Toc175163873)

[4.5. Perbandingan *Delay* 49](#_Toc175163874)

[4.6. Perbandingan *Throughput* 50](#_Toc175163875)

[4.7. Tinjauan Hasil Tugas akhir dibanding dengan Tugas akhir Terdahulu 51](#_Toc175163876)

[BAB V KESIMPULAN DAN SARAN 53](#_Toc175163877)

[5.1. Kesimpulan 53](#_Toc175163879)

[5.2. Saran 54](#_Toc175163880)

[DAFTAR PUSTAKA 55](#_Toc175163881)

[LAMPIRAN 58](#_Toc175163882)

# DAFTAR TABEL

[*Table 1 Standar Delay* 20](#_Toc175165991)

[*Table 2 Standar Throughput TIPHON* 21](#_Toc175165992)

[*Table 3 Parameter Simulasi VANET* 27](#_Toc175165993)

[*Table 4 Tabel data Delay* 50](#_Toc175165994)

[*Table 5 Tabel data Delay* 51](#_Toc175165995)

# DAFTAR GAMBAR

[*Gambar 1 Topology Node Routing protocol*(Upadhyaya & Shah, 2019) 9](#_Toc175154210)

[*Gambar 2 Protocol AODV(Biswas & Dasgupta, 2019)* 13](#_Toc175154211)

[*Gambar 3 Proses serangan Blackhole (M. Kumar et al., 2019)* 18](#_Toc175154212)

[*Gambar 4 Alur Pembuatan Penelitian* 28](#_Toc175154213)

[*Gambar 5 Peta jalur rute AODV* 29](#_Toc175154214)

[*Gambar 6 Lokasi Maps RS. PKU Yogyakarta* 41](#_Toc175154215)

[*Gambar 7 Simulasi menggunakan SUMO* 42](#_Toc175154216)

[*Gambar 8 Gambar AODV MSN* 43](#_Toc175154217)

[*Gambar 9 VANET Normal* 45](#_Toc175154218)

[*Gambar 10 Blackhole VANET* 45](#_Toc175154219)

[*Gambar 11 Data tabel serangan Blackhole* 46](#_Toc175154220)

[*Gambar 12 Grafik perbandingan Delay*  48](#_Toc175154221)

[*Gambar 13 Grafik Perbandingan Throughput* 49](#_Toc175154222)

# DAFTAR SINGKATAN

**A**

**AODV: Ad hoc On-Demand Distance Vector**

**M**

**MATLAB: Matrix Laboratory**

**S**

**SUMO: Simulation of Urban MObility**

**V**

**VANET: Vehicular Ad-Hoc Network**

# ABSTRAK

Dalam tugas akhir ini, dilakukan penelitian untuk menganalisis dampak serangan *Blackhole* terhadap kinerja protokol *Routing* AODV dalam jaringan Vehicular Ad-Hoc Network (VANET). Protokol AODV dimodifikasi dengan menambahkan fungsi deteksi loop untuk mengidentifikasi dan mengurangi dampak serangan *Blackhole*. Penelitian ini menggunakan perangkat lunak SUMO untuk simulasi lalu lintas dan MATLAB untuk simulasi jaringan. Hasil simulasi menunjukkan bahwa serangan *Blackhole* secara signifikan mempengaruhi kinerja jaringan, terutama dalam hal peningkatan *delay* dan penurunan *throughput*. Modifikasi yang dilakukan pada protokol AODV berhasil mengurangi dampak serangan, meskipun belum sepenuhnya menghilangkan ancaman. Penelitian ini memberikan kontribusi dalam pengembangan protokol jaringan yang lebih aman dan efektif untuk digunakan dalam lingkungan VANET yang dinamis.

Kata kunci : VANET (Vehicular Ad-Hoc Network), AODV (Ad hoc On-Demand Distance Vector), *Blackhole* Attack, Modified Sequence Number,MATLAB, SUMO.

# ABSTRACT

This thesis investigates the impact of *Blackhole* attacks on the performance of the AODV *Routing* protocol within Vehicular Ad-Hoc Networks (VANET). The AODV protocol was modified by adding a loop detection function to identify and mitigate the effects of *Blackhole* attacks. The research employed SUMO software for traffic simulation and MATLAB for network simulation. The simulation results indicate that *Blackhole* attacks significantly affect network performance, notably increasing *delay* and reducing *throughput*. The modifications to the AODV protocol successfully reduced the impact of the attack, though they did not entirely eliminate the threat. This research contributes to the development of more secure and effective *Routing* protocols for use in dynamic VANET environments.

Keyword: VANET (Vehicular Ad-Hoc Network), AODV (Ad hoc On-Demand Distance Vector), *Blackhole* Attack, Modified Sequence Number,MATLAB, SUMO.

# BAB I PENDAHULUAN

## Latar Belakang

Dalam perkembangan teknologi yang pesat, keamanan komunikasi antar kendaraan menjadi fokus utama dalam pengembangan sistem transportasi cerdas. Teknologi penting yang mendukung komunikasi ini adalah *Vehicular Ad hoc Vehicular s (*VANET), yang memungkinkan kendaraan untuk saling berkomunikasi langsung tanpa memerlukan infrastruktur tetap. Protokol *Node Routing* seperti *Ad hoc On-Demand Distance* *Vector* (AODV) digunakan untuk mengatur komunikasi dalam VANET. Meskipun AODV berperan krusial dalam menjaga kelancaran pertukaran informasi, protokol ini tetap rentan terhadap serangan yang dapat mengancam stabilitas dan keamanan jaringan, yang menjadi tantangan serius dalam penerapan sistem transportasi cerdas.

*Mobile Ad hoc Vehicular s* (MANET) adalah jaringan nirkabel yang terdiri dari *Node* -*Node* yang dapat diatur secara dinamis tanpa memerlukan infrastruktur jaringan tetap. Setiap perangkat dalam MANET berfungsi sebagai *Node* dan router, memungkinkan komunikasi langsung antar perangkat tanpa perlu pusat administrasi. Keunggulan MANET adalah kemampuannya untuk beradaptasi dengan perubahan topologi jaringan secara *real-time*, sehingga perangkat dapat bergerak bebas sambil tetap mempertahankan konektivitas. MANET digunakan dalam berbagai situasi seperti operasi militer dan bantuan bencana, di mana jaringan yang fleksibel dan tangguh sangat dibutuhkan.(Manapa et al., 2020)

Sedangakan, untuk *Vehicular Ad hoc Vehicular s* (VANET) sering disebut sebagai "jaringan di atas roda," yang dirancang untuk memungkinkan komunikasi antar *Node* kendaraan. Jaringan ini memungkinkan kendaraan untuk saling bertukar informasi penting saat berada di jalan, tanpa memerlukan infrastruktur tetap. Dalam VANET, setiap kendaraan berfungsi sebagai *Node* yang dapat mengirim dan menerima data dengan kendaraan lain di sekitarnya. Komunikasi ini bertujuan untuk meningkatkan keselamatan, efisiensi, dan pengalaman berkendara dengan memungkinkan kendaraan berbagi informasi secara *real-time*. Sebagai salah satu elemen kunci dalam pengembangan sistem transportasi cerdas, VANET memainkan peran penting dalam menghubungkan kendaraan dan mendukung berbagai aplikasi terkait lalu lintas.(Günay et al., 2021)

Perbedaan mendasar antara *MANET* dan *VANET* yaitu, *Mobile Ad hoc Vehicular s* (MANET) dan *Vehicular Ad hoc Vehicular s* (VANET) adalah jaringan nirkabel yang tidak memerlukan infrastruktur tetap, namun keduanya memiliki perbedaan dalam hal penggunaan dan mobilitas. MANET terdiri dari perangkat bergerak yang dapat berfungsi sebagai *Node* dan router, dan digunakan dalam situasi seperti operasi militer dan bantuan darurat yang membutuhkan jaringan yang sangat fleksibel. Sebaliknya, VANET dirancang khusus untuk kendaraan, memungkinkan mereka berkomunikasi saat berada di jalan untuk meningkatkan keselamatan dan efisiensi lalu lintas. MANET beradaptasi dengan topologi jaringan yang acak dan tidak terstruktur, sementara VANET menghadapi perubahan topologi yang cepat tetapi lebih teratur sesuai dengan pola pergerakan kendaraan. Selain itu, VANET berfokus pada pertukaran informasi *real-time* antar kendaraan, yang menjadikannya krusial dalam pengembangan sistem transportasi cerdas.

Penelitian ini berfokus pada analisis performa jaringan dalam *Vehicular Ad-hoc Vehicular* (VANET) dengan menggunakan protokol AODV. AODV adalah salah satu protokol *Node Routing* reaktif yang bekerja dengan cara memulai pencarian rute hanya ketika diperlukan. Ketika sebuah *Node* sumber membutuhkan rute ke *Node* tujuan, ia mengirimkan paket permintaan rute (Request) ke seluruh jaringan. *Node* yang berada di dekat tujuan akan merespons permintaan ini dengan mengirimkan paket balasan rute (Reply) kembali ke *Node* sumber.(Sindhwani et al., 2022)

*Matrix Laboratory* (MATLAB) merupakan sebuah program komputer interaktif yang berperan sebagai alat bantu yang efisien untuk berbagai jenis perhitungan. Program ini menyediakan lingkungan kerja yang nyaman, memungkinkan pengguna untuk melakukan kalkulasi yang kompleks. Salah satu fitur utama MATLAB adalah kemampuannya dalam menangani operasi yang melibatkan matriks. Dengan demikian, MATLAB sering digunakan sebagai “laboratorium” virtual untuk menyelesaikan berbagai masalah matematis dan teknik yang melibatkan matriks.(Moler & Little, 2020)

Penelitian ini memfokuskan pada analisis serangan *Blackhole*, di mana serangan tersebut menyebabkan semua paket data di VANET dibuang, sehingga mengurangi kinerja jaringan secara keseluruhan. Untuk mendeteksi serangan *Blackhole* pada protokol *Node Routing* AODV, dikembangkan solusi berupa modifikasi protokol tersebut. Inovasi ini melibatkan perubahan pada paket *route request packet* (RREQ*)* dan *route Reply packet* (RREP) dalam AODV untuk meningkatkan kinerja jaringan. *Node* akan memverifikasi *Node* tujuan sebelum meneruskan paket dengan menggunakan fitur keamanan baru yang memeriksa nomor urut paket kendaraan. Modifikasi ini bertujuan untuk memperkuat keamanan dan efektivitas protokol *Node Routing* dalam menghadapi serangan *Blackhole*.(A. Kumar et al., 2021)

Penelitian ini bertujuan untuk menganalisis kinerja jaringan AODV dalam VANET ketika mengalami serangan *Blackhole*. Uji coba dilakukan dengan memodifikasi jaringan AODV untuk mensimulasikan serangan *Blackhole*, sehingga memungkinkan pengamatan dampaknya secara mendalam. Pendeteksian serangan ini dilakukan dengan membandingkan *Source Sequence Number* dan *Route Reply Sequence Number* pada protokol *Node Routing* AODV. Untuk meningkatkan akurasi deteksi, ditambahkan fungsi *Threshold* (ambang batas) yang membantu dalam mengidentifikasi *Node* yang bersifat jahat (*malicious*). Dengan pendekatan ini, identifikasi serangan *Blackhole* pada jaringan AODV dapat dilakukan dengan lebih tepat. Penelitian ini juga mencakup evaluasi terhadap dampak serangan *Blackhole* pada *delay* dan *throughput* jaringan. Ditemukan bahwa kedua metrik ini cenderung menurun seiring dengan peningkatan jumlah serangan *Blackhole*. Penurunan tersebut terjadi karena serangan *Blackhole* dapat menyebabkan *timeout* atau pemutusan koneksi pada *Node Routing* , yang berdampak langsung pada kinerja jaringan. Selain itu, modifikasi protokol AODV yang dilakukan juga bertujuan untuk meningkatkan ketahanan jaringan terhadap serangan semacam ini. Hasil penelitian diharapkan dapat memberikan wawasan tentang bagaimana meningkatkan keamanan dan efisiensi jaringan AODV dalam lingkungan VANET

## Rumusan Masalah

Rumusan masalah pada analisis ini yaitu:

1. Bagaimana kinerja jaringan dalam VANET terpengaruh oleh serangan *Blackhole*?
2. Bagaimana cara mendeteksi serangan *Blackhole* pada protokol AODV?
3. Apa pengaruh modifikasi protokol AODV terhadap kinerja jaringan ketika menghadapi serangan *Blackhole*?
4. Bagaimana perubahan *throughput* dan *delay* apabila terjadi serangan *Blackhole* pada jaringan VANET

## Batasan Masalah

Batasan masalah pada analisis:

1. Analisis ini dilakukan menggunakan *Matrix Laboratory* (MATLAB)
2. Penelitian ini hanya akan menganalisis dampak serangan *Blackhole* terhadap kinerja jaringan AODV dalam lingkungan VANET.
3. Analisis akan difokuskan pada modifikasi protokol AODV untuk mendeteksi dan mengatasi serangan *Blackhole*.
4. Evaluasi kinerja jaringan akan dilakukan berdasarkan tiga metrik utama: simulasi, *delay* (latensi) dan *throughput*.
5. Penelitian ini akan mengimplementasikan fungsi *Threshold* sebagai bagian dari modifikasi protokol AODV untuk mendeteksi *Node* jahat.

## Tujuan Tugas Akhir

Tujuan analisis ini yaitu:

1. Untuk memahami dan mengukur dampak serangan *Blackhole* terhadap performa jaringan VANET, termasuk bagaimana serangan ini mempengaruhi *delay* dan *throughput* jaringan.
2. Untuk mengevaluasi efektivitas modifikasi protokol AODV yang telah diterapkan untuk mengidentifikasi dan mendeteksi serangan *Blackhole*

## Manfaat Tugas Akhir

Berdasarkan penjelasan di atas, berikut adalah manfaat penelitian:

1. Penelitian ini memberikan pemahaman mendalam tentang bagaimana protokol AODV beroperasi dalam lingkungan VANET, khususnya dalam situasi terjadinya serangan *Blackhole*.
2. Dengan menganalisis dampak serangan *Blackhole* terhadap *delay* dan *throughput* dalam jaringan AODV, penelitian ini memberikan wawasan tentang bagaimana serangan ini mempengaruhi kinerja jaringan.
3. Hasil penelitian dapat membantu dalam merancang strategi mitigasi yang lebih baik untuk meningkatkan performa jaringan dalam menghadapi serangan semacam ini.
4. penelitian ini memberikan kontribusi pada pengembangan sistem transportasi cerdas yang lebih baik.

## Sistematika Penulisan

Sistematika penulisan tugas akhir ini disusun dengan beberapa bab yang teratur dan terstruktur sebagai berikut:



**Bab I Pendahaluan**

Bab ini mencakup penjelasan mengenai latar belakang, rumusan masalah, batasan masalah, tujuan tugas akhir, manfaat tugas akhir, serta sistematika penulisan.

**Bab II TINJAUAN PUSTAKA DAN LANDASAN TEORI**

Bab ini menyajikan ringkasan dan analisis dari berbagai literatur yang relevan dengan topik penelitian, mencakup teori dasar tentang VANET dan *Blackhole*, serta konsep-konsep yang digunakan dalam penelitian ini.

**Bab III METODOLOGI PENELITIAN**

Bab ini akan berfokus pada penjelasan tentang metode penelitian yang digunakan untuk melakukan analisis serangan *Blackhole* pada jaringan VANET.

**Bab IV HASIL DAN PEMBAHASAN**

Bab ini membahas pendekatan yang diterapkan dalam penelitian. Selain itu, bab ini menjelaskan langkah-langkah yang diambil untuk melaksanakan penelitian tersebut. Setiap tahapan penelitian diuraikan secara rinci. Bab ini bertujuan memberikan gambaran yang jelas tentang proses penelitian analisa serangan *Blackhole* yang dilakukan.

**Bab IV KESIMPULAN DAN SARAN**

Pada bab ini akan dijelaskan kesimpulan dari sistem yang telah dibangun dan saran-saran yang diperoleh dari hasil penelitian. Saran-saran ini dapat digunakan sebagai panduan dalam pengembangan simulasi serangan *Blackhole* pada jaringan VANET.

# BAB II TINJAUAN PUSTAKA DAN DASAR TEORI



## Tinjauan Pustaka

Pada penelitian yang sudah dilakukan oleh beberapa peneliti tentang serangan *Blackhole* menjadikan menjadi salah satu bahan yang digunakan sebagai referensi dan Alasan dilakukannya penelitian ini. Pada bagian ini peneliti menggambarkan bagaimana peneliti melakukan penelitian di tempat yang berbeda dan dalam kasus yang berbeda.

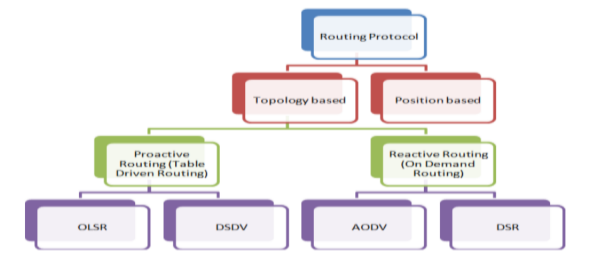
Dalam penelitian yang dilakukan oleh (Singh et al., 2019) dengan judul “*A tutorial survey on Vehicular communication state of the art, and future research directions*”. Disebutkan bahwa komunikasi antar kendaraan dilakukan melalui metode *Vehicle-to-Vehicle* (V2V). Komunikasi V2V ini bersifat sepenuhnya *Ad hoc*, memungkinkan kendaraan untuk berinteraksi tanpa infrastruktur tetap. Mode komunikasi tersebut dikenal sebagai *Vehicular Ad-hoc* *Vehicular* (VANET) atau Inter-*Vehicle* Communication (IVC). Ketika dua kendaraan berada dalam jangkauan komunikasi satu sama lain, mereka dapat berkomunikasi langsung tanpa perantara. Namun, jika kendaraan-kendaraan tersebut berada di luar jangkauan satu sama lain, mereka menggunakan metode komunikasi multi-hop. Dalam komunikasi multi-hop, data diteruskan melalui satu atau lebih kendaraan lain hingga mencapai tujuan. Pendekatan ini memastikan kontinuitas dan efisiensi dalam transfer data di antara kendaraan. Penelitian ini menyoroti pentingnya V2V dalam mendukung komunikasi yang efektif dalam lingkungan VANET. Sistem ini memainkan peran kunci dalam jaringan *ad-hoc*, khususnya dalam skenario mobilitas tinggi. Dengan ini dapat memungkin menggunakan komunikasi antar kendaraan menggunakan *Vehicle-to-Vehicle* pada penelitian ini.

Penelitian yang dilakukan oleh peneliti (Sindhwani et al., 2022) dengan judul penelitian “*Improvisation of optimization technique and AODV Routing protocol in VANET*”. Peneliti ini menggunakan *Vehicular* Adhoc *Vehicular* (VANET) untuk merancang sebuah jaringan protokol *Ad hoc* *On-Demand* *Distance* *Vector* (AODV). Pada penelitian ini VANET digunakan untuk membantu dalam menyebarkan informasi dari satu *Node* ke *Node* kendaraan lainnya. Terdapat perbedaan antara jaringan *ad-hoc* tradisional dan *Node Routing* di VANET karena topologi yang digunakan sangat dinamis dibandingkan dengan yang digunakan pada jaringan *ad-hoc* tradisional. Pada jurnal ini peneliti menjelaskan tentang struktur VANET. Penelitian ini memiliki konteks yang serupa dalam menentukan protokol jaringan. Pada penelitian ini menggunakan *Node Routing Node* ke *Node* yang lain pada jaringan VANET.

Penelitian yang dilakukan oleh(Naim & Hossain, 2019). memanfaatkan VANET sebagai basis karena kemampuannya mendukung berbagai protokol jaringan. Beberapa protokol yang didukung oleh VANET antara lain AODV, DSDV, TORA, dan DSR. Selain itu, protokol lain seperti VADD, GEOpps, GEOcast, GPSR, dan GRANTS juga dapat diimplementasikan dalam jaringan ini. Pemilihan VANET sebagai platform utama memungkinkan fleksibilitas dalam eksperimen dan simulasi jaringan. Dalam konteks penelitian ini, peneliti memfokuskan penggunaan pada protokol AODV. AODV, atau *Ad-hoc* *On-Demand* *Distance* *Vector*, dipilih karena karakteristiknya yang sesuai untuk jaringan *ad-hoc*. Penggunaan protokol ini diharapkan dapat memberikan hasil yang optimal dalam skenario yang diteliti. Penelitian ini digunakan untuk memilih VANET dikarena dapat mendukung protokol jaringan yang akan digunakan pada penelitian ini. Pada penelitian ini protokol jaringan yang digunakan yaitu AODV.

Referensi yang digunakan dalam penelitian ini termasuk judul "*AODV-DSR Hybrid Reactive Routing Protocol and its Generalization for Mobile Ad-hoc Vehicular s*". (Biswas & Dasgupta, 2019) Artikel ini membahas penggabungan protokol *Routing* AODV dan DSR dalam jaringan *ad-hoc* bergerak, serta generalisasinya untuk berbagai skenario jaringan. Dalam artikel tersebut, dijelaskan bahwa protokol AODV memperkenalkan paket baru yang dikenal sebagai RREQ untuk mentransfer data secara efisien. Penggunaan Probabilitas Bayesian dalam protokol ini memungkinkan perhitungan jumlah *Node* yang terlibat antara sumber dan tujuan dengan akurat. Dengan cara ini, kebutuhan untuk mentransmisikan ulang paket dapat dikurangi secara signifikan. Selain itu, metode ini membantu menurunkan volume lalu lintas secara keseluruhan, mengurangi beban jaringan. Pengurangan kebutuhan transmisi ulang paket juga berkontribusi pada penurunan konsumsi energi dalam jaringan. Artikel ini menjelaskan bagaimana pendekatan ini mengoptimalkan perutean, mengurangi biaya perutean hingga tingkat minimum. Generalisasi protokol ini untuk berbagai skenario jaringan memastikan fleksibilitas dan efisiensi dalam aplikasi jaringan *ad-hoc* yang berbeda. Ini memberikan wawasan berharga tentang cara meningkatkan performa dan efisiensi *Routing* dalam konteks mobilitas tinggi.

Pada penelitian yang dilakukan oleh peniliti dengan judul “*Effect On Aodv Routing Protocol Under Blackhole Attack In VANET*” pada penelitian dengan judul *Attack VANET* pada peneltian ini untuk efisiensi dari *Node Routing* protokol terdapat beberapa yang diusulkan oleh penulis, penulis menyertakan topology *Node Routing protocol* untuk menginformasikan tentang penulisannya. *Node Routing protocol* berbasis topologi ini dapat dikategorikan menjadi 2 sub kategori, *Poractive Node Routing* dan *Reactive Node Routing* .



*Gambar 1 Topology Node Routing protocol*(Upadhyaya & Shah, 2019)

Pada penelitian ini penulis menggunakan AODV *protocol Node Routing* . AODV *protocol Node Routing* merupakan *reactive Node Routing protocol*. pada implementasi nya AODV digunakan untuk memanajemen *route* dengan menggunakan mekanisme <RREQ>RREP>. (Upadhyaya & Shah, 2019)

Dalam penelitian berjudul "*Design and Implementation of VANET in Ad hoc Vehicular using MATLAB*,"(Kuthe et al., 2022) penulis memanfaatkan bahasa pemrograman MATLAB serta lingkungan komputasi numerik *multiparadigma* untuk membangun dan menganalisis sistem VANET. MATLAB menyediakan kemampuan untuk membuat antarmuka pengguna kustom dan mengintegrasikan dengan program lain, sambil mendukung kalkulasi matriks dan implementasi algoritme. Dengan MATLAB, peneliti dapat mengembangkan model mobilitas VANET dan memperbarui posisi kendaraan (*Node* ) setiap detik. Penelitian ini menekankan bahwa penggunaan MATLAB dalam simulasi dapat meningkatkan akurasi dan efektivitas sistem, khususnya dalam konteks protokol AODV dan penanganan serangan *Blackhole*. Studi ini relevan dengan kebutuhan penelitian ini karena kabutuhan yang peneliti ini memanfaatkan MATLAB sebagai bahasa pemograman dan lingkungan komputasi numerik yang sangat sesuai dengan kebutuhan analisis dan pengembangan sistem VANET.

Dalam penelitian yang dilakukan oleh (Oberoi, 2020) dengan judul “*Enhancement of QoS in Security Algorithm for Blackhole Attack in VANET*” di jelaskan bahwa Serangan lubang hitam adalah salah satu bentuk dari serangan *Denial of Service* yang bertujuan untuk merusak komunikasi antar *Node* atau kendaraan dalam jaringan VANET. Dalam konteks protokol *Routing* AODV, serangan ini bekerja dengan cara mengganggu proses pencarian jalur. Ketika sebuah *Node* sumber atau kendaraan menginisiasi pencarian jalur dengan mengirimkan pesan Route Request (RREQ), *Node* jahat memanipulasi sistem dengan memberikan informasi jalur yang salah. *Node* jahat ini secara curang mengiklankan bahwa mereka memiliki jalur tercepat atau paling optimal ke tujuan dengan mengirimkan Route *Reply* (RREP) palsu yang menunjukkan jumlah hop yang paling sedikit atau nomor urut (*Sequence Number*) tertinggi. Hal ini membuat *Node* sumber percaya bahwa jalur yang diberikan oleh *Node* jahat adalah yang terbaik, sehingga data yang dikirimkan akan diarahkan melalui *Node* tersebut. Akibatnya, komunikasi terganggu karena data yang masuk ke *Node* jahat tidak akan diteruskan, melainkan dihancurkan, sehingga layanan jaringan terputus. Penelitian ini secara mendalam menganalisis ancaman serangan *Blackhole* dalam jaringan VANET, yang merupakan fokus penting dalam penelitian Anda. Serangan *Blackhole* merupakan salah satu bentuk *serangan Denial of Service* (DoS) yang sangat merusak, terutama dalam konteks protokol *Routing* AODV yang Anda teliti. Penelitian ini menjelaskan mekanisme serangan *Blackhole*, di mana *Node* jahat memberikan informasi jalur yang salah untuk mengelabui *Node* sumber, sehingga data dikirimkan melalui jalur yang dikuasai oleh penyerang. Hal ini menyebabkan data tidak diteruskan ke tujuan dan malah dihancurkan, mengakibatkan gangguan serius pada komunikasi jaringan.

Dalam penelitian berjudul “Comparative Analysis *of Routing* Algorithms in SUMO for VANET” yang dilakukan oleh (nama peneliti), SUMO digunakan sebagai alat simulasi untuk jaringan kendaraan VANET. SUMO mendukung berbagai algoritma pencarian rute yang penting untuk memodelkan skenario jalan raya dan perkotaan dengan presisi, sehingga memudahkan peneliti dalam mengevaluasi kinerja berbagai strategi *Routing* . Karena VANET memiliki karakteristik unik, seperti topologi yang cepat berubah, komunikasi yang sering terputus, serta kemampuan komputasi dan penyimpanan yang luas, SUMO memungkinkan simulasi yang realistis dan sesuai dengan situasi nyata. Selain itu, biaya menjadi pertimbangan penting dalam penerapan sistem transportasi cerdas, dan SUMO menyediakan platform yang efisien dan ekonomis untuk penelitian ini, menjadikannya pilihan yang menarik bagi peneliti yang ingin menguji dan mengoptimalkan strategi *Routing* dalam konteks VANET.

## Dasar Teori

* + 1. ***VANET***

*Vehicular Ad-hoc* *Vehicular* (VANET) merupakan jenis jaringan yang memungkinkan kendaraan berkomunikasi antara satu dengan yang lainnya, serta dengan infrastruktur jalan. Jaringan ini dibentuk dengan menggagas konsep suatu jaringan kendaraan untuk keperluan tertentu atau dalam situasi khusus. Sebagai hasilnya, VANET telah terbukti sebagai jaringan yang dapat diandalkan yang digunakan oleh kendaraan untuk berkomunikasi di jalan raya atau di lingkungan perkotaan.(Kugali & Kadadevar, n.d.)

Dalam VANET terdapat berbagai model komunikasi data telah dikembangkan yang mengusulkan integrasi beragam jenis infrastruktur kendaraan. Bentuk komunikasi ini mencakup *Vehicle-to-Vehicle* (V2V), *Vehicle-to-Infrastructure* (V2I), serta kombinasi dari keduanya, V2V dan V2I.

1. *Vehicle-to-Vehicle (V2V):* V2V adalah jenis komunikasi langsung antar kendaraan yang memungkinkan mereka bertukar informasi secara *real-time*. Misalnya, kendaraan dapat saling berbagi data mengenai kondisi jalan, kecepatan, atau bahaya di depan, yang dapat meningkatkan keselamatan dan efisiensi lalu lintas.
2. *Vehicle-to-Infrastructure (V2I):* V2I melibatkan komunikasi antara kendaraan dan infrastruktur jalan raya, seperti lampu lalu lintas, rambu-rambu, atau pusat pengendalian lalu lintas. Komunikasi ini memungkinkan kendaraan menerima informasi yang dapat membantu dalam navigasi dan pengambilan keputusan, seperti perubahan lampu lalu lintas atau kemacetan di depan.
   * 1. ***AODV***

Protokol *Node Routing Ad hoc* *On-Demand* *Distance* *Vector* (AODV) merupakan hasil kombinasi dari dua protokol *Node Routing* lainnya, yaitu Dynamic *Source* *Node Routing* (DSR) dan *Destination Sequence*d *Distance* *Vector* (DSDV). AODV adalah jenis protokol *Node Routing* on demand yang memanfaatkan mekanisme dari DSR, termasuk Route Discovery dan Route Maintenance, serta melibatkan hop-by-hop *Node Routing* , periodic beacon, dan menggunakan *Sequence*d *Number*s yang juga ditemukan pada DSDV.(Assidiq & Nurcahyani, n.d.).

Penelitian ini berfokus pada bagaimana kinerja protokol *Node Routing* AODV mempengaruhi konsumsi energi dalam jaringan VANET, dengan menggunakan perangkat lunak MATLAB sebagai alat analisis. Dalam konteks ini, AODV digunakan untuk mengatur rute komunikasi antar kendaraan dalam VANET, yang memungkinkan pertukaran informasi secara efisien. Namun, penelitian ini juga mempertimbangkan skenario di mana jaringan AODV terpapar oleh serangan *Blackhole*, di mana *Node* jahat mengelabui *Node* sumber dengan memberikan informasi rute yang salah, sehingga menyebabkan data hilang. Penelitian ini bertujuan untuk menganalisis dan membandingkan perbedaan kinerja antara AODV yang beroperasi dalam kondisi normal dan saat mengalami serangan *Blackhole*. Perbandingan ini mencakup evaluasi terhadap konsumsi energi, yang menjadi salah satu indikator penting dalam menilai efisiensi jaringan VANET. Hasil dari penelitian ini diharapkan dapat memberikan wawasan mengenai dampak serangan *Blackhole* terhadap performa jaringan, khususnya dalam hal penggunaan energi.

A diagram of a diagram

Description automatically generated

Gambar 2 Protocol AODV(Biswas & Dasgupta, 2019)

Pada gambar 2, dijelaskan Terdapat beberapa langkah dalam proses pencarian rute pada protokol komunikasi AODV. Langkah pertama adalah *Node* sumber mengirimkan paket *RouteRequest* (RREQ) ke *Node* tetangganya. Selanjutnya, jika sebuah *Node* memiliki informasi tentang rute menuju *Node* tujuan, *Node* tersebut akan mengirimkan paket RREP kembali ke *Node* sumber melalui jalur yang terbentuk oleh RREQ. Apabila terdeteksi adanya kerusakan pada rute komunikasi, mekanisme perbaikan rute akan mengirimkan paket *routeerror* (RERR) ke salah satu *Node* di sepanjang jalur, dan *Node* sumber akan mengirimkan kembali paket RREQ. *Node* yang menerima RREQ akan memberikan informasi tentang masa aktif rute. Informasi perutean antara *Node* sumber dan *Node* di jalur akan dihapus jika masa aktif rute telah berakhir. Pada gambar 1, merupakan gambar struktur AODV. Penelitian ini digunakan sebagai referensi protokol *Routing* AODV untuk mentransmisikan paket-paket paket RREQ dan RREP. *Node* A adalah titik awal atau sumber dari data yang akan dikirimkan ke tujuan. *Node* H adalah titik tujuan akhir untuk data yang dikirim dari *Node* sumber.Dalam protokol AODV, *Node* sumber (A) akan memulai proses pengiriman dengan mengirimkan permintaan rute (RREQ) untuk mencari jalur menuju *Node* tujuan (H). Permintaan ini akan melewati *Node* -*Node* menengah (seperti B, E, D, G) sampai mencapai *Node* tujuan (H).Setelah *Node* tujuan (H) menerima permintaan rute, ia akan mengirimkan balasan rute (RREP) kembali ke *Node* sumber (A) melalui jalur yang sama, memastikan jalur komunikasi yang efisien.(Assidiq & Nurcahyani, n.d.)

* + 1. ***Modified Sequence Number***

Modified *Sequence Number* (MSN) adalah konsep yang berhubungan dengan perubahan atau penyesuaian dalam protokol *Node Routing* AODV. MSN digunakan untuk mendeteksi *Node Blackhole* yang berbahaya dengan mempertimbangkan nomor urut yang diterima dari *Node* berikutnya. Ketika *Node* sumber menerima pesan balasan rute (RREP) dengan nomor urut yang mencurigakan, *Node* tersebut akan mengirim ulang permintaan rute (RREQ) dengan nomor urut yang telah dimodifikasi untuk mencocokkan nomor urut dari RREP yang diterima sebelumnya. Jika *Node* yang menjawab adalah *Node Blackhole*, ia akan mengirim RREP dengan nomor urut yang lebih tinggi lagi. Penerimaan RREP dengan nomor urut yang lebih tinggi setelah siaran ulang akan mengonfirmasi bahwa *Node* tersebut berbahaya, dan rute melalui *Node* tersebut harus dihindari. Dengan cara ini, risiko paket yang hilang dapat dikurangi, yang pada akhirnya meningkatkan kinerja jaringan.(Shrestha et al., 2020)

* + 1. *Blackhole*

*Blackhole* merupakan manipulasi *Node* kendaraan untuk memblokir semua paket yang diterimanya, sehingga tidak diteruskan ke *Node* kendaraan berikutnya menuju tujuan. Serangan ini memiliki dampak yang signifikan terhadap stabilitas dan kinerja jaringan. Tiga konsekuensi utama dari serangan ini adalah:

1. gangguan pada penerusan paket antar *Node* kendaraan,
2. hambatan dalam penerimaan pesan oleh *Node* tujuan
3. potensi pembuangan semua paket yang diterima oleh *Node* penerima.

Dalam protokol AODV, serangan *Blackhole* dapat menyebabkan pemutusan komunikasi dalam jaringan karena paket data tidak pernah mencapai tujuan yang diinginkan. Ini mengurangi keandalan dan efisiensi komunikasi antar kendaraan, yang merupakan salah satu fungsi utama VANET. Untuk mengatasi atau mendeteksi serangan ini, diperlukan modifikasi atau penguatan pada protokol *Node Routing* yang digunakan, seperti AODV, sehingga dapat mengenali dan menghindari *Node* jahat yang terlibat dalam serangan *Blackhole*.

Serangan *Blackhole* dalam jaringan VANET yang menggunakan protokol AODV dapat dijelaskan melalui beberapa langkah berikut:

1. *Proses Penemuan Rute:* Ketika sebuah *Node* sumber ingin mengirimkan data ke tujuan, ia memulai proses penemuan rute dengan mengirimkan Route Request (RREQ) ke seluruh jaringan. RREQ ini akan diteruskan oleh *Node* -*Node* yang menerima permintaan tersebut hingga mencapai *Node* tujuan atau *Node* yang dapat memberikan rute ke tujuan.
2. *Penawaran Jalur Palsu:* *Node Blackhole*, yang terletak di jalur yang dilalui RREQ, mengirimkan Route *Reply* (RREP) palsu ke *Node* sumber. *Node Blackhole* ini berpura-pura memiliki jalur yang lebih pendek atau lebih baik menuju tujuan dengan menunjukkan nomor urut yang tinggi atau jumlah hop yang sedikit.
3. *Pengalihan dan Pemblokiran Data***:** *Node* sumber, yang menerima RREP palsu, akan memilih jalur yang diklaim sebagai yang terbaik dan mulai mengirimkan data melalui jalur tersebut. Data yang dikirimkan ke *Node Blackhole* tidak diteruskan tetapi dibuang, menyebabkan kehilangan data dan gangguan dalam komunikasi.

Serangan *Blackhole* memiliki beberapa dampak signifikan pada jaringan, antara lain:

1. Gangguan pada Penerusan Paket: Paket data yang dikirimkan melalui jalur yang mengandung *Node Blackhole* tidak diteruskan ke *Node* berikutnya. Hal ini menyebabkan gangguan pada aliran paket dan mengakibatkan data yang dikirim tidak sampai ke tujuan.
2. Hambatan dalam Penerimaan Pesan oleh *Node* Tujuan: *Node* tujuan tidak menerima data yang dikirimkan karena paket-paket tersebut terhenti di *Node Blackhole*. Ini mengakibatkan kegagalan dalam pengiriman pesan yang seharusnya diterima oleh *Node* tujuan.
3. Potensi Pembuangan Paket: *Node Blackhole* membuang semua paket yang diterimanya tanpa meneruskannya. Hal ini mengurangi efisiensi dan keandalan jaringan, karena data yang penting tidak sampai ke *Node* tujuan yang diinginkan.

Dalam protokol AODV, serangan *Blackhole* dapat mengakibatkan:

* Pemutusan Komunikasi: Karena paket data tidak pernah mencapai tujuan yang diinginkan, komunikasi antar *Node* terganggu. Ini mengurangi keandalan dan efisiensi jaringan VANET, yang seharusnya memungkinkan pertukaran informasi antar kendaraan dengan lancar.
* Penurunan Kinerja Jaringan: Kehilangan paket dan gangguan komunikasi dapat menyebabkan penurunan keseluruhan kinerja jaringan, meningkatkan latensi, dan mengurangi *throughput*.

Untuk mengatasi atau mendeteksi serangan *Blackhole* dalam jaringan VANET yang menggunakan AODV, beberapa strategi dapat diterapkan, seperti:

* **Modifikasi Protokol *Routing* :** Melakukan perubahan pada mekanisme deteksi dan pengelolaan rute dalam AODV untuk mengenali dan menghindari *Node* jahat.
* **Penguatan Keamanan:** Mengimplementasikan teknik keamanan tambahan seperti *otentikasi* dan enkripsi untuk memastikan keaslian informasi rute.
* **Metode Deteksi Anomali:** Menggunakan algoritma untuk mendeteksi perilaku yang tidak biasa atau tidak konsisten dalam rute dan mengidentifikasi potensi *Node Blackhole*.

Proses terjadinya serangan *Blackhole* digambarkan sebagai berikut :

A diagram of a flowchart

Description automatically generated

Gambar 3 Proses serangan Blackhole (M. Kumar et al., 2019)

Pada gambar 4 merupakan ilustrasi *Blackhole Attack* pada jaringan VANET. Serangan *Blackhole* dapat diilustrasikan dengan menunjukkan bahwa *Node* jahat merupakan indikasi utama dari serangan ini. *Node* jahat cenderung menarik pengirim dengan meyakinkan pengirim tentang adanya jalur yang baru dan lebih pendek. Ada dua metode yang dapat digunakan oleh *Node* jahat untuk melakukan tugas ini: pertama, dengan mengirimkan nomor urut tujuan yang tetap, dan kedua, dengan mengirimkan nomor urut tujuan yang berubah-ubah. *Node* sumber mengirimkan permintaan rute untuk menemukan jalur menuju *Node* tujuan. *Blackhole* dengan cepat merespons permintaan rute tersebut, mengklaim memiliki jalur langsung ke *Node* tujuan, meskipun kenyataannya tidak. *Node* sumber kemudian mengirimkan semua paket data melalui *Blackhole*, dengan asumsi bahwa itu adalah jalur tercepat atau paling optimal. *Blackhole* menerima semua paket dari *Node* sumber dan kemudian membuangnya secara langsung.(M. Kumar et al., 2019)

* + 1. *Delay*

*Delay* adalah waktu yang dibutuhkan untuk mentransfer data dari satu komputer ke komputer lain dalam suatu jaringan. Penyebab utama *delay* dalam transmisi paket termasuk antrean yang panjang sepanjang rute. *Delay* juga dapat terjadi ketika paket memilih rute alternatif untuk menghindari kemacetan di jalur utama. Jarak antara pengirim dan penerima turut berkontribusi pada peningkatan *delay* . Jenis media fisik yang digunakan, seperti kabel atau gelombang radio, dapat mempengaruhi kecepatan dan stabilitas transmisi, yang kemudian menambah *delay* . Selain itu, *delay* bisa meningkat karena kongesti atau kepadatan lalu lintas data dalam jaringan. Jika banyak data yang dikirimkan secara bersamaan, paket-paket data mungkin perlu menunggu lebih lama sebelum diproses dan dikirim. Waktu pemrosesan di setiap *Node* juga dapat memperpanjang *delay* , terutama jika perangkat keras atau perangkat lunak yang digunakan kurang optimal. Semua faktor ini berkontribusi pada total waktu yang dibutuhkan untuk mengirimkan paket data. Dalam beberapa kasus, *delay* yang tinggi bisa menurunkan kinerja jaringan secara keseluruhan, terutama pada aplikasi yang memerlukan transmisi data secara *real-time*. Untuk menghitung *delay* pada paket yang ditransmisikan, dapat dilakukan dengan membagi panjang paket (dalam bit) dengan *link* *bandwidth* (dalam bit/s). Sedangkan untuk menghitung rata-rata *delay* , digunakan rumus tertentu sebagaimana yang tertuang dalam Persamaan (Nazif et al., n.d.).

|  |  |
| --- | --- |
|  | (1) |

Kategori dan besar *delay* berdasarkan standar versi TIPHON.

|  |  |  |
| --- | --- | --- |
| Kategori | Besar *Delay* | Indeks |
| Sangat bagus | <150 ms | 4 |
| Bagus | 150 – 300 ms | 3 |
| Sedang | 300-450 ms | 2 |
| Buruk | >450 ms | 1 |

Table 1 Standar Delay

* + 1. *Throughput*

*Throughput* adalah ukuran *bandwidth* aktual yang tercatat dalam jangka waktu tertentu selama proses transmisi data. Meskipun satuannya sama dengan *bandwidth*, yaitu bits per *second* (*bps*), *throughput* memberikan gambaran lebih realistis mengenai *bandwidth* yang sebenarnya tersedia pada waktu tertentu dan dalam kondisi jaringan tertentu yang digunakan untuk mengunduh *File* dengan ukuran tertentu. *Throughput* mencerminkan jumlah total paket yang berhasil diterima di tujuan dalam jangka waktu tertentu, dibagi dengan durasi interval waktu tersebut. Nilai *Throughput* dapat dihitung menggunakan rumus tertentu.

|  |  |
| --- | --- |
|  | (2) |

Ada juga kategori dan indeks *throughput* berdasarkan standar TIPHON yang dapat digunakan untuk mengukur kinerja jaringan.

|  |  |  |
| --- | --- | --- |
| Kategori | Besar *Throughput* | Indeks |
| Sangat bagus | 100 bps | 4 |
| Bagus | 75 bps | 3 |
| Sedang | 50 bps | 2 |
| Buruk | 25 bps | 1 |

Table 2 Standar Throughput TIPHON

\

## Analisis perbandingan metode

Analisis yang akan dilakukan dalam tugas akhir ini adalah analisis kuantitatif. Metode kuantitatif dipilih untuk mendapatkan pemahaman yang lebih terukur mengenai dampak jumlah kendaraan terhadap kinerja jaringan, dengan menggunakan studi kasus di wilayah RS. PKU Muhammadiyah Yogyakarta. Pemilihan metode ini didasarkan pada kebutuhan untuk mengumpulkan dan menganalisis data numerik yang dapat memberikan informasi yang lebih objektif.

Pendekatan kuantitatif adalah metode penelitian yang mengandalkan data numerik dan prinsip-prinsip ilmiah untuk menjawab hipotesis. Dalam pendekatan ini, informasi yang dikumpulkan berupa angka dan statistik. Data-data tersebut digunakan untuk menganalisis fenomena secara objektif. Pendekatan ini memfokuskan pada pengukuran yang tepat dan analisis yang dapat menghasilkan temuan yang terukur.(Waruwu, 2023)

Metode pelaksanaan tugas akhir ini mencakup penggunaan dua alat utama: *Simulation of Urban MObility* (SUMO) dan *Vehicular Ad-hoc* *Vehicular* (VANET). SUMO adalah simulator lalu lintas yang memungkinkan pemodelan dan analisis pergerakan kendaraan di area perkotaan. Di sisi lain, VANET adalah jaringan *Ad hoc* yang memungkinkan kendaraan untuk saling berkomunikasi serta berinteraksi dengan infrastruktur jaringan.

Dengan menerapkan metode kuantitatif ini, diharapkan tugas akhir dapat memberikan pemahaman yang lebih mendalam mengenai dampak jumlah kendaraan terhadap kinerja jaringan di wilayah RS. PKU Muhammadiyah Yogyakarta. Metode kuantitatif menawarkan keuntungan berupa data yang dapat diukur dan dianalisis secara statistik, yang memungkinkan peneliti untuk memperoleh wawasan yang lebih objektif dan dapat dipertanggungjawabkan tentang hubungan antara jumlah kendaraan dan kinerja jaringan. Selain itu, penelitian ini juga mencakup analisis AODV dalam konteks VANET untuk menambah pemahaman tentang performa jaringan dalam situasi yang dinamis.

Berikut adalah kelebihan dan kekurangan metode kuantitatif dalam konteks penelitian, beserta penjelasan spesifik untuk penelitian ini:

Kelebihan metode kuantitatif

1. **Objektivitas**: Metode kuantitatif menghasilkan data yang objektif dan terukur karena menggunakan angka dan analisis statistik. Ini memungkinkan peneliti untuk menarik kesimpulan yang kuat berdasarkan fakta dan bukti yang dapat diperiksa dan diverifikasi.
2. **Replikasi**: Metode ini memudahkan peneliti lain untuk mengulang studi dengan metode yang sama untuk memverifikasi hasilnya. Dengan demikian, kevalidan dan kepercayaan terhadap temuan penelitian dapat ditingkatkan.
3. **Generalisasi**: Penelitian kuantitatif memungkinkan hasil penelitian untuk digeneralisasikan ke populasi yang lebih luas. Data yang dikumpulkan dapat mencerminkan kondisi yang lebih umum dan memberikan wawasan yang lebih luas tentang efek variabel yang diteliti, seperti jumlah kendaraan terhadap kinerja jaringan.

Kekurangan Metode Kuantitatif:

1. **Keterbatasan dalam Pengumpulan Data**: Metode kuantitatif bergantung pada data numerik yang dapat diukur. Terkadang, pengumpulan data yang diperlukan dapat sulit dilakukan, atau data yang tersedia mungkin tidak lengkap atau kurang akurat.
2. **Pengabaian Aspek Kualitatif**: Metode ini cenderung fokus pada angka dan hasil yang dapat diukur, yang mungkin mengabaikan aspek kualitatif yang penting. Informasi seperti preferensi pengguna, persepsi, atau faktor sosial mungkin tidak sepenuhnya tercakup dalam analisis kuantitatif.

Pengumpulan dan Analisis Data Numerik: Metode kuantitatif memungkinkan penelitian ini untuk mengumpulkan dan menganalisis data numerik mengenai jumlah kendaraan dan kinerja jaringan secara terukur.

Dalam tugas akhir penelitian ini yang menggunakan MATLAB untuk menganalisis VANET AODV dengan *Blackhole Attack*, metode kuantitatif dipilih karena:

1. **Pengumpulan dan Analisis Data Numerik**: Metode kuantitatif memungkinkan penelitian untuk mengumpulkan dan menganalisis data numerik tentang jumlah kendaraan dan kinerja jaringan dengan cara yang terukur.
2. **Simulasi Menggunakan SUMO dan VANET**: Dengan SUMO untuk simulasi lalu lintas dan VANET untuk model jaringan, penelitian dapat menghasilkan data yang dibutuhkan untuk analisis kuantitatif. Ini memungkinkan pemodelan kondisi lalu lintas secara spesifik dan analisis mendalam dari hasil simulasi.
3. **Pemahaman yang Lebih Mendalam**: Metode kuantitatif diharapkan memberikan pemahaman yang lebih mendalam mengenai dampak jumlah kendaraan terhadap kinerja jaringan di wilayah RS. PKU Muhammadiyah Yogyakarta, serta efek AODV dan serangan *Blackhole* pada jaringan *Vehicular*.

# BAB III METODE TUGAS AKHIR



## Metode Penelitian

Metode penelitian merupakan aspek fundamental dalam pelaksanaan sebuah studi, berfungsi sebagai dasar untuk mencapai tujuan penelitian secara efektif. Dalam tugas akhir ini, pendekatan yang digunakan adalah metode kuantitatif untuk mengevaluasi dampak jumlah kendaraan terhadap kinerja jaringan di Wilayah RS. PKU Muhammadiyah Yogyakarta. Penelitian ini memanfaatkan SUMO (*Simulation of Urban MObility*) dan VANET (*Vehicular Ad-hoc* *Vehicular* ) untuk mengembangkan model simulasi yang mencerminkan kondisi lalu lintas di area tersebut. Data yang diperoleh dari simulasi akan dianalisis secara kuantitatif untuk memberikan pemahaman yang terukur dan objektif tentang bagaimana variasi jumlah kendaraan mempengaruhi performa jaringan. Penjelasan berikut akan menguraikan dengan lebih rinci tentang alat dan bahan yang digunakan serta tahapan yang akan diikuti dalam penelitian ini.

## Alat dan Bahan Tugas akhir



### Alat Tugas akhir

Alat yang digunakan berupa perangkat keras (*hardware*) dan perangkat lunak (*software*).

1. Perangkat keras (*Hardware*)
2. Interl® Core™ i7 9750-H *processor*,
3. 8.00 GB RAM,
4. 64-bit *Operating System,*
5. *Windows 11 pro*.
6. Perangkat lunak (Soft*ware*)
   1. *Matrix Lab (MATLAB),*
   2. *Windows* *Operating* *System*,
   3. Microsoft .Excel

### Bahan

Data survei lapangan merupakan informasi yang dikumpulkan secara langsung dari lapangan untuk memberikan gambaran akurat mengenai kondisi nyata di area penelitian. Dalam konteks tugas akhir ini, bahan survei lapangan meliputi:

1. Data Jumlah Kendaraan:
2. Sumber Data

Data ini diperoleh dari pengamatan langsung di area sekitar RS. PKU Muhammadiyah Yogyakarta.

1. Rincian Data

Termasuk total jumlah kendaraan yang lewat, frekuensi kedatangan kendaraan, serta jam-jam sibuk dan sepi.

1. Tujuan

Memahami volume lalu lintas dan pola pergerakan kendaraan di wilayah tersebut, yang akan membantu dalam analisis bagaimana pergerakan kendaraan memengaruhi performa dan efisiensi jaringan VANET.

1. Data sekunder

Data sekunder adalah informasi yang diperoleh dari sumber yang sudah ada sebelumnya dan relevan untuk analisis. Dalam konteks ini, data sekunder meliputi: Data Lalu Lintas: Jalur dan Rute: Informasi mengenai jalur-jalur utama dan alternatif di sekitar RS. PKU Muhammadiyah Yogyakarta, termasuk tata letak jalan, rute utama, dan area yang sering macet. Lama Waktu Lalu Lintas: Data mengenai durasi perjalanan rata-rata di berbagai jalur, waktu puncak, dan pergerakan lalu lintas yang mungkin menunjukkan pola atau masalah yang dapat mempengaruhi performa sistem VANET.

|  |  |
| --- | --- |
| **Parameter** | **Value** |
| Jumlah Kendaraan | 100 per *lane* |
| Jenis Kendaraan | Mobil, Taxi |
| Kecepatan Maksimum | 140 km/jam |
| *Traffic Light* | Ada |
| Protokol Komunikasi | V2I, V2V |
| Model Propagasi | *Free Space Path Loss* (FSPL) |
| *Path Loss Model* | *Free Space Path Loss* (FSPL) |
| Durasi Simulasi | 120 s |
| Frekuensi Operasi | 5.9 Ghz |
| Algoritma Pemilihan Jalur | *Ad hoc On-Demand Distance Vector* (AODV), *Dynamic Source Routing* (DSR) |

Table 3 Parameter Simulasi VANET



## Alur Tugas akhir

Pada bagian ini, dijelaskan mengenai langkah-langkah yang diambil oleh penulis dalam menyelesaikan penelitian ini. Berikut pembahasan alur penelitian melalui beberapa alur.

A diagram of a program

Description automatically generated

Gambar 4 Alur Pembuatan Penelitian

* + 1. Tinjauan Pustaka

Pada tahap tinjauan pustaka, penulis melakukan observasi terkait penelitian sebelumnya dengan cara mempelajari dan memahami metode-metode pada *Ad hoc On-Demand Distance Vector Node Routing* (AODV) dan serangan *Blackhole.* Penulis dapat mempelajari dengan mendalam tentang AODV dan serangan *Blackhole* pada literatur penelitian sebelumnya.

* + 1. Pengumpulan Data

Pada tahap pengumpulan data, data dikumpulkan dari implementasi jalur jalan yang berada di RS. PKU Yogyakarta. Data ini diperoleh dengan menggunakan citra peta yang diimplementasikan sebagai desain jalur AODV untuk simulasi jaringan VANET. Implementasi ini dilakukan untuk memodelkan lalu lintas kendaraan di area tersebut, yang kemudian digunakan sebagai dasar untuk menganalisis kinerja protokol *Routing* AODV dalam kondisi normal dan saat terjadi serangan *Blackhole*.

Data yang dikumpulkan meliputi posisi *Node* (koordinat kendaraan), waktu kedatangan dan kepergian kendaraan, serta berbagai parameter jaringan yang relevan seperti jumlah kendaraan, kecepatan kendaraan, dan konfigurasi lalu lintas.



Gambar 5 Peta jalur rute AODV

* + 1. Implementesi AODV *Node Routing* Protocol

Pada tahap ini, merupakan tahap yang dilakukan ketika penulis sudah mengumpulkan data dengan valid. Penulis akan mengimplementasi kan simulasi menggunakan data yang sudah dikumpulkan sebelumnya.

* + 1. Membuat Program Simulasi

Tahap ini, merupakan tahap dari implementasi data yang sudah dikumpulkan untuk diterjemahkan menggunakan program simulasi AODV. Pada tahap ini penulis melakukan *run* pada program, apakah program sudah sesuai dengan data yang sudah ditentukan sebelumnya atau belum?. Pada simulasi ini data-data yang dikumpulkan di implementasikan dengan *Node* -*Node* kendaraan yang melintas pada jalur tersebut. Pada tahap ini peneliti membuat jaringan koneksi pada *Node* berdasarkan data yang telah dikumpulkan

* + 1. Memberi Serangan

Setelah peneliti menyelesaikan pembuatan program simulasi AODV dengan benar, langkah berikutnya adalah menerapkan serangan *Blackhole* pada *Node* yang telah ditentukan. Dalam program ini, peneliti menggunakan perbandingan antara *Source* *Sequence Number* (SSN) dan *Reply Sequence Number* (RREPSN) untuk mendeteksi serangan *Blackhole*. Peneliti menentukan *Threshold* untuk menetapkan ambang batas jumlah *Sequence*. Untuk menentukan *Threshold* menggunakan SSN teritnggi sebagai ambang batas *Threshold*. Rumus yang digunakan untuk mendeteksi serangan *Blackhole* adalah:

|  |  |
| --- | --- |
|  | (3) |

A diagram of a process

Description automatically generated

Gambar 6 Blackhole attack pada AODV

Pada gambar.., menjelaskan jika, *Node* D menerima permintaan RREQ dari *Node* S. Kemudian, *Node* S mengirimkan balasan RREP melalui *Node* I. Ketika *Node* M atau *Node* *Blackhole* menerima pesan RREQ, *Node* M akan mengubah *Sequence Number* menjadi angka acak. Karena protokol *Routing* AODV memilih rute dengan jarak terpendek, *Node* S akan memilih rute yang melalui *Node* M. Akibatnya, rute tersebut, yang mengandung *Node* blackhole, akan dipilih sebagai jalur transmisi data. Sebagai hasilnya, data yang dikirimkan melalui rute ini bisa mengalami kegagalan atau kerugian karena adanya *Node* *Blackhole* dalam jalur tersebut.

* + 1. Analisa Hasil Simulasi

Untuk menganalisis hasil simulasi peniliti menggunakan *delay* dan *throughput*.

1. *Delay*

Untuk pengukuran *delay* rata-rata *delay* , menghitung semua paket yang dikirimkan dalam simulasi. *Delay* terdapat *delay* pada jaringan VANET normal dan jaringan VANET dengan serangan. Dengan ini bisa dilihat grafik perubahan apabila jaringan VANET ter serang oleh *Blackhole*.

1. *Throughput*

Untuk pengukuran *throughput*, pengukuran dilakukan berdasarkan jumlah data yang berhasil dikirim dan diterima dalam satu unit waktu. Ini biasanya diukur dalam satuan bits per detik (bps) atau bytes per detik (Bps). *Throughput* mengukur efisiensi dari jaringan dalam mentransfer data. Dengan ini peneliti akan membandingkan *throughput* rata-rata antara kondisi jaringan normal dan jaringan yang terkena serangan *Blackhole*. Diskusikan bagaimana serangan *Blackhole* mempengaruhi *throughput* jaringan.

* + 1. Menampilkan Hasil

Hasil ditampilkan menggunakan 4 *Subplot* plot

1. ***Subplot* 1: Jaringan VANET Normal**:

Menampilkan visualisasi jaringan VANET dalam kondisi normal.

1. ***Subplot* 2: Jaringan VANET dengan Serangan**:

Menampilkan visualisasi jaringan VANET dengan serangan *Blackhole*. Tandai *Node* yang terkena serangan dengan warna merah.

1. ***Subplot* 3:** *Delay*:

Menampilkan perbandingan *delay* rata-rata untuk jaringan VANET dalam kondisi normal dan dengan serangan. Gunakan grafik garis untuk memperlihatkan perbedaan *delay* .

1. ***Subplot* 4:** *Throughput*:

Menampilkan perbandingan *throughput* rata-rata untuk jaringan VANET dalam kondisi normal dan dengan serangan. Gunakan grafik garis untuk membandingkan *throughput*.

* + 1. Menyusun Laporan

Pada tahap ini peneliti mulai meyusun laporan dengan membuat makalah skripsi. Peneliti dapat memasukkan hasil penelitian ke dalam makalah skripsi. Hasil dan pembahasan meliputi.

1. Pembuatan jaringan VANET.

Penjelasan mengenai proses pembuatan jaringan VANET dalam simulasi, termasuk konfigurasi *Node* , pengaturan topologi, dan parameter yang digunakan.

1. Pembuatan protokol jaringan AODV.

Uraian tentang implementasi protokol AODV (*Ad hoc* *On-Demand* *Distance* *Vector*) dalam jaringan VANET, termasuk cara kerja protokol dalam mengatur rute komunikasi antara *Node* .

1. Modifikasi *Routing protocol* AODV.

Penjelasan mengenai perubahan yang dilakukan pada protokol AODV untuk meningkatkan ketahanannya terhadap serangan *Blackhole*. Modifikasi ini mencakup perubahan pada paket permintaan rute (RREQ) dan paket balasan rute (RREP).

1. Pemberian serangan *Blackhole*.

Uraian tentang bagaimana serangan *Blackhole* diterapkan dalam simulasi, termasuk mekanisme serangan dan pengaruhnya terhadap jaringan VANET.

1. Penampilan hasil serangan.

Penyajian hasil visualisasi serangan *Blackhole*, termasuk perbedaan antara jaringan VANET yang terkena serangan dan yang tidak, serta dampak serangan pada jaringan.

1. Pengumpulan data penelitian terkait serangan *Blackhole*

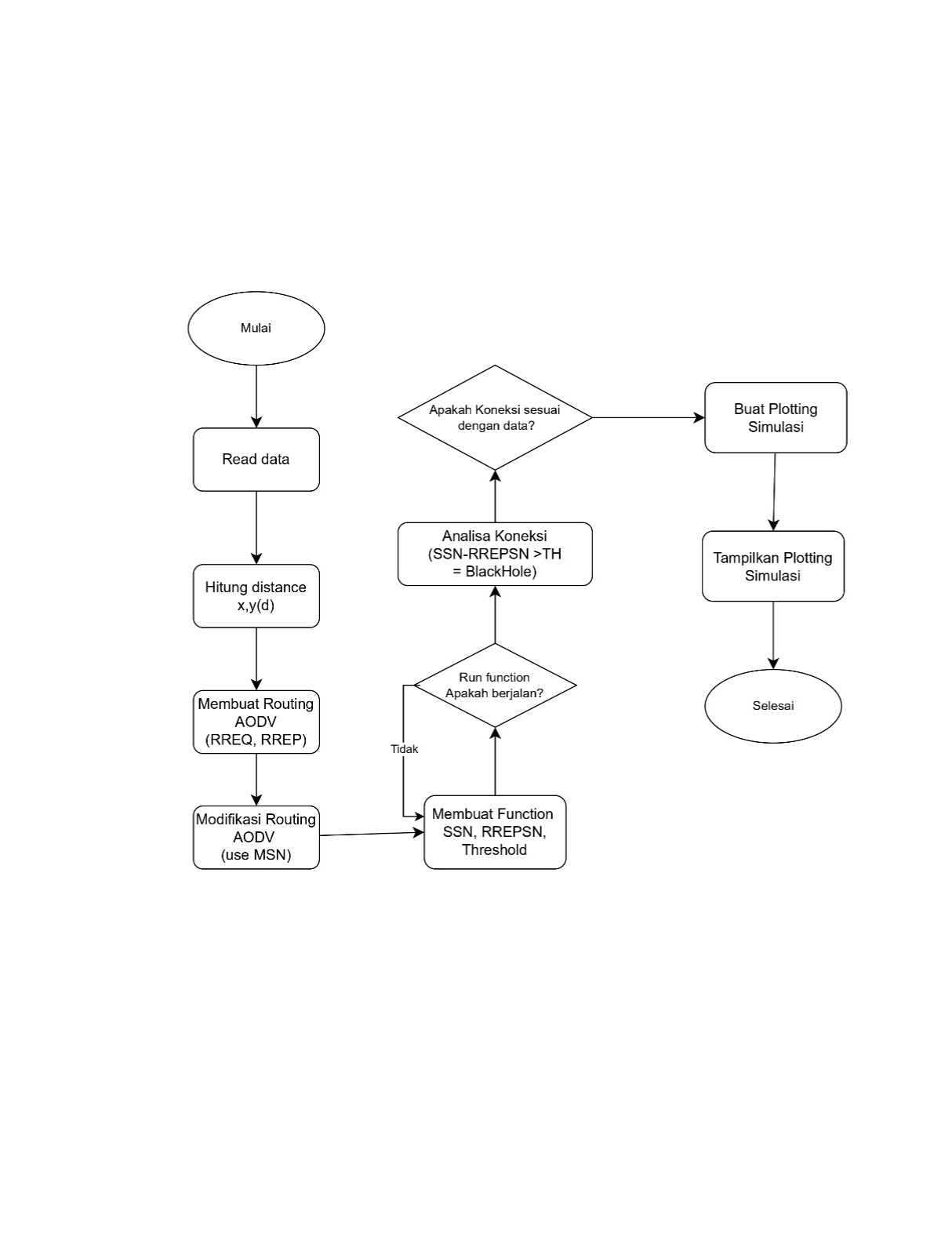
Proses pengumpulan data terkait serangan *Blackhole*, termasuk pengukuran *delay* dan *throughput* dalam kondisi normal dan saat terjadinya serangan. Data ini digunakan untuk analisis performa jaringan.

1. Kesimpulan dari penelitian yang dibuat.

Ringkasan temuan utama dari penelitian, termasuk kesimpulan tentang dampak serangan *Blackhole* terhadap kinerja jaringan VANET dan efektivitas modifikasi protokol AODV dalam menghadapi serangan tersebut.

## Alur Program Tugas akhir

Bagian ini menguraikan langkah-langkah yang ditempuh oleh penulis dalam menyelesaikan penelitian ini. Berikut adalah pembahasan mengenai alur program penelitian yang disusun melalui beberapa tahapan.



Gambar 7 Alur Program VANET AODV

1. Read data

Pada tahap ini, data yang dibutuhkan untuk simulasi diperoleh dari *File* sumber, seperti *File* Excel atau *File* sejenis yang menyimpan informasi mengenai *Node* dalam jaringan. *File* tersebut menyediakan data penting yang digunakan dalam proses simulasi. Data ini mencakup berbagai informasi terkait karakteristik dan kondisi *Node* . Informasi yang tersedia memastikan bahwa setiap aspek penting dari *Node* dalam jaringan tercakup. Dengan data tersebut, simulasi dapat berjalan sesuai dengan skenario yang diinginkan. Data ini mencakup informasi penting seperti:

1. Posisi *Node*

Posisi *Node* adalah koordinat geografis atau posisi dalam ruang simulasi di mana *Node* berada pada titik waktu tertentu. Dalam konteks jaringan *ad-hoc* atau VANET, posisi ini biasanya diberikan dalam bentuk koordinat *Cartesian* (x, y) dalam suatu area simulasi.

1. Waktu

Waktu mencatat kapan peristiwa tertentu terjadi dalam simulasi. Ini bisa berupa waktu simulasi yang sebenarnya atau waktu relatif dalam simulasi.

1. *Id* *Node*

Ide *Node* adalah identifikasi unik untuk setiap *Node* dalam jaringan. Ini penting untuk membedakan *Node* satu dengan yang lainnya, terutama dalam proses *Node Routing* dan manajemen jaringan.

1. Hitung *Distance*

Pada tahap ini, dilakukan perhitungan jarak antara setiap pasangan *Node* dalam jaringan. Perhitungan jarak menggunakan rumus Euclidean yang dinyatakan sebagai:

|  |  |
| --- | --- |
|  | (4) |

dengan menggunakan posisi (x, y) yang telah dibaca sebelumnya. Jarak ini penting untuk menentukan apakah dua *Node* dapat berkomunikasi langsung satu sama lain. Berdasarkan hasil perhitungan, akan diketahui apakah *Node* -*Node* tersebut berada dalam jangkauan komunikasi langsung. Ini membantu dalam menentukan konektivitas dan pengaturan rute dalam simulasi jaringan. Perhitungan ini bertujuan untuk mendukung proses *Node Routing* .

1. Membuat *Node Routing* AODV

Pada tahap ini, diimplementasikan protokol *Node Routing* AODV, yang merupakan protokol dinamis untuk jaringan *ad-hoc*. Proses utama yang dilakukan meliputi:

1. Permintaan Rute (RREQ): *Node* yang memerlukan rute mengirimkan permintaan ke seluruh jaringan untuk mencari jalur menuju *Node* tujuan.
2. Respon Rute (RREP): Setelah rute ditemukan, *Node* tujuan mengirimkan respons kembali ke *Node* asal dengan informasi tentang rute terbaik yang tersedia.
3. Pemeliharaan Rute: Rute akan diperbarui atau dibatalkan jika terjadi perubahan dalam jaringan, seperti pergerakan *Node* yang keluar dari jangkauan.

Hal ini membantu untuk mengatur komunikasi antara *Node* dengan cara menemukan dan mempertahankan rute yang optimal.

1. Modifikasi *Node Routing* AODV

Setelah implementasi dasar protokol *Node Routing* AODV selesai, berbagai modifikasi dapat dilakukan untuk meningkatkan kinerja atau menyesuaikan dengan kebutuhan spesifik. Misalnya:

1. Optimasi Rute: Mengurangi *overhead* dan mempercepat proses pencarian rute untuk efisiensi yang lebih baik.
2. Penambahan Fitur Keamanan: Menambahkan mekanisme untuk mendeteksi dan mencegah serangan *Blackhole* dalam jaringan.
3. Penanganan *Node* yang Bergerak Cepat: Menyesuaikan AODV untuk beroperasi secara efektif dalam kondisi jaringan dengan mobilitas tinggi, seperti dalam VANET.

Tujuan dari modifikasi ini adalah untuk meningkatkan efisiensi dan keamanan *Node Routing* AODV dalam berbagai situasi jaringan.

1. Membuat Function SSN, RREPSN, *Threshold*

1. SSN (*Source Sequence Number*)

* Definisi: SSN adalah angka urut yang digunakan dalam protokol *Node Routing* untuk melacak pesan dan memastikan urutan yang benar dalam transmisi data. SSN membantu untuk mengidentifikasi pesan yang lebih baru dan memastikan bahwa tidak ada pesan yang hilang atau duplikat.
* Penggunaan: Dalam *Node Routing* AODV (*Ad hoc On-Demand Distance Vector*), setiap pesan kontrol seperti *Route Request* (RREQ) dan *Route Reply* (RREP) memiliki SSN. *Node* yang menerima pesan dapat menggunakan SSN untuk menentukan apakah pesan tersebut adalah yang terbaru dan valid.

2. RREPSN (Route *Reply Sequence Number*)

* Definisi: RREPSN adalah angka urut yang digunakan dalam proses Route *Reply* untuk memvalidasi rute yang diterima. RREPSN memastikan bahwa *Node* yang menerima RREP dapat memverifikasi dan memilih rute yang lebih baru dan lebih optimal.
* Penggunaan: RREPSN dalam pesan RREP membantu menentukan apakah rute yang diterima adalah yang terbaru. *Node* tujuan dalam AODV akan membandingkan RREPSN dari berbagai RREP yang diterimanya untuk memilih rute yang terbaik.

3. *Threshold* (Ambang Batas)

Definisi: *Threshold* adalah nilai ambang yang digunakan untuk memutuskan apakah sebuah rute atau *Node* masih valid atau perlu diperbarui. Ini sering digunakan dalam perhitungan kinerja untuk memastikan bahwa *Node* atau rute yang memiliki metrik atau nilai di bawah ambang batas dianggap tidak valid atau perlu diperbaiki.

* Penggunaan: Dalam analisis koneksi, *Threshold* digunakan untuk menentukan apakah suatu rute atau *Node* harus dianggap sebagai *Blackhole* atau tidak. Jika perbedaan antara SSN dan RREPSN melebihi *Threshold*, rute atau *Node* dapat dianggap sebagai *Blackhole*.

Rumus SSN - RREPSN > TH = *Blackhole*

* Rumus: SSN - RREPSN > TH = *Blackhole*
  + SSN: *Sequence Sequence Number*, angka urut dari pesan.
  + RREPSN: Route *Reply Sequence Number*, angka urut dari rute balasan.
  + TH: *Threshold*, nilai ambang batas.

1. Analisa koneksi

Pada tahap ini, konektivitas jaringan dianalisis menggunakan data yang dikumpulkan dari simulasi. Beberapa analisis yang dilakukan termasuk:

* Validasi Rute: Memeriksa apakah rute yang ditemukan masih valid berdasarkan SSN dan RREPSN.
* Deteksi *Node* yang Terputus: Menentukan apakah ada *Node* yang keluar dari jangkauan atau tidak lagi dapat dihubungi.
* Evaluasi Kinerja Jaringan: Misalnya, berapa banyak rute yang valid, berapa banyak yang gagal, dan bagaimana kinerja jaringan secara keseluruhan.

Tujuan: Menilai stabilitas dan kinerja koneksi jaringan.

1. Buat *plotting* Simulasi

Setelah analisis selesai, hasil simulasi divisualisasikan. *Plotting* dapat mencakup:

* Jaringan *Node* : Representasi visual dari *Node* dan rute yang terbentuk di antara mereka.
* Status *Node* : Warna atau simbol yang berbeda untuk menunjukkan status *Node* (misalnya, hijau untuk terhubung, merah untuk terputus).
* Perubahan Jaringan: Bagaimana jaringan berubah dari waktu ke waktu, terutama jika *Node* bergerak.

Tujuan: Menyediakan cara visual untuk memahami hasil simulasi dan dinamika jaringan.

1. Tampilkan *Plotting* simulasi

Tahap akhir adalah menampilkan hasil *plotting* kepada pengguna. Ini dapat dilakukan dalam bentuk grafik atau animasi yang menunjukkan perkembangan jaringan dari waktu ke waktu.

Hal ini bertujuan untuk menyampaikan hasil simulasi secara visual agar lebih mudah dipahami dan dianalisis.

# BAB IV HASIL DAN PEMBAHASAN



## Hasil

Bab ini membahas hasil dari simulasi dan eksperimen yang dilakukan untuk menganalisis dampak serangan *Blackhole* pada jaringan *Vehicular Ad-hoc* *Vehicular* (VANET) dengan menggunakan protokol *Routing* AODV, yang telah diimplementasikan dan dimodifikasi dalam lingkungan MATLAB. Serangan *Blackhole* adalah salah satu ancaman serius dalam jaringan *ad-hoc*, di mana *Node* jahat mengklaim memiliki rute optimal ke tujuan, tetapi kemudian membuang semua paket data yang diterimanya, mengakibatkan hilangnya data dan penurunan kinerja jaringan secara keseluruhan.

Pada bab ini, akan disajikan hasil simulasi yang meliputi performa jaringan dalam kondisi normal serta saat terjadi serangan *Blackhole*. Parameter kinerja utama yang dianalisis adalah *delay* dan *throughput*, di mana kedua metrik ini memberikan gambaran yang jelas mengenai dampak serangan terhadap efisiensi dan stabilitas jaringan VANET. Selain itu, perbandingan antara jaringan VANET normal dan jaringan yang terpapar serangan *Blackhole* akan dijelaskan melalui berbagai visualisasi data, termasuk grafik yang dihasilkan menggunakan MATLAB.

Pembahasan akan difokuskan pada analisis mendalam mengenai bagaimana serangan *Blackhole* mempengaruhi *delay* dan *throughput*, serta bagaimana modifikasi protokol AODV yang dilakukan dapat membantu dalam mendeteksi dan mengurangi dampak dari serangan ini. Hasil dari analisis ini diharapkan dapat memberikan wawasan yang lebih dalam mengenai strategi peningkatan keamanan dan kinerja jaringan VANET di masa depan.

1. SUMO

Pada bagian ini, hasil simulasi *Vehicle Ad-hoc* *Vehicular* (VANET) akan dipresentasikan. Simulasi tersebut dilakukan menggunakan perangkat lunak SUMO dan MATLAB. Area studi yang digunakan dalam simulasi adalah wilayah RS. PKU Muhammadiyah Yogyakarta. Gambar 6 menunjukkan peta area studi yang digunakan dalam penelitian ini. Analisis dilakukan berdasarkan data yang dikumpulkan dari simulasi di area tersebut.

A map of a city

Description automatically generated

Gambar 8 Lokasi Maps RS. PKU Yogyakarta

Setelah memperoleh data lokasi dari wilayah yang telah ditentukan, langkah berikutnya adalah mengimpor data tersebut ke dalam perangkat lunak SUMO. Proses ini digunakan untuk merancang simulasi yang akan dilakukan. Setelah data berhasil diimpor, simulasi dirancang dengan menetapkan parameter dan skenario yang spesifik. Parameter tersebut mencakup berbagai variabel yang akan mempengaruhi perilaku jaringan kendaraan. Skenario yang berbeda juga disiapkan untuk menggambarkan berbagai kondisi lalu lintas di wilayah tersebut. Dengan pengaturan yang telah ditetapkan, simulasi kemudian dijalankan. Tujuannya adalah untuk mengamati dan menganalisis perilaku jaringan kendaraan di area yang dimodelkan. A screenshot of a computer screen

Description automatically generated

Gambar 9 Simulasi menggunakan SUMO

Dapat dilihat pada gambar bahwa sistem lalu lintas yang dihasilkan dalam simulasi memiliki kemiripan dengan kondisi lalu lintas nyata di wilayah RS. PKU Muhammadiyah Yogyakarta. Simulasi ini mencakup berbagai aspek, mulai dari arah pergerakan kendaraan hingga pengaturan lampu lalu lintas. Setiap elemen dalam sistem lalu lintas simulasi dirancang untuk mencerminkan situasi yang ada di dunia nyata. Dengan demikian, simulasi ini mampu menciptakan representasi yang realistis dari lalu lintas di wilayah tersebut. Representasi ini penting untuk memahami bagaimana lalu lintas beroperasi di area yang diteliti.

Dalam simulasi yang telah dilakukan, digunakan dua jenis kendaraan, yaitu mobil dan taksi. Penggunaan dua jenis kendaraan ini bertujuan untuk melihat bagaimana masing-masing jenis kendaraan berinteraksi satu sama lain. Interaksi ini penting karena dapat mempengaruhi arus lalu lintas secara keseluruhan. Selain itu, dengan adanya variasi jenis kendaraan, simulasi menjadi lebih komprehensif dalam menggambarkan kondisi lalu lintas. Pemahaman tentang bagaimana kendaraan-kendaraan ini berperilaku dalam situasi yang berbeda memberikan wawasan penting untuk analisis lebih lanjut.

Dalam simulasi yang telah dilakukan, dua jenis kendaraan, yaitu mobil dan taksi, digunakan untuk melihat bagaimana interaksi antara keduanya dapat mempengaruhi arus lalu lintas secara keseluruhan. Interaksi ini penting karena memberikan gambaran yang lebih komprehensif mengenai kondisi lalu lintas.

## Hasil Modified *Sequence Number* (MSN)

A screenshot of a computer

Description automatically generated

Gambar 10 Gambar AODV MSN

Pada gambar .. menggambarkan bagaimana AODV menggunakan pesan RREQ untuk mencari rute dari satu *Node* ke *Node* lainnya, dan bagaimana pesan RREP digunakan untuk memberi tahu *Node* sumber bahwa rute telah ditemukan. Setiap *Node* yang menerima RREQ dan tahu rute akan mengirimkan RREP kembali ke *Node* asal, yang ditunjukkan dengan *Sequence Number* (RREPSN) sebagai referensi untuk memastikan rute yang dikirim adalah yang terbaru.

Pada gambar dijelaskan *Node* 3 mengirim pesan RREP dengan *Sequence Number* 4 (RRREPSN=4) ke *Node* 61. *Node* 3, *Node* 7, *Node* 19. *Node* -*Node* ini merespons dengan RREP setelah menemukan rute kembali ke *Node* 61. *Sequence Number* digunakan untuk memastikan informasi rute yang diterima oleh *Node* 61 adalah informasi terbaru dari *Node* 3. Setelah menerima RREP, *Node* sumber mengetahui jalur yang harus diambil untuk mencapai *Node* tujuan.

Pada modifikasi AODV, dilakukan penambahan fungsi perulangan untuk deteksi *Blackhole*. Modifikasi dimulai dengan pemberian *Source* *Sequence Number* (SSN), yang mengacu pada data waktu (t) dan id *Node* (id). Pada saat *Node* sumber mengirimkan pesan, SSN diberikan secara berurutan.

Selanjutnya, dibuat RREP *Sequence Number* (RREPSN), di mana saat *Node* sumber mengirimkan RREQ dan *Node* penerima membalas dengan RREP, *Node* sumber akan menerima RREPSN yang juga berurutan. Setelah membuat fungsi ini, dilanjutkan dengan pemberian fungsi *Threshold* (ambang batas) untuk mendeteksi kecurigaan *Node* .

Semua *Sequence Number* (SSN dan RREPSN) disimpan dalam sebuah tabel, sehingga semua data dapat diakses dan dianalisis. Setelah data tersimpan, langkah berikutnya adalah membuat fungsi untuk mendeteksi serangan *Blackhole*. Serangan *Blackhole* ditentukan berdasarkan perbandingan antara SSN dan RREPSN. Jika perbedaan antara SSN dan RREPSN melebihi ambang batas yang telah ditentukan, maka *Node* tersebut diidentifikasi sebagai *Node Blackhole*.



## Normal VANET

A graph with lines and dots

Description automatically generated

Gambar 11 VANET Normal

Pada Gambar , menggambarkan koneksi VANET, koneksi ini menunjukkan koneksi komunikasi kendaraan-antar-kendaraan V2V. Koneksi ini menunjukkan bahwa kendaraan telah berhasil menjalin komunikasi langsung dengan kendaraan di sekitarnya untuk bertukar informasi tentang posisi, kecepatan, dan arah pergerakan, sehingga memungkinkan adanya kolaborasi dan koordinasi untuk menghindari kemacetan atau kecelakaan.

## *Blackhole Attack* VANET

A graph of a graph

Description automatically generated

Gambar 12 Blackhole VANET

Pada gambar , menunjukkan beberapa adanya koneksi dengan berbagai warna, warna biru untuk koneksi antar kendaraan terhubung(*connected*) dan merah menunjukkan koneksi antar kendaraan terputus (*disconnected*). Selain pewarnaan untuk koneksi, terdapat juga perbedaaan warna *Node* , yaitu biru untuk *Node* normal tidak terkena ketentuan *Blackhole*, sedangkan unutk *Node* merah merupakan *Node* yang melakukan serangan *Blackhole*, di mana *Node* ini menyamar sebagai tujuan yang sah dan menarik semua lalu lintas data.

Ketika serangan *Blackhole* terdeteksi yang sebelumnya sudah di tentukan oleh *Threshold*, dilakukan kondisi pada koneksi untuk mengubah rute tujuan ke *Node Blackhole*, tujuan perubahan rute karna konsep serangan *Blackhole* yaitu membuang pesan yang akan dikirim ke *destination Node* , sehingga *destination Node* tidak mendapatkan pesan dari *source*, namun *source* mendapatkan pesan *error* berupa *Sequence Number Reply* menjadi acak dan tidak menentu seperti hasil pada gambar dibawah ini.

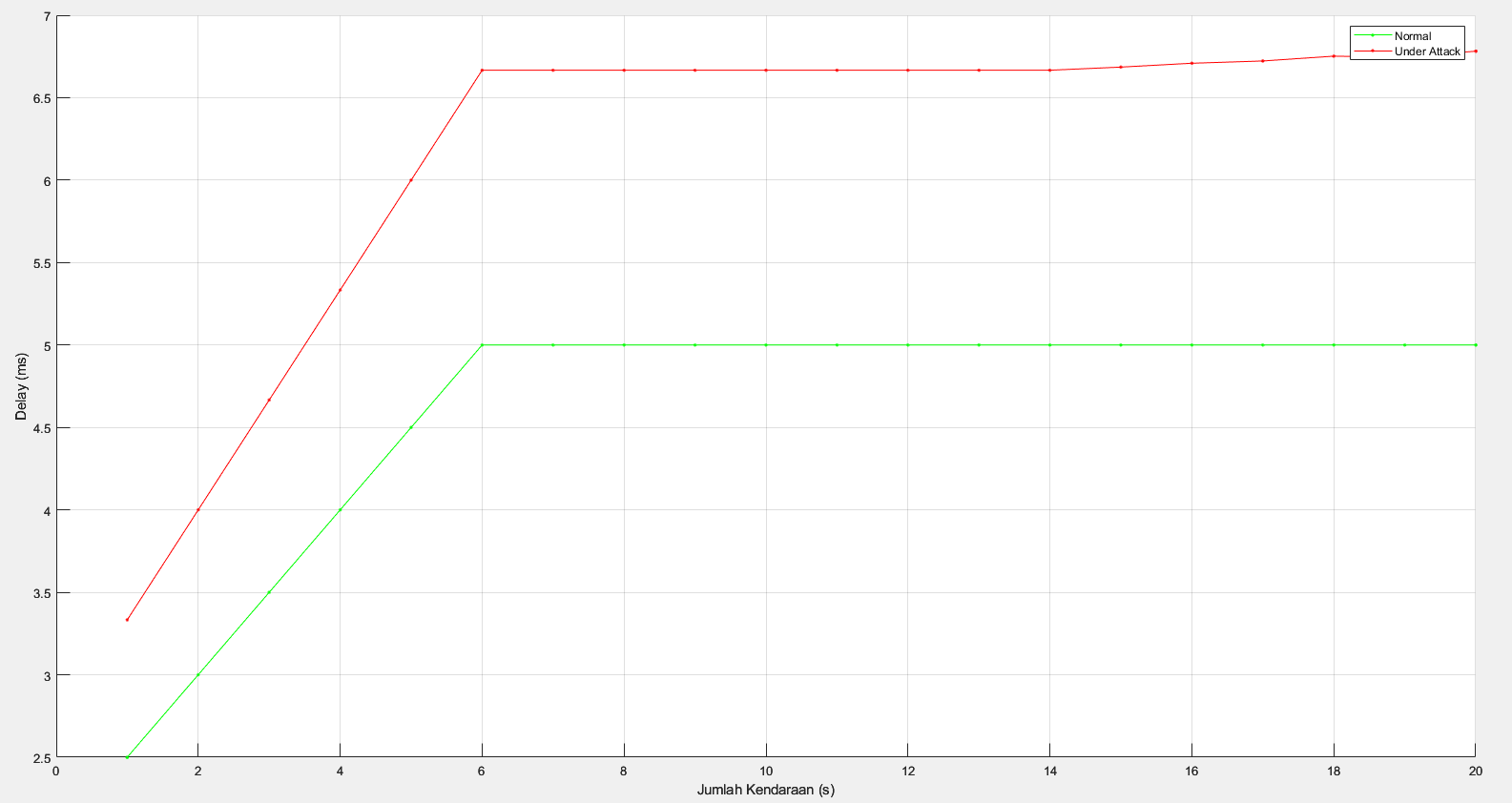
A screenshot of a computer

Description automatically generated

Gambar 13 Data tabel serangan Blackhole

Pada gambar, terlihat bahwa *Node* yang berfungsi sebagai *Blackhole* menerima nomor urut balasan (*Sequence Number*) yang acak dan tidak beraturan. Nomor urut ini adalah yang seharusnya diterima oleh *Node* sumber. Akibat dari serangan *Blackhole* ini, *Node* sumber memperoleh pesan kesalahan yang berupa nomor urut yang tidak konsisten. Hal ini disebabkan oleh fakta bahwa *Node Blackhole* sengaja membuang pesan yang dikirim ke *Node* tujuan. Akibatnya, *Node* sumber mengalami gangguan dalam proses komunikasi karena nomor urut yang diterima tidak sesuai dengan yang diharapkan.

## Perbandingan *Delay*



Gambar 14 Grafik perbandingan Delay

Setelah pendeteksian serangan *Blackhole*, terlihat pada gambar perbandingan perubahan *delay* . Grafik ini memiliki dua warna, yaitu hijau dan merah. Grafik berwarna hijau menggambarkan *delay* pada jaringan VANET yang beroperasi secara normal tanpa adanya serangan. Grafik tersebut menunjukkan peningkatan *delay* yang gradual seiring bertambahnya jumlah kendaraan.

Di sisi lain, grafik berwarna merah menggambarkan *delay* pada jaringan yang terganggu oleh serangan *Blackhole*. Terlihat bahwa pada t=15 terjadi lonjakan signifikan pada *delay* , yang menandakan adanya penurunan performa jaringan. Peningkatan *delay* ini menunjukkan bahwa jaringan mengalami gangguan, yang berdampak pada penurunan performa.

*Delay* dalam jaringan dipengaruhi oleh keterhubungan antar *Node* . Ketika terjadi peningkatan *delay* , ini mengindikasikan penurunan performa jaringan. Dalam penelitian ini, peningkatan *delay* terjadi akibat serangan *Blackhole* yang menyebabkan *Node* menjadi terputus. *Node* yang terputus ini berdampak pada kinerja jaringan secara keseluruhan. Selain itu, *Node* yang terputus akan mencari rute lain, yang juga mempengaruhi koneksi jaringan. Sebagai perbandingan, grafik menunjukkan bagaimana *delay* pada jaringan normal dibandingkan dengan jaringan yang terkena serangan *Blackhole*.

Table 4 Tabel data Delay

|  |  |  |  |
| --- | --- | --- | --- |
| Waktu  (t) | indeks | | Selisih  (Normal - Blackhole) |
| Normal | *Blackhole* |
| 14 | 5 | 6.67 | 1.67 |
| 15 | 5 | 6.68 | 1.68 |
| 16 | 5 | 6.70 | 1.70 |
| 17 | 5 | 6.72 | 1.72 |
| 18 | 5 | 6.75 | 1.75 |
| 19 | 5 | 6.74 | 1.74 |
| 20 | 5 | 6.78 | 1.78 |
| Total | | | 12.04 |
| Rata-Rata | | | 1.72 |

Dapat terlihat pada tabel perbandingan antara nilai pada jaringan VANET Normal dan terjadi serangan *Blackhole*. Pada tabel, terlihat data ketika terjadi serangan nilai *delay* meninggi. Selisih antara jaringan VANET normal dan terjadi serangan tidaklah signifikan, namun terjadi peningkatan *delay* pada jaringan VANET ketika terjadi hambatan perutean akibat terjadinya serangan *blackhole*.

## Perbandingan Throughput

A graph with red and green lines

Description automatically generated

Gambar 15 Grafik Perbandingan Throughput

Pada Gambar 13 terlihat bahwa grafik memiliki dua warna, yaitu hijau dan merah. Grafik berwarna hijau menunjukkan *throughput* pada jaringan VANET yang beroperasi secara normal, sedangkan grafik berwarna merah menunjukkan *throughput* pada jaringan VANET yang terkena serangan *Blackhole*. Dari gambar tersebut, terlihat adanya penurunan *throughput* yang cukup signifikan pada jaringan VANET ketika mengalami serangan *Blackhole*.

Setelah menganalisis data perbandingan *delay* akibat serangan *Blackhole*, data *throughput* digunakan untuk mendukung pengukuran kinerja jaringan VANET. Grafik tersebut menunjukkan perbandingan *throughput* antara koneksi jaringan VANET normal dan jaringan VANET yang diserang oleh *Blackhole*. Penurunan *throughput* yang cukup drastis terjadi saat jaringan VANET terserang oleh *Blackhole*, yang menunjukkan bahwa serangan ini memiliki dampak signifikan terhadap kepadatan lalu lintas. Penurunan ini terjadi karena koneksi yang terputus dan perubahan rute akibat *Node* yang diserang oleh *Blackhole*.

Table 5 Tabel data Delay

|  |  |  |  |
| --- | --- | --- | --- |
| Waktu  (t) | indeks | | Selisih  (Normal - Blackhole) |
| Normal | *Blackhole* |
| 14 | 0.2 | 0.15 | 0.05 |
| 15 | 0.2 | 0.09 | 0.11 |
| 16 | 0.2 | 0.03 | 0.17 |
| 17 | 0.2 | 0.02 | 0.18 |
| 18 | 0.2 | -0.05 | 0.25 |
| 19 | 0.2 | 0.03 | 0.17 |
| 20 | 0.2 | 0.09 | 0.11 |
| Total | | | **1.04** |
| Rata-rata | | | **0.149** |

Pada tabel, terlihat nilai data pada jaringan VANET normal yaitu cukup gradual diangka nilai 0.2, sedangkan untuk jaringan VANET terkena serangan *Blackhole* terlihat berubah ubah seiring dengan bertambahnya jumlah serangan dan berubahnya perutean koneksi. Selisih antara *throughput* jaringan VANET dan terjadi serangan *Blackhole* cukup signifikan, sehingga dapat terlihat jelas pada grafik perubahan atau dampak *throughput* ketika terjadi serangan *Blackhole* pada jaringan VANET

## Tinjauan Hasil Tugas akhir dibanding dengan Tugas akhir Terdahulu

Penelitian ini berfokus pada analisis dampak serangan *Blackhole* terhadap kinerja jaringan VANET menggunakan protokol *Routing* AODV. Dalam tugas akhir ini, telah dilakukan modifikasi pada protokol AODV untuk mendeteksi serangan *Blackhole*, yang merupakan peningkatan signifikan dibandingkan dengan penelitian terdahulu.

Penelitian terdahulu umumnya hanya berfokus pada performa dasar protokol AODV tanpa memperhitungkan adanya serangan atau ancaman terhadap jaringan. Beberapa penelitian sebelumnya juga membandingkan kinerja beberapa protokol *Routing* , tetapi tidak banyak yang mengeksplorasi dampak dari serangan spesifik seperti *Blackhole*. Sebaliknya, penelitian ini tidak hanya mengevaluasi kinerja AODV di bawah kondisi normal, tetapi juga menganalisis secara mendalam bagaimana serangan *Blackhole* mempengaruhi *delay* dan *throughput* jaringan. Modifikasi protokol AODV yang dikembangkan juga memungkinkan deteksi dan mitigasi serangan *Blackhole*, yang memberikan kontribusi baru dalam penelitian di bidang keamanan jaringan VANET.

Dengan demikian, hasil dari penelitian ini menawarkan perspektif baru yang lebih komprehensif mengenai keamanan dan kinerja jaringan VANET dibandingkan dengan tugas akhir terdahulu, yang cenderung lebih berfokus pada aspek performa tanpa mempertimbangkan ancaman keamanan​

# BAB V KESIMPULAN DAN SARAN



## Kesimpulan

Dalam tugas akhir ini, dilakukan penelitian terhadap dampak serangan *Blackhole* pada kinerja jaringan VANET menggunakan protokol *Routing* AODV. Untuk itu, modifikasi dilakukan pada protokol AODV guna mendeteksi dan mengurangi dampak dari serangan tersebut. Penelitian ini melibatkan simulasi menggunakan MATLAB untuk menganalisis kinerja jaringan dalam kondisi normal dan di bawah serangan *Blackhole*. Berdasarkan penelitian yang telah dilakukan, dapat disimpulkan beberapa hal sebagai berikut:

1. Modifikasi pada protokol AODV berhasil diimplementasikan dan efektif dalam mendeteksi serta memitigasi dampak serangan *Blackhole* dalam jaringan VANET.
2. Serangan *Blackhole* terbukti memberikan dampak negatif yang signifikan terhadap kinerja jaringan VANET, terutama dalam hal peningkatan *delay* dan penurunan *throughput*.
3. Penerapan modifikasi pada protokol AODV mampu meningkatkan kinerja jaringan dengan mengurangi dampak serangan, meskipun belum sepenuhnya menghilangkan ancaman.
4. Penelitian ini menunjukkan pentingnya aspek keamanan dalam jaringan VANET dan bagaimana modifikasi protokol dapat memperkuat jaringan terhadap serangan tertentu.
5. Hasil penelitian ini dapat menjadi referensi untuk pengembangan lebih lanjut di bidang keamanan jaringan *ad-hoc*, terutama dalam konteks VANET yang dinamis.

## Saran

1. Disarankan untuk melakukan pengujian tambahan dengan skenario lalu lintas yang lebih kompleks dan realistis untuk menguji efektivitas protokol dalam situasi yang lebih mendekati kenyataan.
2. Pengintegrasian dengan teknologi keamanan lain, seperti enkripsi, diharapkan dapat meningkatkan keamanan jaringan VANET secara keseluruhan.
3. Pengembangan dan pengujian lebih lanjut di lingkungan nyata akan memberikan validasi yang lebih akurat terhadap efektivitas modifikasi protokol yang dilakukan.
4. Pemberian pengamanan ketika serangan terjadi dapat ditambahkan untuk pengembangan lebih lanjut.

**DAFTAR PUSTAKA**

Assidiq, N. F., & Nurcahyani, I. (n.d.). *ANALISIS PENGARUH KINERJA ROUTING PROTOCOL AODV DAN DSDV TERHADAP KONSUMSI ENERGI NODE PADA JARINGAN*.

Biswas, A. K., & Dasgupta, M. (2019). AODV-DSR Hybrid Reactive *Routing Protocol* and its Generalization for Mobile *Ad-hoc* *Vehicular* s. *2019 3rd International Conference on Electronics, Materials Engineering & Nano-Technology (IEMENTech)*, 1–5. https://doi.org/10.1109/IEMENTech48150.2019.8981052

Günay, F. B., Öztürk, E., Çavdar, T., Hanay, Y. S., & Khan, A. U. R. (2021). *Vehicular Ad hoc* *Vehicular* (VANET) Localization Techniques: A Survey. *Archives of Computational Methods in Engineering*, *28*(4), 3001–3033. https://doi.org/10.1007/s11831-020-09487-1

Kugali, S. N., & Kadadevar, S. (n.d.). *Vehicular* ADHOC *Vehicular* (VANET):-A Brief Knowledge. *International Journal of Engineering Research*, *9*(06).

Kumar, A., Varadarajan, V., Kumar, A., Dadheech, P., Choudhary, S. S., Kumar, V. D. A., Panigrahi, B. K., & Veluvolu, K. C. (2021). Black hole *Attack* detection in *Vehicular ad-hoc* *Vehicular* using secure AODV *Routing* algorithm. *Microprocessors and Microsystems*, *80*, 103352. https://doi.org/10.1016/j.micpro.2020.103352

Kumar, M., Jain, V., Jain, A., Bisht, U. S., & Gupta, N. (2019). Evaluation *of* black hole *Attack* with avoidance scheme using AODV *protocol* in VANET. *Journal of Discrete Mathematical Sciences and Cryptography*, *22*(2), 277–291. https://doi.org/10.1080/09720529.2019.1585635

Kuthe, A., Chaudhari, T., Thakare, S., Dohtare, P., & Waghmare, D. (2022). Design and Implementation *of* VANET in *Ad hoc* *Vehicular* using MATLAB. *International Journal of Computer Science and Mobile Computing*, *11*(3), 49–55. https://doi.org/10.47760/ijcsmc.2022.v11i03.007

Manapa, E. S., Sampetoding, E. A. M., & Lewakabessy, G. (2020). POTENSI PENGGUNAAN MOBILE *AD-HOC* *VEHICULAR* (MANET) SEBAGAI ALAT KOMUNIKASI DATA PADA TRANSPORTASI DI INDONESIA. *Journal Dynamic Saint*, *4*(2), 865–868. https://doi.org/10.47178/dynamicsaint.v4i2.889

Moler, C., & Little, J. (2020). A history *of* MATLAB. *Proceedings of the ACM on Programming Languages*, *4*(HOPL), 1–67. https://doi.org/10.1145/3386331

Naim, Z., & Hossain, Md. I. (2019). Performance Analysis *of* AODV, DSDV And DSR in *Vehicular* Adhoc *Vehicular* (VANET). *2019 International Conference on Robotics,Electrical and Signal Processing Techniques (ICREST)*, 17–22. https://doi.org/10.1109/ICREST.2019.8644313

Nazif, F., Wibawa, I. P. D., & Rizal, A. (n.d.). *Pemantauan Dan Notifikasi Kondisi Tanah Pada Tanaman Menggunakan Platform Iot*.

Oberoi, V. (2020). Enhancement *of* QoS in Security Algorithm for *Blackhole Attack* in VANET. *2020 IEEE Pune Section International Conference (PuneCon)*, 33–37. https://doi.org/10.1109/PuneCon50868.2020.9362444

Shrestha, S., Baidya, R., Giri, B., & Thapa, A. (2020). Securing *Blackhole Attack*s in MANETs using Modified *Sequence Number* in AODV *Routing* Protocol. *2020 8th International Electrical Engineering Congress (iEECON)*, 1–4. https://doi.org/10.1109/iEECON48109.2020.229555

Sindhwani, M., Singh, R., Sachdeva, A., & Singh, C. (2022). Improvisation *of* optimization technique and AODV *Routing protocol* in VANET. *Materials Today: Proceedings*, *49*, 3457–3461. https://doi.org/10.1016/j.matpr.2021.03.727

Singh, P. K., Nandi, S. K., & Nandi, S. (2019). A tutorial survey on *Vehicular* communication state *of* the art, and future research directions. *Vehicular Communications*, *18*, 100164. https://doi.org/10.1016/j.vehcom.2019.100164

Upadhyaya, A. N., & Shah, J. S. (2019). Effect on aodv *Routing protocol* under *Blackhole Attack* in VANET. *INTERNATIONAL JOURNAL OF COMPUTER ENGINEERING AND TECHNOLOGY*, *10*(3). https://doi.org/10.34218/IJCET.10.3.2019.019

Waruwu, M. (2023). *Pendekatan Penelitian Pendidikan: Metode Penelitian Kualitatif, Metode Penelitian Kuantitatif dan Metode Penelitian Kombinasi*. *7*.

**LAMPIRAN**

filename = 'Hsimulasi.xlsx';

sheet = 'Sheet2';

data = readtable(filename, 'Sheet', sheet);

t = data.time;

x = data.x;

y = data.y;

l = data.lane;

p = data.type;

a = data.angle;

s = data.speed;

r = data.id;

K = 30; % Konstanta berbeda setiap lingkungan

start1 = 1;

%figure; % Membuat figure baru

Data\_t = unique(t);

Data\_p = unique(p);

Data\_l = unique(l);

% Inisialisasi variabel baru dengan zeros

selectedData = zeros(80, 3);

% A6 = 500; % Satuan Kbps

% B6 = 30;

% Mengambil 80 baris pertama dari kolom x, y, dan id

selectedData(:, 1) = data.x(1:80);

selectedData(:, 2) = data.y(1:80);

% Mengambil angka setelah karakter 'f\_'

id = str2double(extractAfter(data.id(1:80), 'f\_'));

% Mengisi kolom ketiga dari newVariable dengan data numerik

selectedData(:, 3) = id;

% Inisialisasi indeks t

t = 1;

% Maksimum iterasi yang diinginkan

maxIterations = height(data);

% Inisialisasi tabel untuk menyimpan hasil

result = table('Size', [80, 5], ...

'VariableTypes', {'double', 'double', 'string', 'double', 'double'}, ...

'VariableNames', {'t', 'd', 'id', 'x', 'y'});

% Inisialisasi matriks untuk menyimpan jarak antar titik

jarakAntarTitik = zeros(maxIterations, maxIterations);

% while t <= 80

% while t + 1 <= maxIterations && t <= 80

while t + 1 <= maxIterations

% Increment t

t = t + 1;

% Kalkulasi nilai d hanya untuk titik tertentu

d = sqrt((data.x(t) - data.x(t- 1)).^2 + (data.y(t) - data.y(t- 1)).^2);

% Menyimpan nilai t, d, id, x, dan y ke dalam result

result.t(t) = data.time(t);

result.d(t) = d;

result.id{t} = data.id{t};

result.x(t) = data.x(t);

result.y(t) = data.y(t);

result.RREPSN = zeros(height(result), 1);

result(result.t == 0, :) = [];

% Menyimpan jarak antar titik ke dalam matriks

jarakAntarTitik(t-1, t) = d;

jarakAntarTitik(t, t-1) = d;

% % Tambahkan kondisi untuk keluar dari loop

% if t >= height(data)

% break;

% end

end

% Inisialisasi variabel baru untuk menyimpan data

group = table('Size', [100, 1], ...

'VariableTypes', {'cell'}, ...

'VariableNames', {'Result'});

% Menggabungkan data t dan id menjadi data baru '*Sequence*' di tabel result

result.*Sequence* = strcat(string(result.id), '\_', string(result.t));

% Inisialisasi struktur untuk menyimpan jumlah kemunculan setiap ID pada setiap iterasi

id\_counts = containers.Map('KeyType', 'char', 'ValueType', 'double');

id\_count = containers.Map('KeyType', 'char', 'ValueType', 'double');

for t = 1:max(result.t)

% Mendapatkan ID yang muncul pada iterasi saat ini

ids\_current = unique(result.id(result.t == t));

% Loop melalui setiap ID yang muncul pada iterasi saat ini

for id\_idx = 1:numel(ids\_current)

id = ids\_current{id\_idx};

% Jika ID tidak ada dalam struktur id\_count, tambahkan dan atur nilai awalnya menjadi 0

if ~isKey(id\_count, id)

id\_count(id) = 0;

end

% Mendapatkan jumlah kemunculan ID pada iterasi sebelumnya

count\_prev = id\_count(id);

% Mendapatkan indeks ID pada iterasi saat ini

idx\_current = find(strcmp(result.id, id) & result.t == t);

% Memperbarui *Sequence* untuk ID pada iterasi saat ini dengan indeks unik yang tepat

for i = 1:numel(idx\_current)

% Mengubah tipe data SSN menjadi integer dan memulai pengurutan dari time 1

result.SSN(idx\_current(i)) = count\_prev + i;

end

% Mengupdate jumlah kemunculan ID

id\_count(id) = count\_prev + numel(idx\_current);

end

end

% Inisialisasi variabel

num*Node* s = height(unique(result));

validDValues = zeros(num*Node* s, num*Node* s);

% Tentukan jumlah baris yang ingin digunakan

jumlah\_baris = 313;

% Ambil sejumlah baris tertentu dari tabel result

data\_terbatas = result(1:jumlah\_baris, :);

% Mengambil jumlah unik dari kolom 'id' dalam tabel 'data\_terbatas' untuk mendapatkan jumlah *Node*

num*Node* s = numel(unique(data\_terbatas.*Sequence*));

% Menginisialisasi matriks validDValues dengan jarak antar *Node*

for i = 1:num*Node* s

for j = 1:num*Node* s

% Perhitungan jarak antar *Node* i dan j

validDValues(i, j) = sqrt((data\_terbatas.x(i) - data\_terbatas.x(j))^2 + (data\_terbatas.y(i) - data\_terbatas.y(j))^2);

end

end

% Inisialisasi AODV

status = '!';

dist = inf(1, num*Node* s);

next = zeros(1, num*Node* s);

% Inisialisasi status, dist, dan next

for i = 1:num*Node* s

if i == 1

status(i) = '!';

dist(i) = 0;

next(i) = 0;

else

status(i) = '?';

% Gunakan hasil perhitungan jarak dari tabel result

dist(i) = data\_terbatas.d(i);

next(i) = 1;

end

end

% Inisialisasi variabel lainnya

flag = 0;

temp = 0;

% Set goal*Node*

goal*Node* = 20; % Sesuaikan dengan *Node* tujuan

% Initialize variables to store ping information

pingResults = cell(num*Node* s, num*Node* s);

rrepsn = zeros(max(result.t), numel(unique(result.id)));

*Threshold*\_lower = -14;

*Threshold*\_upper = 14;

result.Status = repmat("Connected", height(result), 1);

% Main loop

while flag ~= 1 && temp < num*Node* s

temp = temp + 1; % Tambahkan iterasi

% Pilih *Node* dengan dist terkecil dan status '?'

[minDist, vert] = min(dist(status == '?'));

% Perbarui status

status(vert) = '!';

% Perbarui dist dan next untuk *Node* tetangga

for i = 1:num*Node* s

if status(i) == '?' && dist(i) > dist(vert) + validDValues(vert, i)

dist(i) = dist(vert) + validDValues(vert, i);

next(i) = vert;

% Simulasi RREQ hanya jika tidak dalam keadaan Timeout

if validDValues(vert, i) < 300

pingResults{vert, i} = 'Ping: *Reply* 100%';

% Log RREQ

disp(['*Node* ' num2str(vert) ' sends RREQ message to *Node* ' num2str(i)]);

% Simulasikan penerimaan RREQ dan kirimkan RREP

% Update RREPSN values

rrepsn(i) = rrepsn(i) + 1; % Tingkatkan nilai rrepsn untuk *Node* yang membalas

% Update tableSSN with RREPSN values

result.RREPSN(i) = rrepsn(i); % Update nilai RREPSN untuk *Node* yang membalas

disp(['*Node* ' num2str(i) ' sends RREP message (RREPSN=' num2str(rrepsn(i)) ') to *Node* ' num2str(vert)]);

if i == goal*Node*

flag = 1;

break;

end

else

% Jika mencapai Timeout, set hasil ping menjadi Timeout

pingResults{vert, i} = 'Timeout';

pingResults{vert, i} = ['*Node* ' num2str(vert) ' timeout to *Node* ' num2str(i)];

disp(pingResults{vert, i});

result.Status(vert) = "Timeout";

end

% Tambahkan kondisi untuk keluar dari loop jika goal*Node* tercapai

if i == goal*Node*

flag = 1;

break;

end

end

end

result.Difference = result.RREPSN - result.SSN;

result.Status(result.Difference < *Threshold*\_lower | result.Difference > *Threshold*\_upper) = "Disconnected";

result.Status(contains(result.Status, 'Timeout')) = "Timeout";

if all(status == '!')

flag = 1;

break;

end

end

% Check for *Node* s that initiated RREQ but did not receive RREP (Timeout)

% disp('Timeout Results:');

for i = 1:num*Node* s

initiatedRREQ = find(~cellfun('isempty', pingResults(i, :)));

% Initialize receivedRREP as an empty array

receivedRREP = [];

% Loop through each *Node* 's ping result at time t

for j = 1:num*Node* s

% Check if there's a ping result for the current *Node* at time t

if ~isempty(pingResults{i, j})

% Check if the ping result indicates a successful *Reply* at time t

if contains(pingResults{i, j}, 'Ping: *Reply* 100%')

receivedRREP = [receivedRREP j];

end

end

end

% Check if there are *Node* s that initiated RREQ but did not receive RREP

% if isempty(receivedRREP)

% % *Node* initiated RREQ but did not receive RREP (Timeout)

% disp(['*Node* ' num2str(i) ' tidak RREP Ping : Timeout']);

% end

end

% Inisialisasi variabel untuk menyimpan rute

i = goal*Node* ; % Ganti dengan goal*Node*

count = 1;

route(count) = goal*Node* ;

% Bangun rute dari *Node* terakhir ke *Node* pertama

while next(i) ~= 0 % Ganti dengan *Node* awal

count = count + 1;

route(count) = next(i);

i = next(i);

end

% result.Difference = result.RREPSN - result.SSN;

% % Set the *Threshold* for disconnect status

% *Threshold*\_lower = -14;

% *Threshold*\_upper = 14;

%

% % Initialize the 'Status' column as 'Connected'

% result.Status = repmat("Connected", height(result), 1);

%

% % Update 'Status' to 'Disconnected' if the difference is beyond the *Threshold*s

% result.Status(result.Difference < *Threshold*\_lower | result.Difference > *Threshold*\_upper) = "Disconnected";

% result.Status(contains(result.Status, 'Timeout')) = "Timeout";

% Membuat loop untuk mengecek setiap nilai t

for t = 1:max(result.t)

% Mendapatkan ID yang muncul pada iterasi saat ini

ids\_current = unique(result.id(result.t == t));

% Loop melalui setiap ID yang muncul pada iterasi saat ini

for id\_idx = 1:numel(ids\_current)

id = ids\_current{id\_idx};

% Jika ID tidak ada dalam struktur id\_counts, tambahkan dan atur nilai awalnya menjadi 0

if ~isKey(id\_counts, id)

id\_counts(id) = 0;

end

% Mendapatkan jumlah kemunculan ID pada iterasi sebelumnya

count\_prev = id\_counts(id);

% Mendapatkan indeks ID pada iterasi saat ini

idx\_current = find(strcmp(result.id, id) & result.t == t);

% Memperbarui *Sequence* untuk ID pada iterasi saat ini dengan indeks unik yang tepat

for i = 1:numel(idx\_current)

result.*Sequence*{idx\_current(i)} = [id, '\_', num2str(count\_prev + i)];

end

% Mengupdate jumlah kemunculan ID

id\_counts(id) = count\_prev + numel(idx\_current);

end

end

% Iterasi untuk t = 1 hingga 100

for t = 1:50

% Mengambil data dengan nilai 't' sesuai iterasi

resultTime = result(result.t == t, :);

% Perhitungan nilai d

if t > 1

d = sqrt((data.x(t) - data.x(t-1)).^2 + (data.y(t) - data.y(t-1)).^2);

else

d = 0;

end

% Jika data tidak mencapai 80 baris, tambahkan baris dengan nilai 0

if size(resultTime, 1) < 80

rowsTotal = 80 - size(resultTime, 1);

rowsZero = array2table(zeros(rowsTotal, width(resultTime)), 'VariableNames', resultTime.Properties.VariableNames);

resultTime = [resultTime; rowsZero];

end

% Simpan resultTime ke dalam group

group.Result{t} = resultTime;

% Hapus variabel yang tidak ingin ditampilkan di workspace

clear nonZeroDIdx rowsTotal rowsZero;

end

% Iterasi untuk t = 1 hingga 100

for t = 1:50

% Mengambil tabel dari dalam cell array

resultTableTime = group.Result{t};

% Menambahkan kolom warna ke dalam tabel hanya jika d > 0

resultTableTime.color = cell(height(resultTableTime), 1);

% Temukan indeks baris dengan nilai d terkecil dan terbesar

minD = find(resultTableTime.d == min(resultTableTime.d(resultTableTime.d > 0)), 1, 'first');

maxD = find(resultTableTime.d >= 300);

% Berikan warna hijau untuk nilai d terkecil jika d > 0

if ~isempty(minD)

resultTableTime.color{minD} = 'green';

end

% Berikan warna merah untuk nilai d terbesar jika d > 0

if ~isempty(maxD)

resultTableTime.color(maxD)= {'magenta'};

% Ubah status menjadi 'Timeout'

resultTableTime.Status(maxD) = {'Timeout'};

end

% Isi nilai biru hanya untuk baris dengan nilai d sama dengan 0

zeroDIdx = find(resultTableTime.d == 0);

% Hapus *Node* biru dengan nilai d = 0 dari hasil plot

resultTableTime(zeroDIdx, :) = [];

% Isi nilai biru untuk baris dengan nilai d tidak sama dengan 0 dan tidak memiliki warna

nonZeroDIdx = find(resultTableTime.d > 0 & cellfun('isempty', resultTableTime.color));

resultTableTime.color(nonZeroDIdx) = {'blue'};

% Menyimpan indeks baris dengan nilai d terkecil sebagai Head Cluster (warna hijau)

headClusterIdx = find(strcmp(resultTableTime.color, 'green'));

if ~isempty(headClusterIdx)

resultTableTime.color{headClusterIdx} = 'Head Cluster';

end

% Menghasilkan nilai pt dalam rentang [200, 300] berdasarkan t

pt = 50 + (t - 1) \* 10; % Pertambahan 10 setiap iterasi t

% Pastikan pt tidak melebihi 300

if pt > 100

pt = 100;

end

% Membuat kolom pt untuk setiap baris

resultTableTime.pt = repmat(pt, height(resultTableTime), 1);

% Mengatur semua nilai dalam rt menjadi 40

rt = repmat(20, height(resultTableTime), 1);

resultTableTime.rt = rt;

% Menyimpan tabel yang telah dimodifikasi ke dalam cell array

group.Result{t} = resultTableTime;

% Hapus variabel yang tidak ingin ditampilkan di workspace

clear nonZeroDIdx zeroDIdx;

clear headClusterIdx maxD minD;

end

% Inisialisasi variabel baru untuk warna pada result

result.color = cell(height(result), 1);

% Iterasi untuk t = 1 hingga 50

for t = 1:50

% Mengambil tabel dari dalam cell array

resultTableTimeSerangan = group.Result{t};

% Menambahkan kolom warna ke dalam tabel hanya jika d > 0

resultTableTimeSerangan.color = cell(height(resultTableTimeSerangan), 1);

% Temukan indeks baris dengan nilai d terkecil dan terbesar

minD = find(resultTableTimeSerangan.d == min(resultTableTimeSerangan.d(resultTableTimeSerangan.d > 0)), 1, 'first');

maxD = find(resultTableTimeSerangan.d >= 300);

% Berikan warna hijau untuk nilai d terkecil jika d > 0

if ~isempty(minD)

resultTableTimeSerangan.color{minD} = 'green';

end

% Berikan warna merah untuk nilai d terbesar jika d > 0

if ~isempty(maxD)

resultTableTimeSerangan.color(maxD)= {'magenta'};

% Ubah status menjadi 'Timeout'

resultTableTimeSerangan.Status(maxD) = {'Timeout'};

end

% Isi nilai biru hanya untuk baris dengan nilai d sama dengan 0

zeroDIdx = find(resultTableTimeSerangan.d == 0);

% Hapus *Node* biru dengan nilai d = 0 dari hasil plot

resultTableTimeSerangan(zeroDIdx, :) = [];

% Isi nilai biru untuk baris dengan nilai d tidak sama dengan 0 dan tidak memiliki warna

nonZeroDIdx = find(resultTableTimeSerangan.d > 0 & cellfun('isempty', resultTableTimeSerangan.color));

resultTableTimeSerangan.color(nonZeroDIdx) = {'blue'};

% Menyimpan indeks baris dengan nilai d terkecil sebagai Head Cluster (warna hijau)

headClusterIdx = find(strcmp(resultTableTimeSerangan.color, 'green'));

if ~isempty(headClusterIdx)

resultTableTimeSerangan.color{headClusterIdx} = 'Head Cluster';

end

% Menyimpan indeks baris dengan status "Disconnected" dan mengubah warna menjadi merah

disconnectedIdx = find(strcmp(resultTableTimeSerangan.Status, 'Disconnected'));

if ~isempty(disconnectedIdx)

resultTableTimeSerangan.color(disconnectedIdx) = {'red'};

end

% Check if the status is "Disconnected" and update SSN accordingly

disconnectedIdx = find(result.Status == "Disconnected");

for idx = 1:numel(disconnectedIdx)

% Generate a random RREPSN for disconnected *Node* s

result.RREPSN(disconnectedIdx(idx)) = randi([0, 1000000000]); % Assuming the range for RREPSN

end

% Inisialisasi matriks koneksi

resultTableTimeSerangan.koneksi = zeros(size(resultTableTimeSerangan, 1), size(resultTableTimeSerangan, 1));

% Mendapatkan indeks *Node* yang belum terkoneksi

unconnected*Node* sIdx = find(sum(resultTableTimeSerangan.koneksi, 2) == 0);

% Urutkan *Node* yang belum terkoneksi berdasarkan nilai d dari terkecil hingga terbesar

[~, sortedIdx] = sort(resultTableTimeSerangan.d(unconnected*Node* sIdx));

sortedUnconnected*Node* sIdx = unconnected*Node* sIdx(sortedIdx);

% Membuat koneksi ulang berdasarkan *Node* yang tidak terkoneksi yang sudah diurutkan

for i = 1:length(sortedUnconnected*Node* sIdx)

current*Node* = sortedUnconnected*Node* sIdx(i);

for j = (i+1):length(sortedUnconnected*Node* sIdx)

next*Node* = sortedUnconnected*Node* sIdx(j);

if resultTableTimeSerangan.d(next*Node* ) < 300 % Jika jarak antara *Node* saat ini dengan *Node* berikutnya kurang dari 300

if sum(resultTableTimeSerangan.koneksi(current*Node* , :)) < 2 && sum(resultTableTimeSerangan.koneksi(next*Node* , :)) < 2 % Pastikan setiap *Node* memiliki maksimal 2 koneksi

resultTableTimeSerangan.koneksi(current*Node* , next*Node* ) = 1;

resultTableTimeSerangan.koneksi(next*Node* , current*Node* ) = 1;

break; % Hanya satu koneksi yang perlu ditambahkan

end

end

end

end

% Nonaktifkan koneksi ke dan dari *Node* -*Node* merah

red*Node* sIdx = find(strcmp(resultTableTimeSerangan.color, 'red'));

if ~isempty(red*Node* sIdx)

for i = 1:length(red*Node* sIdx)

red*Node* = red*Node* sIdx(i);

resultTableTimeSerangan.koneksi(red*Node* , :) = 0; % Nonaktifkan koneksi ke *Node* lain

resultTableTimeSerangan.koneksi(:, red*Node* ) = 0; % Nonaktifkan koneksi dari *Node* lain

end

end

% Membuat koneksi ulang berdasarkan *Node* yang tidak terkoneksi dan bukan berwarna merah

for i = 1:size(resultTableTimeSerangan.koneksi, 1)

if sum(resultTableTimeSerangan.koneksi(i, :)) == 0 && ~strcmp(resultTableTimeSerangan.color(i), 'red') % Jika *Node* belum terkoneksi dengan siapa pun dan bukan berwarna merah

for j = 1:size(resultTableTimeSerangan.koneksi, 2)

if i ~= j && sum(resultTableTimeSerangan.koneksi(j, :)) < 2 && resultTableTimeSerangan.d(i) < 300 && resultTableTimeSerangan.d(j) < 300

if sum(resultTableTimeSerangan.koneksi(i, :)) < 2 && sum(resultTableTimeSerangan.koneksi(j, :)) < 2 % Pastikan setiap *Node* memiliki maksimal 2 koneksi

resultTableTimeSerangan.koneksi(i, j) = 1;

resultTableTimeSerangan.koneksi(j, i) = 1;

break; % Hanya satu koneksi yang perlu ditambahkan

end

end

end

end

end

% Menghubungkan *Node* "Disconnected" ke *Node* "Connected"

disconnected*Node* sIdx = find(strcmp(resultTableTimeSerangan.Status, 'Disconnected'));

connected*Node* sIdx = find(~strcmp(resultTableTimeSerangan.Status, 'Disconnected') & ~strcmp(resultTableTimeSerangan.color, 'red'));

for i = 1:length(disconnected*Node* sIdx)

disconnected*Node* = disconnected*Node* sIdx(i);

for j = 1:length(connected*Node* sIdx)

connected*Node* = connected*Node* sIdx(j);

if resultTableTimeSerangan.d(disconnected*Node* ) < 300 && resultTableTimeSerangan.d(connected*Node* ) < 300

if sum(resultTableTimeSerangan.koneksi(disconnected*Node* , :)) < 2 && sum(resultTableTimeSerangan.koneksi(connected*Node* , :)) < 2 % Pastikan setiap *Node* memiliki maksimal 2 koneksi

resultTableTimeSerangan.koneksi(disconnected*Node* , connected*Node* ) = 1;

resultTableTimeSerangan.koneksi(connected*Node* , disconnected*Node* ) = 1;

break; % Hanya satu koneksi yang perlu ditambahkan

end

end

end

end

% Menghasilkan nilai pt dalam rentang [200, 300] berdasarkan t

pt = 50 + (t - 1) \* 10; % Pertambahan 10 setiap iterasi t

% Pastikan pt tidak melebihi 300

if pt > 100

pt = 100;

end

% Membuat kolom pt untuk setiap baris

resultTableTimeSerangan.pt = repmat(pt, height(resultTableTimeSerangan), 1);

% Mengatur semua nilai dalam rt menjadi 40

rt = repmat(15, height(resultTableTimeSerangan), 1);

resultTableTimeSerangan.rt = rt;

% Set pt dan rt menjadi 0 untuk *Node* yang memiliki warna merah

if ~isempty(red*Node* sIdx)

for i = 1:length(red*Node* sIdx)

red*Node* = red*Node* sIdx(i);

resultTableTimeSerangan.pt(red*Node* , :) = -0.5;

resultTableTimeSerangan.rt(red*Node* , :) = -0.5;

end

end

% Menyimpan tabel yang telah dimodifikasi ke dalam cell array

group.ResultTime{t} = resultTableTimeSerangan;

% Hapus variabel yang tidak ingin ditampilkan di workspace

clear nonZeroDIdx zeroDIdx;

clear headClusterIdx maxD minD;

end

% % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %

% Inisialisasi warna untuk plotting

warna = {'blue', 'red', 'green', 'black', 'cyan', 'magenta', 'yellow', 'white'};

% Inisialisasi *delay* dan *throughput*

*delay* 1 = zeros(1, 100);

*throughput*1 = zeros(1, 100);

% Inisialisasi *delay* dan *throughput*

*delay* 2 = zeros(1, 100);

*throughput*2 = zeros(1, 100);

% Membuat plot untuk setiap nilai t dari 1 hingga 100

for t\_idx = 1:20

% Mengambil tabel dari dalam cell array untuk plot kedua

resultTableTimeSerangan = group.Result{t};

% Hitung jumlah *Node* merah

red*Node* Count = sum(strcmp(resultTableTimeSerangan.color, 'red'));

% Membersihkan figur pertama sebelum memplot iterasi berikutnya

figure(1);

clf;

axis([-50 350 -40 120]);

% title('Jalur PKU - *Node* Kendaraan & Head Cluster');

title(['Simulasi 1 Tanpa Serangan - Iterasi ', num2str(t\_idx)]);

xlabel('Data x');

ylabel('Data y');

grid on;

hold on;

% Membersihkan figur kedua sebelum memplot iterasi berikutnya

figure(2);

clf;

axis([-50 350 -40 120]);

% title(['Jalur PKU - *Node* Kendaraan & Malicious - Iterasi ', num2str(t\_idx)]);

title(['Simulasi 2 Serangan - Iterasi ', num2str(t\_idx), ' - Malicious *Node* s: ', num2str(red*Node* Count)]);

xlabel('Data x');

ylabel('Data y');

grid on;

hold on;

% Membersihkan figur *delay* sebelum memplot iterasi berikutnya

figure(3);

axis('auto');

% title('*Delay* ');

xlabel('Jumlah Kendaraan (s)');

ylabel('*Delay* (ms)');

grid on;

hold on;

% Membersihkan figur *throughput* sebelum memplot iterasi berikutnya

figure(4);

axis('auto');

% title('*Throughput*');

xlabel('Jumlah Kendaraan (s)');

ylabel('*Throughput* (kbps)');

grid on;

hold on;

% Mengambil tabel dari dalam cell array untuk plot pertama

resultTableTime = group.Result{t\_idx};

% Urutkan berdasarkan nilai d

[~, idxSorted] = sort(resultTableTime.d);

resultTableTime = resultTableTime(idxSorted, :);

for i = 1:size(resultTableTime, 1)

if strcmp(resultTableTime.color{i}, 'Head Cluster')

figure(1);

scatter(resultTableTime.x(i), resultTableTime.y(i), 100, 'green', 'X', 'LineWidth', 1.5); % Symbol X for Head Cluster

elseif strcmp(resultTableTime.color{i}, 'blue')

figure(1);

scatter(resultTableTime.x(i), resultTableTime.y(i), 64, 'blue', 'o', 'filled'); % Blue dots

elseif strcmp(resultTableTime.color{i}, 'magenta')

figure(1);

scatter(resultTableTime.x(i), resultTableTime.y(i), 64, 'magenta', 'o', 'filled'); % Magenta dots

end

% Plot garis antar *Node*

if i < size(resultTableTime, 1)

figure(1);

plot([resultTableTime.x(i), resultTableTime.x(i+1)], [resultTableTime.y(i), resultTableTime.y(i+1)], 'b--', 'LineWidth', 1);

end

end

%

% Menambahkan legenda untuk *Subplot* pertama

figure(1);

hold on;

h1 = scatter(NaN, NaN, 100, 'green', 'X', 'LineWidth', 1.5);

h2 = scatter(NaN, NaN, 64, 'blue', 'o', 'filled');

h3 = scatter(NaN, NaN, 64, 'magenta', 'o', 'filled');

h4 = scatter(NaN, NaN, 64, 'red', 'o', 'filled');

leg1 = legend([h1, h2], 'Head Cluster', '*Node* Kendaraan', 'Location', 'northeast');

set(leg1, 'Box', 'on');

hold off;

% Plot data pada *Subplot* pertama

figure(2);

xlabel('Data x');

ylabel('Data y');

clf;

grid on;

hold on;

% Mengambil tabel dari dalam cell array untuk plot kedua

resultTableTimeSerangan = group.ResultTime{t\_idx};

% Menentukan newHeadCluster berdasarkan nilai d terkecil yang tidak 'red'

minD = min(resultTableTimeSerangan.d(~strcmp(resultTableTimeSerangan.color, 'red')));

newHeadClusterIndex = find(resultTableTimeSerangan.d == minD, 1);

% for i = 1:size(resultTableTimeSerangan, 1)

% if i == newHeadClusterIndex

% figure(2);

% scatter(resultTableTimeSerangan.x(i), resultTableTimeSerangan.y(i), 100, 'g', 'X', 'LineWidth', 1.5);

% elseif strcmp(resultTableTimeSerangan.color{i}, 'red') || strcmp(resultTableTimeSerangan.color{i}, 'Malicious')

% figure(2);

% scatter(resultTableTimeSerangan.x(i), resultTableTimeSerangan.y(i), 64, 'r', 'filled');

% else

% figure(2);

% scatter(resultTableTimeSerangan.x(i), resultTableTimeSerangan.y(i), 64, 'b', 'filled');

% % else

% % figure(2);

% % scatter(resulttime.x(i), resulttime.y(i), 64, 'b', 'filled');

% end

% % Menggambar koneksi antar *Node*

% for j = i + 1:size(resulttime.koneksi, 2)

% if resulttime.koneksi(i, j) == 1 && ~strcmp(resulttime.color{i}, 'red') && ~strcmp(resulttime.color{j}, 'red')

% plot([resulttime.x(i), resulttime.x(j)], [resulttime.y(i), resulttime.y(j)], 'b--', 'LineWidth', 1);

% end

% end

% end

%

% Iterate over all *Node* s to plot them based on their properties

for i = 1:size(resultTableTimeSerangan, 1)

if i == newHeadClusterIndex

figure(2);

scatter(resultTableTimeSerangan.x(i), resultTableTimeSerangan.y(i), 100, 'g', 'X', 'LineWidth', 1.5);

elseif strcmp(resultTableTimeSerangan.color{i}, 'red') || strcmp(resultTableTimeSerangan.color{i}, 'Malicious')

figure(2);

scatter(resultTableTimeSerangan.x(i), resultTableTimeSerangan.y(i), 64, 'r', 'filled');

elseif strcmp(resultTableTimeSerangan.color{i}, 'magenta')

figure(2);

scatter(resultTableTimeSerangan.x(i), resultTableTimeSerangan.y(i), 64, 'm', 'filled');

else

figure(2);

scatter(resultTableTimeSerangan.x(i), resultTableTimeSerangan.y(i), 64, 'b', 'filled');

end

% Plot connections between *Node* s

for j = i + 1:size(resultTableTimeSerangan.koneksi, 2)

if resultTableTimeSerangan.koneksi(i, j) == 1

if strcmp(resultTableTimeSerangan.color{i}, 'red') || strcmp(resultTableTimeSerangan.color{j}, 'red')

% Draw connections involving red *Node* s with a different style

plot([resultTableTimeSerangan.x(i), resultTableTimeSerangan.x(j)], [resultTableTimeSerangan.y(i), resultTableTimeSerangan.y(j)], 'r--', 'LineWidth', 1.5);

else

% Draw connections between non-red *Node* s

plot([resultTableTimeSerangan.x(i), resultTableTimeSerangan.x(j)], [resultTableTimeSerangan.y(i), resultTableTimeSerangan.y(j)], 'b--', 'LineWidth', 1);

end

end

end

end

% % Plot garis antar *Node* berdasarkan koneksi

% for i = 1:size(resultTableTimeSerangan.koneksi, 1)

% for j = i+1:size(resultTableTimeSerangan.koneksi, 2)

% if resultTableTimeSerangan.koneksi(i, j) == 1 % Jika ada koneksi antara *Node* i dan *Node* j

% d = sqrt((resultTableTimeSerangan.x(i) - resultTableTimeSerangan.x(j))^2 + (resultTableTimeSerangan.y(i) - resultTableTimeSerangan.y(j))^2);

% if d <= 300

% plot([resultTableTimeSerangan.x(i), resultTableTimeSerangan.x(j)], [resultTableTimeSerangan.y(i), resultTableTimeSerangan.y(j)], 'b--', 'LineWidth', 1);

% else

% plot([resultTableTimeSerangan.x(i), resultTableTimeSerangan.x(j)], [resultTableTimeSerangan.y(i), resultTableTimeSerangan.y(j)], 'r--', 'LineWidth', 1);

% end

% end

% end

% end

figure(2);

hold on;

h1 = scatter(NaN, NaN, 100, 'green', 'X', 'LineWidth', 1.5);

h2 = scatter(NaN, NaN, 64, 'blue', 'o', 'filled');

h3 = scatter(NaN, NaN, 64, 'red', 'o', 'filled');

leg2 = legend([h1, h2, h3], 'Head Cluster', '*Node* Kendaraan', 'Malicious', 'Location', 'northeast');

set(leg2, 'Box', 'on');

hold off;

% Perhitungan *delay* dan *throughput* pada detik t\_idx untuk group.Result

total\_pt\_1 = sum(group.Result{t\_idx}.pt);

total\_rt\_1 = sum(group.Result{t\_idx}.rt);

% *Delay* 1 = total\_pt\_1 / max(total\_rt\_1, 1);

*Delay* 1 = total\_pt\_1 / total\_rt\_1;

% Perhitungan *throughput* pada detik t\_idx untuk group.Result

paket\_diterima\_1 = group.Result{t\_idx}.rt; % paket data yang diterima dalam kb

waktu\_pengiriman\_1 = group.Result{t\_idx}.pt; % waktu pengiriman dalam detik

*Throughput*1 = paket\_diterima\_1 ./ max(waktu\_pengiriman\_1, 1);

% Perhitungan *delay* pada detik t\_idx untuk group.ResultTime

total\_pt\_2 = sum(group.ResultTime{t\_idx}.pt);

total\_rt\_2 = sum(group.ResultTime{t\_idx}.rt);

% *Delay* 2 = total\_pt\_2 / max(total\_rt\_2, 1);

*Delay* 2 = total\_pt\_2 / total\_rt\_2;

% Perhitungan *throughput* pada detik t\_idx untuk group.ResultTime

paket\_diterima\_2 = group.ResultTime{t\_idx}.rt; % paket data yang diterima dalam kb

waktu\_pengiriman\_2 = group.ResultTime{t\_idx}.pt; % waktu pengiriman dalam detik

*Throughput*2 = paket\_diterima\_2 ./ max(waktu\_pengiriman\_2, 1);

% Menyimpan hasil perhitungan *delay* dan *throughput*

% *delay* 1(t\_idx) = *Delay* 1;

*delay* 1(t\_idx) = mean(*Delay* 1);

*throughput*1(t\_idx) = mean(*Throughput*1); % Menggunakan mean untuk mendapatkan nilai rata-rata jika ada beberapa elemen

% *delay* 2(t\_idx) = *Delay* 2;

*delay* 2(t\_idx) = mean(*Delay* 2);

*throughput*2(t\_idx) = mean(*Throughput*2); % Menggunakan mean untuk mendapatkan nilai rata-rata jika ada beberapa elemen

% Plot *delay*

figure(3);

plot(1:t\_idx, *delay* 1(1:t\_idx), 'g.-'); % Plot *delay* dari figure 1

hold on;

plot(1:t\_idx, *delay* 2(1:t\_idx), 'r.-');

h\_*delay* = legend('Normal', 'Under *Attack*', 'Location', 'northeast');

set(h\_*delay* , 'Box', 'on'); % Menghilangkan kotak di sekitar legenda

hold off;

% Plot *throughput*

figure(4);

plot(1:t\_idx, *throughput*1(1:t\_idx), 'g.-'); % Plot *throughput* dari figure 1

hold on;

plot(1:t\_idx, *throughput*2(1:t\_idx), 'r.-');

h\_*throughput* = legend('Normal', 'Under *Attack*', 'Location', 'northeast');

set(h\_*throughput*, 'Box', 'on'); % Menghilangkan kotak di sekitar legenda

hold off;

pause(0.0);

end

hold off;

% Tampilkan hasil rute

disp('AODV Route:');

disp(route);