SIMULASI SERANGAN BLACK HOLE PADA JARINGAN MANET MENGGUNAKAN NS-3

Disusun Oleh:

N a m a : Alfi Syahri Nuzula

NIM : 13523008

JURUSAN TEKNIK INFORMATIKA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM INDONESIA

2018

HALAMAN PENGESAHAN DOSEN PEMBIMBING

SIMULASI SERANGAN BLACK HOLE PADA JARINGAN MANET MENGGUNAKAN NS-3

TUGAS AKHIR

Disusun Oleh:

N a m a : Alfi Syahri Nuzula

NIM : 132523008

Yogyakarta, 3 Maret 2018

Pembimbing,

(Yudi Prayudi S.Si., M.Kom.)

HALAMAN PENGESAHAN DOSEN PENGUJI

SIMULASI SERANGAN BLACK HOLE PADA JARINGAN MANET MENGGUNAKAN NS-3

TUGAS AKHIR

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik Informatika
di Fakultas Teknologi Industri Universitas Islam Indonesia
Yogyakarta, 3 Maret 2018

Tim Penguji

Yudi Prayudi, S.Si.,M.Kom.

Anggota 1

Ari Sujarwo, S.Kom., MIT.

Anggota 2

Almed Hamzah, S.T.,M.Eng.

Mengetahui,
Ketua Jurusan Teknik Informatika
Fakultas Teknologi Industri
Universitas Islam Indonesia

(Hendrik, S.T., M.Eng.)

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Alfi Syahri Nuzula

NIM : 13523008

Tugas akhir dengan judul:

SIMULASI SERANGAN BLACK HOLE PADA JARINGAN MANET MENGGUNAKAN NS-3

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila dikemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung resiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 3 Maret 2018

(Alfi Syahri Nuzula)

HALAMAN PERSEMBAHAN

Tugas akhir ini saya persembahkan untuk orang-orang yang saya hormati, saya cintai, dan saya sayangi.

Kepada kedua orang tua saya, Ayahku Bambang Suharyanto dan Ibuku Muzaidah yang selalu memberi semangat, dukungan moril dan materi dan doa yang tiada putusnya pada tiap sholatnya hingga penulis dapat menyelesaikan tugas akhir ini. Semoga Alloh Subhanahu wa Ta'ala selalu melindungi dan memberi kesehatan utuk ayah dan ibu.

Untuk seluruh teman-teman Teknik Informatika angkatan 2013 terutama teman-teman kelas A yang tidak bisa penulis sebutkan satu persatu yang selalu tanya skripsi sampai dimana, terima kasih karena pertanyaan kalian memberi semangat saya untuk bisa lebih cepat menyelesaikan tugas akhir ini.

Untuk teman-teman Ranger, yang selalu bikin tertawa setiap nongkrong bareng dari semester 1 dimanapun itu, terima kasih bro sudah menghibur saat suntuk mengerjakan tugas akhir ini. Semoga kalian cepat menyusul saya.

Untuk teman-teman KKN Unit 294, terima kasih juga sering memberi semangat penulis untuk cepat menyelesaikan tugas akhir ini dengan cara kalian yang kadang membuat penulis dongkol sendiri. Saya mengaku kalah guys dari kalian semua, saya yang paling terakhir lulus. Dimanapun kalian berada sekarang, semoga komunikasi tetap terjalin ya guys. Aamiin

Terima kasih banyak semuanya.

HALAMAN MOTO

"Jika kita belum mampu berlomba dengan orang sholeh meningkatkan kebaikan, sebaiknya berlomba dengan para pendosa untuk memperbaiki diri bertaubat dan membuat lebih baik lagi dalam kehidupan."

(Ustadz Adi Hidayat, Lc., MA)

KATA PENGANTAR

Assalamu'alaikum warahmatullahi wabarakatuh

Alhamdulillah, segala puji syukur penulis ucapkan kepada Alloh Subhanahu wa Ta'ala, untuk segala rahmat dan nikmatnya yang diberikannya. Sholawat dan salam tak lupa penulis haturkan kepada Rasulullo nabi Muhammad Shallallahu'alaihi Wasallam sehingga penulis dapat menyelesaikan tugas akhir ini yang berjudul "Simulasi Serangan Black Hole pada Jaringan MANET Menggunakan NS-3".

Laporan tugas akhir ini disusun sebagai salah satu syarat yang harus dipenuhi untuk memperoleh gelar sarjana di jurusan Teknik Informatika, Fakultas Teknologi Industri, Universitas Islam Indonesia. Dalam penyelesaian laporan tugas akhir ini, penulis menemui beberapa kesulitan dan mendapatkan pengetahuan baru dengan bantuan dan bimbingan dari berbagai pihak. Untuk itu penulis mengucapkan banyak terima kasih kepada :

1. Kedua orang tua penulis yang selalu memberikan dukungan moril, materil, motivasi dan doa yag tiada putusnya.
2. Bapak Yudi Prayudi, S.Si., M.Kom., selaku dosen pembimbing tugas akhir yang banyak memberikan bimbingan selama pengerjaan laporan tugas akhir ini.
3. Admin dan member Google Group ns-3-users yang banyak memberikan petunjuk bagi penulis untuk menggunakan Network Simulator 3 dan menyelesaikan permasalahan saat menjalakan simulasi black hole.
4. Seluruh teman-teman jurusan Teknik Informatika angkatan 2013 yang tidak bisa penulis sebutkan satu persatu atas bantuan dan dukungannya.

Penulis menyadari laporan tugas akhir ini, masih jauh dari kata sempurna karena keterbatasan pengetahuan dan pengalaman penulis. Oleh karena itu kritik dan saran sangat diperlukan bagi penulis agar penelitian ini dapat berguna bagi semua pihak yang membutuhkan.

Wassalamu'alaikum warahmatullahi wabarakatuh

Yogyakarta, 3 Maret 2018

(Alfi Syahri Nuzula)

SARI

MANET merupakan teknologi jaringan nirkabel yang terdiri dari beberapa node yang dapat bergerak secara dinamis, sehingga dapat membentuk topologi jaringan yang berbeda-beda. Kelebihan dari MANET ini tidak diperlukannya infrastuktur dalam pembangunannya sehingga potensial untuk diimplementaikan pada daerah-daerah terpencil, atau pada daerah-daerah yang sedang dilanda bencana alam yang merusakkan infrastruktur komunikasi di daerah tersebut, dan membuat MANET dapat dimanfaatkan sebagai alat komunikasi sementara. Sementara kelemahan jaringan MANET adalah, jaringan ini sangat rentan dengan serangan jaringan salah satunya adalah serangan black hole.

Black hole adalah sebuah serangan dalam jaringan komputer yang akan menghapus paket yang dikirimkan node pengirim yang mengakibatkan paket tersebut tidak akan sampai ke node tujuan. Node ini mampu masuk ke dalam jaringan dan menyamar sebagai node normal sehingga dapat membuat node pengirim melewatkan paketnya melalui node ini. Pada penelitian ini dibahas apa pengaruh black hole terhadap kualitas jaringan yang diserangnya. Penelitian ini menggunakan Network Simulator 3 untuk membuat simulasi jaringan yang terserang black hole dan penelitian ini menggunakan *routing protocol* AODV serta beberapa skenario untuk mengetahui lebih banyak pengaruh black hole pada kualitas jaringan yang diserangnya. Skenario yang diujikan pada penelitian ini adalah skenario 4 node dengan 1 dan 2 node black hole, skenario 20 node dengan 1 dan 2 node black hole, skenario 50 node dengan 1 dan 2 node black hole.

Dari skenario simulasi yang diujikan, node black hole pada skenario 4 node membuat *throughput* dan *delay* menjadi menurun dan *packet loss* menjadi tinggi dibandingkan dengan jaringan yang tidak terserang node black hole. Sedangkan pada skenario 20 dan 50 node, pengaruh black hole secara signifikan terjadi pada kondisi 1 node black hole di jaringan yang uji, sedangkan pada kondisi 2 node black hole pada jaringan, pengaruh black hole terhadap *throughput* dan *packet loss* lebih baik dibandingkan dengan kondisi 1 node black hole. Kesimpulan yang dapat diambil dari penelitian ini adalah black hole dapat menurunkan kualitas jaringan bergantung dari topologi dan letak blackhole itu sendiri. Semakin padat node dalam jaringan, dapat mengurangi pengaruh black hole karena node pengirim akan memiliki banyak pilihan rute yang dapat dilalui untuk mengirimkan paketnya tanpa melalui node black hole.

Kata kunci: MANET, AODV, Network Simulator 3, Black Hole.

GLOSARIUM

Bandwidth	Kapasitas maksimal jalur komunikasi yang dapat digunakan untuk mengirim data dalam hitungan detik.
Black Hole	Sebuah serangan dalam jaringan komputer yang akan menghapus data yang dikirimkan pengirim tanpa memberitahukan pengirim bahwa data yang dikirimkan tidak sampai ke tujuan.
Broadcast	Metode untuk mengirimkan data ke banyak tujuan secara sekaligus, tidak memperdulikan data yang dikirimkan sampai ke tujuan atau tidak.
Channel	Media yang digunakan Network Simulator 3 untuk mengirimkan data dalam jaringan.
Delay	Waktu keterlambatan yang terjadi karena proses pengiriman data dari titik pengirim ke titik tujuan.
Drop	Dihapus.
Freeware	Perangkat lunak gratis yang dapat digunakan tanpa lisensi khusus.
Hop	Titik-titik yang dilalui data dari pengirim ke titik tujuan
Mobile	Dapat berpindah-pindah ke berbagai tempat.
Node	Perangkat dalam jaringan komputer.
Packet Loss	Data yang hilang pada saat pengiriman dan tidak sampai ke titik tujuan.
Terminal	Aplikasi di sistem operasi ubuntu untuk mrngakses sistem.
Throughput	Kapasitas jalur sebenarnya yang digunakan data yang dikirimkan dari titik pengirim menuju titik tujuan.
Transmisi	Proses pengiriman data dari satu titik ke titik yang lain.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Informasi adalah kebutuhan yang penting saat ini. Kecepatan pertukaran informasi sangat dibutuhkan karena kebutuhan masyarakat akan informasi saat ini. Terlebih informasi yang beredar harus mampu menjangkau ke seluruh kalangan masyarakat, tak terkecuali masyarakat yang ada di daerah terpencil maupun daerah-daerah yang sedang terjadi konflik militer ataupun daerah yang sedang terkena bencana alam. Jaringan *Mobile AdHoc Network* **(**MANET) merupakan jaringan nirkabel yang tidak menggunakan infrastruktur. Sehingga jaringan MANET cocok digunakan jika terjadi bencana alam yang merusakkan infrastruktur telekomunikasi, atau daerah terpencil yang tidak memiliki infrastruktur telekomunikasi. Jaringan MANET adalah jaringan yang terdiri dari beberapa node yang saling berkoordinasi dan berkomunikasi satu sama lain. Setiap node memiliki keterbatasan jangkauan sehingga digunakan konsep *multi-hop forwarding* dimana setiap node beroperasi seperti sebuah *router* yang dapat meneruskan paket node lain pada jaringan. MANET diimplementasikan untuk komunikasi militer, tim penolong saat bencana, kepolisian, dan sebagai alat komunikasi saat infrastuktur komunikasi rusak akibat bencana alam. (Pratomo & Hizburrahman, 2015).

Setiap node pada MANET memiliki *transmitter* dan *receiver wireless* yang menggunakan antena atau sejenisnya yang bersifat *omnidirectional (broadcast), highly directional (point to point)* sehingga MANET memungkinkan untuk diatur arahnya atau mengkombinasikan kemampuannya tersebut. MANET mempunyai topologi yang dinamis, karena node pada MANET dapat bergerak kemana saja. Topologi yang berbentuk hubungan antara hop ke hop dapat berubah secara acak dan terjadi secara terus menerus tanpa adanya batasan waktu, sehingga berpengaruh pada susunan topologi jaringan. Selain karakteristik diatas, MANET juga memiliki karakteristik bersifat otonomi yaitu setiap node pada MANET berperan sebagai *end-user* sekaligus sebagai *router* yang mencari *route-path* yang akan dipilih. *Bandwidth* yang terbatas dibandingkan jaringan yang menggunakan kabel, energi yang terbatas karena node pada MANET bersifat *mobile* sehingga dipastikan menggunakan tenaga baterai untuk beroperasi dan keamanan yang rendah. (Khozaimi, 2013)

Aspek yang penting pada MANET adalah *routing protocol*, yang mana protokol inilah yang mengatur pencarian rute dalam jaringan tersebut. Terdapat 2 jenis *routing protocol* untuk

jaringan *AdHoc* yaitu *table driven* yang bersifat proaktif dan *on demand* yang bersifat reaktif. Beberapa contoh diantaranya adalah DSDV, OLSR, TBRPF, CGSR (*table driven)* dan AODV, DSR, LMR, TORA, ABR *(on demand)* dan ZRP (*Hybrid*). Masing-masing *routing protocol* tersebut memiliki karakteristik yang berbeda beda berdasarkan cara kerjanya.

Alasan keamanan saat melakukan transmisi menjadi hal penting yang perlu diperhatikan pada jaringan MANET. Jaringan MANET lebih rentan terhadap serangan, salah satu contohnya adalah black hole. Black hole adalah serangan pada MANET yang akan men*drop* semua paket yang dikirimkan oleh node *sender*. Node black hole meng*advertise* dirinya sebagai node yang rute tempuhnya paling pendek untuk menuju node *receiver* sehingga node *sender* akan mengirimkan paketnya melalui node black hole. Ketika paket sampai ke node black hole, paket tersebut tidak di teruskan ke node selanjutnya akan tetapi paket tersebut di *drop* dan tidak akan sampai ke node *receiver*. Hal ini akan sangat berbahaya jika paket yang dikirimkan berisi informasi penting ataupun informasi *urgent*. Untuk itu diperlukan upaya pencegahan agar paket yang kita kirimkan tidak terjebak ke dalam node black hole.

Tugas akhir ini akan mensimulasikan bagaimana serangan black hole bekerja menggunakan network simulator 3 (NS-3). Topik tugas akhir ini diusulkan diharapkan agar bisa menjadi upaya antisipasi agar jaringan MANET tidak terkena serangan black hole, sehingga dapat mengurangi kerugian yang akan didapatkan jika terkena serangan black hole.

1.2 Rumusan Masalah

Berdasarkan latar belakang masalah diatas, maka penulis dapat merumuskan masalah sebagai berikut :

a. Bagaimana serangan black hole bekerja pada jaringan MANET.
b. Bagaimana cara membuat model simulasi jaringan dengan menggunakan simulator NS-3.
c. Bagaimana menganalisa hasil pengukuran parameter *throughput, delay* dan *packet loss.*

1.3 Batasan Masalah

Dalam pengerjaan skripsi ini terdapat batasan masalah agar pengerjaan lebih terarah. Beberapa batasan masalah di penelitian ini adalah sebagai berikut:

a. *Routing protocol* yang digunakan adalah protokol *adhoc on demand distance vector (AODV).*
b. Pemodelan simulasi menggunakan *software* Network Simulator 3 (NS-3) versi 3.25
c. *Software* simulator jaringan NS-3 dijalankan pada sistem operasi ubuntu versi 16.04 LTS.

d. Node yang diujikan dalam simulasi ini sejumlah 4, 20, 50 node dengan 1 dan 2 node black hole.
e. Seluruh node yang diujikan dibuat statis
f. Parameter yang di analisa adalah *throughput, delay* dan *packet loss.*
g. Penelitian ini hanya untuk mengetahui pengaruh black hole pada *throughput, delay* dan *packet loss.*

1.4 Tujuan Penelitian

Tujuan dilakukannya penelitian ini adalah untuk mengetahui bagaimana serangan black hole mempengaruhi nilai QoS (*Quality of Service)* jaringan yaitu *throughput, delay* dan *packet loss* dalam suatu jaringan melalui simulasi yang menggunakan Network Simulator 3.

1.5 Manfaat Penelitian

Manfaat dari dilakukannya penelitian ini diantaranya adalah diharapkan dapat mengantisipasi serangan black hole dalam jaringan MANET dan meningkatkan keamanan terhadap serangan black hole pada jaringan MANET yang dibangun.

1.6 Metodologi Penelitian

Dalam menyelesaikan penelitian ini, diperlukan metode untuk mempermudah penulis menyelesaikan penelitian ini. Metode penelitian juga dapat membuat penelitian lebih terarah sehingga mendapatkan hasil yang diinginkan. Metode-metode tersebut antara lain:

a. Studi pustaka: Tahap ini dilakukan untuk mencari beberapa materi dan referensi tentang black hole dan Network Simulator 3 yang berkaitan dengan penelitian ini. Pustaka yang digunakan dapat berupa jurnal ilmiah, artikel maupun tugas akhir.
b. Analisa kebutuhan: Tahap ini dilakukan untuk menentukan apa saja yang dibutuhkan untuk mengerjakan penelitian ini, baik kebutuhan perangkat keras maupun perangkat lunak yang dibutuhkan.
c. Perancangan Simulasi: Pada tahap ini dilakukan untuk merancang simulasi, diagram alur dan parameter yang akan dihitung. Kemudian rancangan tersebut ditulis dalam baris kode bahasa pemrograman C++, agar dapat dijalankan di Network Simulator 3.
d. Penetapan parameter: Parameter digunakan sebagai nilai acuan dalam menjalankan simulasi. Nantiya nilai pada parameter akan dianalisa perbandingannya antara jaringan yang terkena serangan black hole dengan yang tidak terkena serangan black hole.

e. Implementasi simulasi: Tahap ini adalah tahapan membuat simulasi jaringan berdasarkan perancangan yang dibuat sebelumnya.

f. Analisa hasil: Setelah simulasi jaringan di jalankan, maka akan muncul nilai hasil pengukuran dari parameter yang di tetapkan, berupa nilai *throughput*, *delay* dan *packet loss.* Nantinya akan dibandingkan pengaruh parameter terhadap nilai *throughput*, *delay* dan *packet loss.*

1.7 Sistematika Penulisan

Sistematika penulisan disusun untuk mempermudah pembaca dalam membaca tugas akhir ini. Penulis menyusun sistematika tugas akhir ini sebagai berikut:

a. **BAB I PENDAHULUAN**

Bab ini berisi tentang latar belakang permasalahan, rumusan masalah, batasan masalah yang akan dibahas, tujuan dan manfaat dari penelitian ini, metode penelitian dan sistematika penulisan.

b. **BAB II LANDASAN TEORI**

Bab ini membahas beberapa teori yang penulis gunakan sebagai landasan untuk membuat simulasi serangan black hole menggunakan network simulator 3 pada penelitian ini.

c. **BAB III METODOLOGI**

Bab ini berisi metode yang digunakan untuk menyelesaikan penelitian ini. Bab ini juga membahas tentang kebutuhan yang diperlukan untuk menyelesaikan penelitian ini dan merancang simulasi jaringan yang akan dibuat menggunakan network simulator 3.

d. **BAB IV HASIL DAN PEMBAHASAN**

Bab ini membahas hasil implementasi dari simulasi jaringan yang dibuat. Pada bab ini dibahas pengaruh nilai paramer terhadap *throughput*, *packet loss* dan *delay*.

e. **BAB V KESIMPULAN**

Bab ini berisi kesimpulan yang diperoleh dari seluruh tahapan yang dilalui dalam penelitian ini. Berisi pula saran yang dapat digunakan untuk penelitian selanjutnya.

BAB II
LANDASAN TEORI

2.1 Jaringan Nirkabel

Jaringan nirkabel (*wireless)* adalah teknologi komunikasi untuk menghubungkan dua atau lebih alat komunikasi tanpa menggunakan kabel sebagai media transmisi. Jaringan nirkabel menggunakan gelombang frekuensi tertentu untuk dapat saling berkomunikasi. Jaringan nirkabel memiliki beberapa macam teknologi yang penggunaannya disesuaikan dengan kebutuhan. Diantaranya adalah *wireless personal area network, wireless local area network, wireless mesh network, wireless wide area network, point to point,* dan *mobile adhoc network* (Sanjaya, 2015). Teknologi jaringan *wireless* saat ini sudah banyak digunakan untuk mempermudah pekerjaan manusia. Contohnya telepon selular. Jika jaman dulu orang masih harus menggunakan telepon kabel untuk saling berkomunikasi. Untuk membangun jaringan komunikasi menggunakan kabel, diperlukan biaya yang sangat besar, karena harus menggali tanah untuk menanam kabel komunikasinya. Setelah muncul teknologi jaringan nirkabel, maka pembangunan jaringan komunikasi dapat lebih murah, karena tidak diperlukan lagi menggali tanah untuk menanam kabel komunikasi. Dengan keunggulan jaringan nirkabel tersebut, teknologi ini sangat cocok diterapkan untuk daerah terpencil yang sulit di jangkau. Tetapi ada kelemahan pada teknologi jaringan nirkabel, diantaranya komunikasi yang kurang stabil dibandingkan dengan jaringan kabel dan kecepatan transmisi yang lebih rendah dari kecepatan transmisi jaringan kabel (Ali Pangera, 2008).

Namun saat ini kecepatan jaringan nirkabel sudah semakin meningkat sejak pertama kali dikembangkan, terdapat beberapa standar yang dibuat oleh Institute of Electrical and Electronic Engineers (IEEE), diantaranya adalah:

a. **IEEE 802.11**: Standar yang ditetapkan oleh IEEE pada tahun 1997 yang bekerja pada frekuensi 2,4 Ghz hanya mampu mendukung *bandwith* jaringan sebesar 2 Mbps saja.
b. **IEEE 802.11b**: Pada tahun 1999 IEEE membuat standar baru yang mampu mendukung *bandwith* sebesar 11 Mbps, dan masih bekerja pada frekuensi 2,4 Ghz. Jarak maksimal yang bisa dijangkau dengan standar ini adalah sekitar 45 meter untuk area dalam ruangan dan 90 meter untuk area luar ruangan (Asrullah, 2015).
c. **IEEE 802.11a**: Saat IEEE sedang mengembangkan standar 802.11b, secara bersamaan mereka juga mengambangkan standar 802.11a. standar ini sudah mampu mendukung

bandwith sebesar 54 Mbps tetapi bekerja pada frekuensi 5 Ghz yang artinya standar ini tidak kompatibel dengan 802.11b karena bekerja pada frekuensi yang berbeda. Jarak maksimal yang bisa dijangkau dengan standar ini adalah sekitar 15 meter untuk area dalam ruangan dan 30 meter untuk area luar ruangan (Asrullah, 2015).

d. **IEEE 802.11g**: Standar ini ditetapkan pada tahun 2002. Standar ini menyatukan kemampuan pada standar 802.11a dan 802.11b. Standar ini mampu mendukung *bandwith* sebesar 54 Mbps dan bekerja pada frekuensi 2,4 Ghz.

e. **IEEE 802.11n**: Standar yang ditetapkan pada tahun 2009 dikembangkan untuk memperbaiki kualitas dari standar 802.11g. dengan kemampuan *bandwith* sebesar 300 Mbps dan jangkauan sinyal yang lebih baik dibandingkan dengan standar *wireless* sebelumnya. Standar ini juga mampu bekerja pada 2 frekuensi yaitu frekuensi 2,4 Ghz dan 5 Ghz.

f. **IEEE 802.11ac**: Standar baru yang mampu memanfaatkan teknologi *wireless dual band* bekerja pada dua frekuensi dan kemampuan mendukung *bandwith* sebesar 450 Mbps pada frekuensi 2,4Ghz dan *bandwith* sebesar 1300 Mbps pada frekuensi 5 Ghz.

(Nusanet, 2016).

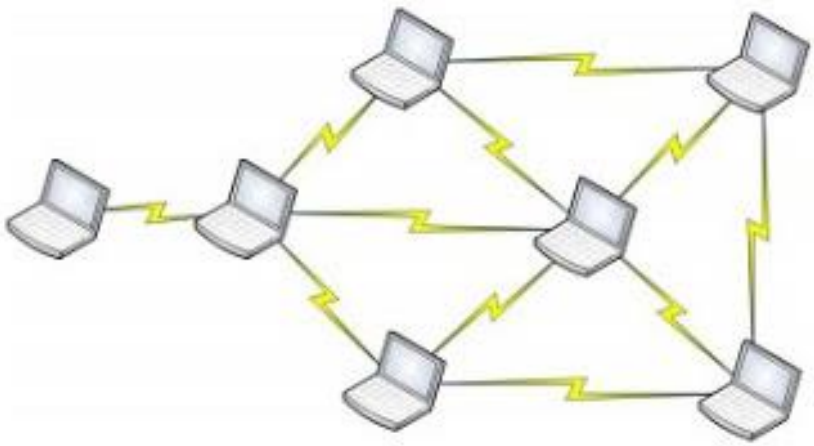
Gambar 2.1 Jaringan Nirkabel

2.2 *Mobile Adhoc Network* (MANET)

Mobile adhoc Network (MANET) adalah jaringan nirkabel yang terdiri dari beberapa node yang bersifat *mobile* sehingga dapat membentuk topologi yang berbeda-beda. Jaringan MANET tidak membutuhkan infrastruktur, tiap-tiap perangkat dapat bertindak seperti *router* yang dalam hal ini akan disebut dengan "node", sehingga jaringan MANET potensial diterapkan pada wilayah yang sedang terkena bencana alam, konflik militer dan kondisi darurat lainnya (Irawan, 2011). MANET memiliki beberapa karakteristik seperti berikut ini:

a. *Multiple Wireless link*: Setiap node mampu saling berhubungan dengan node lainnya.
b. *Limited Resource*: Jaringan MANET memiliki daya dan kapasitas memori yang terbatas.
c. *Dynamic Topology*: Tiap-tiap node bersifat *mobile*, sehingga topologi jaringan yang terbentuk dapat berubah-ubah sesuai dengan jumlah node dan pergerakkannya.
d. *Low Security*: Jaringan MANET menggunakan gelombang radio untuk saling menghubungkan antar node sehingga keamanan dalam jaringan MANET rendah.

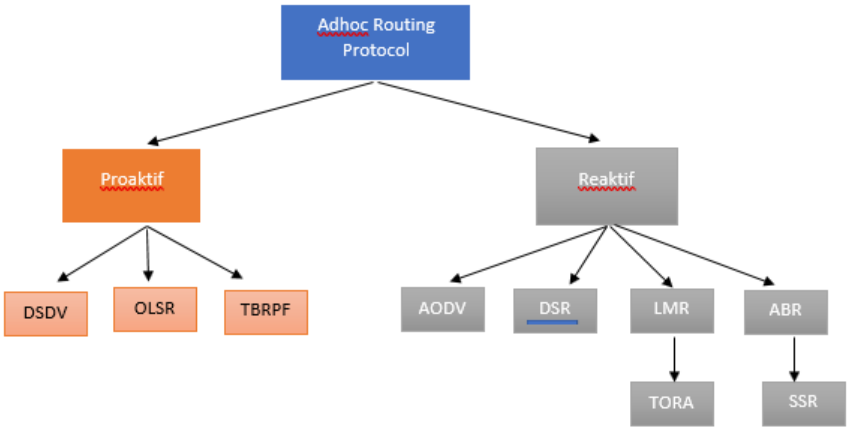
(Chitkara & Ahmad, 2014).



Gambar 2.2 Jaringan MANET

MANET memiliki dua jenis *routing protocol* yang berfungsi untuk mengatur jalur yang akan digunakan untuk mengirimkan paket ke node tujuan, yaitu protokol reaktif dan protokol proaktif. Protokol reaktif artinya node membentuk sebuah rute dari node asal ke node tujuan berdasarkan permintaan node asal saja. Sehingga pencarian rute hanya terjadi jika node asal ingin berkomunikasi dengan node tujuan saja. Protokol reaktif akan mencari rute dengan cara meyebarkan paket *request* ke seluruh node yang ada di dalam jaringan, sehingga memungkinkan jaringan akan menjadi penuh. Contoh *routing protocol* reaktif adalah *Adhoc on-demand Distance Vector* (AODV), *Dynamic Source Routing* (DSR), *Associatively Based*

Routing (ABR), *Temporary Ordered Routing Algorithm* (TORA), *Signal Stability Routing* (SSR) dan *Associativity Based Routing* (ASR). Sedangkan protokol proaktif artinya setiap node menyimpan tabel yang berisi informasi rute ke setiap node yang diketahuinya. Informasi rute tersebut akan diperbaharui setiap ada perubahan pada rutenya. Protokol proaktif akan berusaha terus untuk mengevaluasi rute dalam jaringan. Jika diketahui ada paket yang harus diteruskan, maka rute yang harus dilalui sudah diketahui dan dapat segera digunakan (Ekaputra, 2016) Contoh *routing protocol* proaktif adalah *Destination Sequenced Distance Vector* (DSDV), *Clusterhead Gateway Switch Routing* (CGSR), *Wireless Routing Protocol* (WRP) (Jiatmiko, 2015). Beberapa penelitian yang penulis rujuk untuk penelitian ini, telah menguji jarak yang dapat dicapai MANET yaitu antara 100-250 meter pada penelitian (Sasson, Cavin, & Schiper, 2003), dan jarak antara 100-600 meter pada penelitian (Asrullah, 2015).



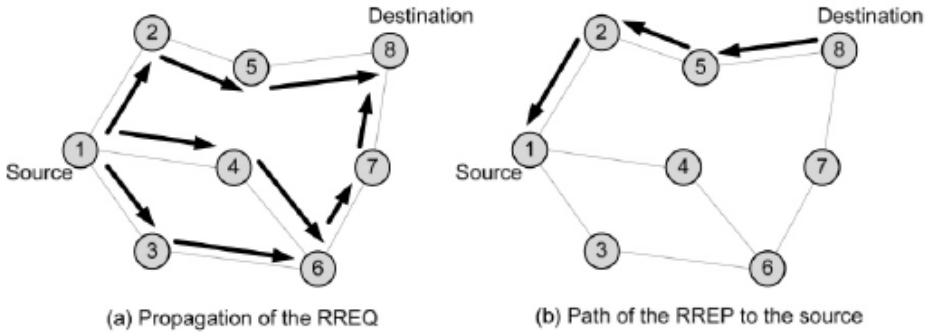
Gambar 2.3 *Routing Protocol* MANET

2.3 *Adhoc On-demand Distance Vector* (AODV)

Adhoc On-Demand Distance Vector(AODV) adalah salah satu *routing protocol* yang bersifat reaktif. AODV akan menyimpan *path* dan rute node pada tiap-tiap nodenya. Ketika terdapat node yang ingin mengirim paket namun tidak ada rutenya pada *routing table*, maka node tersebut akan melakukan *route discovery.* Node asal akan melakukan *broadcast* paket *route request* (RREQ) yang berisi alamat node tujuan ke seluruh node yang ada disekitarnya. Selain berisi alamat node tujuan, paket ini juga berisi alamat node asal, nomer urut rute,dan urutan paling akhir nomor node tujuan berada. Ketika node tujuan menerima paket RREQ dari node asal, node tujuan akan membalasnya dengan mengirimkan paket *route replay* (RREP). Node tujuan mengirimkan paket RREP dengan mengikuti kebalikan dari *path* yang dihasilkan oleh paket RREQ sebelumnya. Setelah node asal menerima paket RREP yang dikirimkan oleh node tujuan, kedua node menambahkan rute di *routing table* dengan RREP ke node tujuannya.

Kemudian node asal mengirimkan paket ke node tujuan dengan memilih rute yang melewati node paling sedikit (Pratama, Azaim, & Fauzi, 2015). Jika proses diatas gagal, maka node asal akan melakukan *reboardcast* RREQ, node akan melacak sumber RREQ, alamat IP dan *broadcast ID*. Jika node tujuan telah menerima RREQ yang telah diproses, maka node tujuan akan membuang paket RREQ dan tidak akan meneruskannya. Ketika RREP telah sampai kembali ke node asal, node akan mengatur pointer maju ke tujuan. Jika node sumber menerima RREP yang berisi nomor urut paling besar atau berisi nomor urut yang sama atau lebih kecil dengan hopcount, maka node akan melakukan pembaruan *routing table* dan menggunakan rute yang lebih baik ini. Selama rute tersebut tetap aktif, maka rute tersebut akan terus digunakan. Tabel routing yang ada di AODV berisi:

1. *Destination IP Address*: Berisi alamat IP node tujuan yang digunakan untuk menentukan rute.
2. *Next Hop*: Berupa node-node perantara yang berfungsi untuk meneruskan paket ke node tujuan, atau bisa juga berupa node tujuan.
3. *Hop Count*: Berisi jumlah node yang akan dilalui node sumber untuk dapat mencapai node tujuan.
4. *Lifetime*: Berisi catatan waktu yang digunakan untuk node menerima RREP dari node tujuan.
5. *Routing Flags*: Berisi status sebuah rute, apakah rute dalam keadaan aktif (up) atau tidak aktif (down).

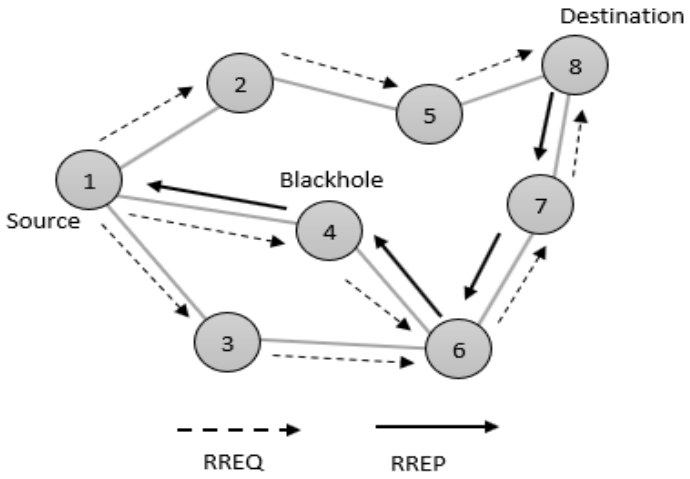


Gambar 2.4 *Adhoc On-Demand Distance Vector* (AODV)

2.4 Black Hole

Black hole adalah suatu node yang menyerang node-node lain dalam suatu jaringan. Node black hole bergerak secara acak dan akan menggangu proses pengiriman paket pada node lainnya. Node black hole bertindak menyerupai node-node normal lainnya, sehingga node lainnya menganggap node black hole aman dan normal. Padahal node ini akan mengganggu

proses pengiriman paket yang ada di jaringan tersebut. Ketika node asal mengirimkan RREQ untuk mencari rute ke node tujuan, Node black hole akan meng*advertise* bahwa dirinya adalah node yang memiliki jalur tercepat untuk menuju node tujuan, sehingga node asal akan menggunakan jalur yang diberikan oleh node black hole dan mengabaikan masukan rute dari node lainnya. setelah node asal mulai mengirimkan paket melalui node black hole, secara diam-diam node black hole akan menghapus paket yang diterimanya (Adiwicaksono, 2017).



Gambar 2.5 Black Hole

2.5 Simulasi Jaringan

Dalam komunikasi dan penelitian jaringan komputer, simulasi jaringan adalah sebuah teknik dimana sebuah program mampu menirukan perilaku jaringan sebenarnya dengan menghitung interaksi antar entitas jaringan yang berbeda dengan menggunakan rumusan tertentu atau memutar kembali pengamatan dari produksi jaringan. Kelebihan dari melakukan simulasi jaringan adalah simulasi yang dibuat tidak akan menimbulkan permasalahan pada jaringan sebenarnya, sehingga perilaku jaringan, aplikasi dan layanan pendukung dapat diamati secara bebas dan berbagai atribut juga dapat dimodifikasi secara terkontrol untuk menilai perilaku jaringan dengan kondisi atribut yang berbeda. Beberapa komponen-komponen dasar dalam simulasi yaitu:

a. Entitas

   Entitas adalah objek yang saling berkomunikasi satu sama lain dalam simulasi. Dalam konteks simulasi jaringan komputer, entitas dapat berupa perangkat komputer, paket data, atau objek non fisik seperti waktu simulasi. Setiap entitas memiliki atribut yang unik agar dapat dibedakan masing-masing fungsinya. Misalnya entitas paket akan memiliki atribut header, ukuran paket, nomor urut dan prioritas yang berbeda-beda.

b. Scheduler

Scheduler berfungsi untuk mencatatkan seluruh kegiatan dan waktu selama eksekusi berlangsung. Saat simulasi berlangsung, scheduler menjalankan waktu simulasi, mengatur kegiatan simulasi dan waktu eksekusi yang telah dibuat oleh user.

c. Resource

Resource atau sumber daya adalah bagian dari sistem yang kompleks. Sumber daya yang terbatas dalam jaringan komputer harus mampu dibagi diatara masing-masing entitas dalam simulasi.

d. Event dan Aktifitas

Dalam pelaksanaan simulasi akan terjadi beberapa peristiwa dan perubahan yang melibatkan entitas dalam simualasi. Misalnya saat terjadi peristiwa *delay*, entitas yang terlibat harus menunggu media transmisi yang sibuk untuk mengirimkan paket sampai media yang akan digunakan tidak sibuk. Paket tesebut dapat dikatakan terlibat pada aktifitas menunggu.

e. Global Varieties

Variabel global diguakan oleh beberapa fungsi atau etitas sebagai batasan nilai dalam simulasi yang sedang dijalankan. Dalam simulasi jaringan komputer variabel tersebut dapat berupa panjang antrian paket atau jumlah paket yang akan ditransmisikan.

f. Statistics Gatherer

Tugas dari pengumpul statistik adalah mengumpulkan data yang dihasilkan selama simulasi berlangsung, sehingga kesimpulan bisa didapatkan berdasarkan data yang dihasilkan oleh simulasi yang dijalankan (Ekaputra, 2016).

2.6 Network Simulator 3

Network Simulator 3 (NS-3) adalah simulator jaringan yang dibuat untuk penelitian dan pendidikan. NS3 adalah *freeware* yang berlisensi GNU GPLv2. Tujuan dari pembuatan NS-3 adalah untuk pengembangan simulasi di bidang jaringan, disesuaikan dengan kebutuhan simulasi riset jaringan modern. NS-3 berkomitmen untuk membangun simulasi yang terdokumentasi dengan baik mudah digunakan dan sesuai dengan kebutuhan alur kerja simulasi, mulai dari konfigurasi sampai dengan pengumpulan hasil dan analis. Network Simulator 3 mengembangkan model simulasi yang cukup realistis untuk memungkinkan NS-3 menjadi simulator jaringan secara *realtime*, yang dapat disesuaikan dengan keadaan *real*nya, sehingga dapat di implementasikan di keadaan nyatanya (Nsnam, 2011). Network Simulator 3

juga memiliki *tool* yang dapat digunakan *user* untuk memvisualisasikan simulasi yang dibuatnya. NetAnim adalah tool yang dibuat oleh Network Simulator 3 untuk memvisualisasikan simulasi yang dibuat dengan Network Simulator 3. Sama seperti Network Simulator 3, NetAnim dijalankan di sistem operasi Ubuntu. Beberapa istilah yang terdapat di jaringan, namun memiliki istilah spesifik pada Network Simulator 3 adalah:

a. Node

   Dalam istilah jaringan, perangkat komputer biasa disebut dengan *host* atau *end-user*. Tetapi dalam Network Simulator 3 perangkat komputer disebut dengan istilah node. Kelas NodeContainer dalam Network Simulator 3 berfungsi untuk merepresentasikan perangkat komputer saat simulasi.

b. Application

   Dalam Network Simulator 3 abstraksi dasar untuk program pengguna yang menghasilkan beberapa prosedur yang akan disimulasikan adalah aplikasi. Abstraksi ini dalam Network Simulator 3 adalah kelas AplicationContainer. Kelas ApplicationContainer menyediakan metode untuk mengatur representasi dengan versi Network Simulator 3 pada aplikasi-aplikasi level user dalam simulasi.

c. Channel

   Media yang dilalui paket data untuk saling bertransmisi dalam jaringan disebut dengan channel. Di Network Simulator 3 media yang digunakan node untuk saling mengirimkan paket dan saling berkomunikasi tersebut diistilahkan dengan kelas YansWifiChannelHelper untuk channel dalam bentuk wifi.

d. Net Device

   Dalam jaringan komputer, sebuah perangkat komputer harus memiliki perangkat Network Interface Card (NIC) agar komputer tersebut dapat saling berkomunikasi dengan perangkat komputer lainnya. Perangkat NIC tersebut juga membutuhkan sebuah software driver agar perangkat NIC dapat berfungsi dengan baik. Di Network Simulator 3 agar node-node dapat saling berkomunikasi, maka diperlukan untuk menginstall Net Device. Network Simulator 3 merepresentasikan Net Device dengan kelas NetDeviceContainer.

e. Topology Helpers

   Network Simulator 3 menyediakan fitur Topology Helpers ini untuk memudahkan user mengatur simulasi jaringan seperti node, NetDevice dan Channel. Fitur ini sangat membantu user saat membuat simulasi jaringan dalam skala besar.

2.7 Parameter Kinerja Jaringan

Suatu jaringan dapat dinilai baik dan buruk kualitasnya dari pengukuran kinerja jaringan atau biasa disebut dengan *Quality of Service* (QoS). QoS dapat didefinisikan sebagai suatu pengukuran tentang seberapa baik kualitas suatu jaringan. Kualitas baik buruknya QoS bergantung pada penilaian dari parameter-parameter berikut ini:

a. *Throughput*

Throughput adalah ukuran laju data aktual per satuan waktu, atau dapat disebut juga dengan ukuran *bandwidth* yang sebenarnya. Perbedaan dengan *bandwidth* adalah, ukuran *bandwidth* bersifat statis, sedangkan ukuran *throughput* bersifat dinamis bergantung dari trafik yang terjadi (Jiatmiko, 2015). *Bandwidth* adalah kemampuan sebuah jaringan untuk mentransmisikan data. *Bandwidth* biasanya diukur dalam satuan bit per detik (bps). *Bandwidth* dianggap tidak cukup untuk menyimpulkan kecepatan suatu jaringan. Oleh karena itu dibutuhkan pengukuran *throughput* untuk dapat meyimpulkan kecepatan jaringan, karena pengukuran *throughput* berdasarkan total kedatangan paket yang diterima destinasi dibagi dengan waktu yang dibutuhkan untuk mengirimkan paket tersebut sampai ke destinasi. Sehingga dapat dihasilkan kecepatan (*rate*) transmisi yang efektif. Rumus untuk menghitung *throughput* adalah:

$$Throughput = \frac{\text{Ukuran paket diterima}}{\text{Waktu pengiriman paket}}$$

b. Delay

Delay adalah waktu tunda suatu paket yang disebabkan proses transmisi dari node asal ke node tujuannya. Dalam suatu jaringan *delay* dapat menjadi acuan penilaian kualitas jaringan. Semakin kecil nilai delay yang dihasilkan, maka semakin baik jaringan tersebut (Jiatmiko, 2015). *Delay* dapat dipengaruhi oleh beberapa hal, diantaranya jarak antar node asal ke node tujuan, media transmisi, atau bisa juga karena waktu proses yang lama. Rumus untuk menghitung *delay* adalah:

$$Delay = \text{waktu paket diterima} - \text{waktu paket dikirim}$$

Karena *delay* dapat dijadikan acuan penilaian kualitas jaringan, maka diusahakan agar *delay* yang dihasilkan harus seminim mungkin. Ada 4 kategori jaringan berdasarkan nilai *delay* yang terjadi, sebagaimana pada Tabel 2.1

Tabel 2.1 Kategori Jaringan Berdasarkan Nilai *Delay*

Kategori Jaringan	Nilai *Delay*
Sangat Baik	0 ms

Kategori Jaringan	Nilai *Delay*
Baik	75 ms
Buruk	125 ms
Sangat Buruk	225 ms

Sumber: (Ekaputra, 2016)

c. Packet Loss

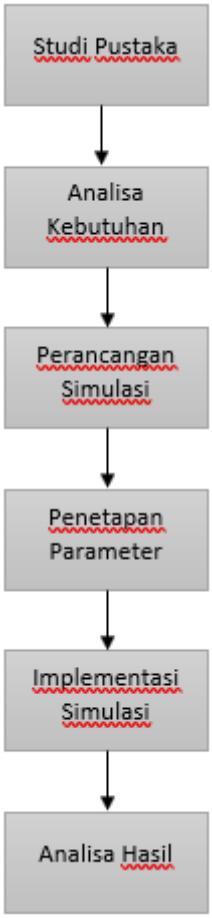
Packet Loss adalah paket yang hilang selama proses transmisi paket dari node asal ke node tujuan. *Packet loss* dapat terjadi karena beberapa hal diantarnya: antrian yang melebihi kapasitas jaringan, node yang bekerja melebihi kapasitas bufer atau memori node yang terbatas, kontrol jaringan yang mengatur jumlah trafik yang mengalir harus sesuai dengan jumlah besaran *bandwidth*, sehingga jika ada besaran trafik yang melebihi kapasitas bandwidth, maka *policing control* akan membuang kelebihan trafik tersebut dan adanya serangan dalam jaringan (Ekaputra, 2016). Rumus untuk menghitung *packet loss* adalah:

$$Packet\ Loss = \frac{\text{Paket yang dikirim} - \text{Paket yang diterima}}{\text{Paket yang dikirim}} 100\%$$

BAB III
METODOLOGI PENELITIAN

3.1 Langkah Penelitian

Dalam menyelesaikan penelitian ini, penulis menyusun beberapa tahapan atau metode agar penelitian yang dilakukan bisa terarah dan dapat menghasilkan hasil akhir yang tepat. Beberapa tahapan yang dilakukan penulis adalah seperti yang terdapat pada diagram alur Gambar 3,1.



Gambar 3.1 Diagram Alur Langkah Penelitian

3.2 Studi Pustaka

Beberapa referensi yang penulis dapatkan sebagai acuan untuk menyelesaikan penelitian ini berasal dari beberapa sumber. Penulis mengambil referensi dari jurnal, skripsi, tutorial di internet dan penelitian sebelumnya. Pada penelitian yang dilakukan oleh Istas Pratomo dan M Hizrian Hizburrahman, mereka membahas tentang bagaimana mekanisme keamanan untuk

mendeteksi keberadaan serangan black hole dan grey hole di *routing protocol* AODV pada jaringan MANET. dalam penelitian tersebut dijelaskan bagaimana cara menemukan keberadaan node berbahaya dan mengisolir node tersebut dari jaringan dengan cara memanfaatkan informasi yang dibawa oleh paket yang berisi informasi mengenai keberadaan node berbahaya yang telah terdeteksi. Pada penelitian tersebut juga dibahas mengenai pengujian beberapa pengaruh parameter terhadap *Throughput*, pengaruh parameter terhadap *delay* dan pengaruh parameter terhadap daya yang digunakan dengan metode *flow control* yang berbeda-beda. Kesimpulan yang bisa didapatkan dari pengujian diatas adalah untuk pengaruh parameter terhadap *throughput*, semakin besar ukuran jaringan, maka nilai *throughput* akan semakin menurun, hal ini disebabkan karena ukuran paket, ukuran *buffer* dan *packet injection rate* yang tetap apabila ukuran jaringan diperbesar, maka waktu yang diperlukan paket untuk mencapai tujuan akan bertambah sehingga menyebabkan nilai *throughput* menurun. Sedangkan untuk pengaruh parameter terhadap delay, *delay* berbanding lurus dengan ukuran jaringan, nilai delay akan meningkat jika ukuran jaringan ditingkatkan. Untuk dapat menurunkan *delay* maka *paket injection rate*, ukuran paket dan ukuran jaringan harus diperkecil dengan ukuran *buffer* yang diperbesar. Sedangkan untuk pengaruh parameter terhadap daya yang digunkan dapat disimpulkan bahwa peningkatan ukuran jaringan akan membawa dampak meningkatnya penggunaan daya pada jaringan. Maka untuk mengurangi penggunaan daya, ukuran paket, ukuran *buffer*, dan ukuran jaringan juga harus diperkecil (Pratomo & Hizburrahman, 2015).

Referensi lainnya yang membahas tentang serangan black hole pada jaringan MANET adalah penelitian yang dilakukan oleh Neelam Janak Kumar Patel dan Dr. Khushboo Tripathi. Jurnal internasional penelitian ini terbit pada mei 2017. Penelitian ini menguji jaringan MANET dengan menggunakan 25 node dngan skenario 0 node black hole, 1 node black hole, 3 node black hole dan 5 node black hole dan menggunakan *routing protocol* AODV. Pembahasan hasil pada penelitian ini juga berdasarkan dari pengukuran nilai *throughput, delay* dan *packet loss* yang dihasilkan selama simulasi berlangsung. Dalam hasil analisa penelitian ini dijelaskan bahwa nilai *throughput* dan *delay* semakin mengecil seiring dengan bertambahnya jumlah node black hole. Hal ini disebabkan karena semakin bertamah banyak node black hole, maka semakin sedikit atau bahkan tidak ada paket yang berjalan di jaringan tersebut sehingga tidak ada nilai *throughput* dan *delay* yang tercatat dalam simulasi. Hal ini berbanding terbalik dengan nilai *packet loss* yang dihasilkan, semakin banyak node black hole,

maka nilai *packet loss* akan semakin tinggi dibandingkan dengan keadaan jaringan normal (Patel & Tripathi, 2017).

3.3 Analisa Kebutuhan

Analisa kebutuhan dilakukan penulis untuk mengetahui kebutuhan-kebutuhan apa saja yang diperlukan penulis, agar penelitian yang dijalankan penulis dapat berjalan dengan lancar dan tidak ada kekurangan yang dapat menghambat penelitian. Dalam hal ini, penulis mengelompokkan kebutuhan-kebutuhan dalam dua jenis, yaitu:

3.3.1 Kebutuhan Perangkat Keras

Untuk melakukan penelitian simulasi black hole dalam jaringan MANET ini, penulis membutuhkan perangkat komputer untuk menjalankan simulasinya. Pada penelitian ini, penulis menggunakan laptop merk Lenovo model ideapad Z480 dengan spesifikasi *prosessor* Intel core i5-3210M CPU @2.50 GHz (4 CPUs) dan memori RAM sebesar 4 MB. Dengan spesifikasi tersebut, perangkat yang digunakan penulis sudah cukup mumpuni untuk melakukan penelitian simulasi black hole dalam jaringan MANET ini.

3.3.2 Kebutuhan Perangkat Lunak

Selain kebutuhan perangkat keras, penulis juga membutuhkan beberapa perangkat lunak untuk menyelesaikan penlitian ini. diantaranya:

a. Sistem Operasi Ubuntu 16.04 LTS

   Penulis menggunakan dua sistem operasi pada perangkat yang digunakan untuk melakukan penelitian ini, yaitu sistem operasi Ubuntu dan sistem operasi Windows. Sistem operasi Ubuntu adalah sistem operasi distribusi linux yang mempunyai basis Debian. Sistem operasi ini penulis gunakan karena merupakan *requirement* dari network simulator 3 karena simulator ini belum dapat berjalan di sistem operasi windows.

b. Sistem Operasi Windows 10

   Sistem operasi ini penulis gunakan untuk menulis laporan penelitian ini dan untuk memanajemen referensi yang penulis jadikan acuan menggunakan aplikasi mendeley.

c. Network Simulator 3

   Network Simulator 3 adalah simulator yag dibuat untuk penelitian dan pendidikan yang menyangkut tentang jaringan dan keamanan komuputer. Network Simulator 3 bersifat *freeware*. Network Simulator 3 penulis pilih karena dianggap mampu untuk mensimulasikan jaringan seperti keadaan aslinya dan dapat diatur secara *realtime*. Network Simulator 3 juga mampu mensimulasikan beberapa protokol-protokol jaringan

mirip dengan aslinya dengan baik. Langkah – langkah untuk menginstall Network Simulator 3 adalah sebagai berikut:

1. Unduh file master Network Simulator 3 di situs resmi Network Simulator 3 https://www.nsnam.org
2. Install beberapa *library-libarary* yang mendukung network Simulator 3 dengan mengetikkan perintah berikut ini di konsol terminal Ubuntu.

```
sudo apt-get install gcc g++ python python-dev mercurial bzr gdb valgrind gsl-bin libgsl0-dev libgsl0ldbl flex bison tcpdump sqlite sqlite3 libsqlite3-dev libxml2 libxml2-dev libgtk2.0-0 libgtk2.0-dev uncrustify doxygen graphviz imagemagick texlive texlive-latex-extra texlive-generic-extra texlive-generic-recommended texinfo dia texlive texlive-latex-extra texlive-extra-utils texlive-generic-recommended texi2html python-pygraphviz python-kiwi python-pygoocanvas libgoocanvas-dev python-pygccxml
```

3. Buat direktori baru dan ekstrak file master pada direktori tersebut.
4. Dengan menggunakan terminal konsol, masuk ke direktori file master NS 3 berada.
5. Ketikkan perintah berikut ini untuk menginstall Network Simuator 3.

```
./build.py --enable-examples --enable-tests
```

6. Selanjutnya ketikkan perintah berikut untuk mengkonfigurasi sistem dengan waf (*build tool*)

```
./waf -d debug --enable-examples --enable-tests configure
```

d. NetAnim

NetAnim adalah aplikasi yang merupakan satu paket dengan Network Simulator 3. NetAnim berfungsi untuk memvisualisasikan model yang jaringan yang dibuat oleh penulis di Network Simulator 3 dalam bentuk animasi. Langkah-langkah mengistall NetAnim adalah sebagai berikut:

1. Install beberapa *library-libarary* yang dibutuhkan dengan mengetikkan perintah berikut ini

```
sudo apt-get download qt4-qmake libqt4-dev libxml2-dev
```

2. Masuk ke direktori NetAnim yang ada di folder NS3 kemudian ketikkan perintah berikut untuk menginstall NetAnim.

```
make clean
qmake NetAnim.pro
make
```

3. Jalankan NetAnim dengan mengetikkan perintah

```
./NetAnim
```

e. Sublime Text 3

Sublime Text 3 adalah sebuah aplikasi text editor untuk menulis source code dalam berbagai bahasa pemrograman. Diantaranya bahasa pemrograman PHP, Java, C dan C++

yang digunakan penulis untuk membuat simulasi jaringan di Network Simulator 3 ini. Sublime text 3 dirilis pada tanggal 29 Januari 2013 dan dapat berjalan pada beberapa sistem operasi seperti sistem operasi Ubuntu dan sistem operasi Windows. Penulis memilih menggunakan Sublime Text 3 karena selain ringan, aplikasi text editor ini memiliki beberapa fitur yang memudahkan penulis saat menulis *source code,* seperti fitur *auto correct* dan *auto complete.*

f. Microsoft Office Word

Penulis menggunakan Microsoft Office Word untuk menulis laporan penelitian karena selain sudah terbiasa menggunakan aplikasi pengolah kata ini, pihak jurusan Teknik Informatika juga sudah menyiapkan template laporan dalam bentuk *file* word.

g. Mendeley Desktop

Mendeley penulis gunakan untuk memanajemen berbagai referensi yang penulis gunakan untuk melakukan penelitian ini. Mendeley desktop sangat bermanfaat bagi penulis ada saat melakukan penulisan laporan karena saat mencantumkan daftar referensi yang penulis gunakan, dapat otomatis terhubung dengan Microsoft office word.

h. Opera Browser

Selain referensi dalam bentuk pdf, penulis juga mencari referensi dalam bentuk tutorial video di youtube dan referensi dalam bentuk *library document* di situs resmi dari network Simulator 3. Untuk mengakses referensi tersebut maka penulis menggunakan browser Opera, karena menurut penulis browser ini lebih ringan dibandingkan browser Chrome dan Mozilla Firefox.

3.4 Perancangan Simulasi

Setelah dilakukannya analisis kebutuhan, maka diketahui apa saja yang dibutuhkan untuk menyelesaikan penelitian ini. Tahapan selanjutnya adalah tahap merancang simulasi yang akan dibuat. Ini diperlukan agar simulasi yang dibuat dapat memberikan hasil seperti yang ingin dicapai. Pada penelitian ini ada beberapa node dengan masing-masing peran sebagai berikut:

a. Node Asal: Node tempat pertama yang akan mengirimkan paket
b. Node Hop: Node yang dilalui paket yang dikirimkan node sumber sebelum paket sampai ke node tujuan.
c. Node Tujuan: Node yang menjadi tujuan akhir paket yang dikirimkan dari node asal.
d. Node Black Hole: Node yang akan menghapus paket yang dikirimkan node asal, dan mengakui dirinya memiliki jalur terpendek menuju node tujuan.

Pada simulasi ini juga akan dibuat beberapa skenario agar dapat diketahui perbedaan hasil dari *throughut, delay* dan *packet loss*, dengan jumlah node dan black hole yang berbeda-beda. Skenario tersebut adalah :

a. Skenario 4 Node, 1 Black hole
b. Skenario 4 Node, 2 Black Hole
c. Skenario 20 Node, 1 Black Hole
d. Skenario 20 Node, 2 Black Hole
e. Skenario 50 Node, 1 Black Hole
f. Skenario 50 Node, 2 Black Hole

3.5 Penetapan Parameter

Parameter simulasi adalah beberapa nilai yang digunakan sebagai acuan dalam proses simulasi. Parameter yang berbeda pada tiap skenario akan mengakibatkan perbedaan hasil pada nilai *throughput, delay* dan *packet loss*. Maka dari itu parameter yang ditentukan oleh penulis bersifat tetap selama proses simulasi berlangsung. Parameter-parameter yang penulis tetapkan selama simulasi berlangsung seperti pada Tabel 3.1.

Tabel 3.1 Parameter Simulasi

Parameter	Nilai
Routing Protocol	AODV
Waktu Simulasi	100 detik
Tx Power	50
Tipe Pergerakan	Constan Position Mobility Model
Tipe Koneksi	UDP
Tipe Wifi Mac	Standard 802.11b
Ukuran Paket	64 bytes
Tipe Kanal	Wireless

a. Routing Protocol

Routing protocol adalah aturan yang digunakan node utnuk saling bertukar informasi sehingga akan menghasilkan *routing table* yang akan membuat pengalamatan lebih jelas dan mempermudah node mengirimkan paket ke alamat yang dituju.

b. Waktu Simulasi

Waktu simulasi mengatur lama waktu simulasi akan berlangsung. Penulis menetapkan waktu 100 detik karena dalam waktu tersebut dianggap sudah cukup untuk mensimulasikan serangan black hole dalam jaringan MANET.

c. Tx Power

Tx Power adalah daya yang akan digunakan wifi untuk mengirimkan paket data.

d. Tipe Pergerakan

Tipe pergerakan digunakan untuk mengatur pergerakan node selama proses simulasi berlangsung. Dalam simulasi ini dipilih tipe pergerakan *Constan Position Mobility Model* yang berarti node akan berada dalam keadaan diam saat simulasi berlangsung.

e. Tipe Koneksi

Tipe Koneksi yang digunakan untuk mentransmisikan paket data pada simulasi ini adalah tipe koneksi UDP.

f. Tipe Wifi Mac

Tipe wifi yang digunakan dalam simulasi ini adalah tipe standar 802.11b

g. Ukuran Paket

Ukuran paket adalah besaran paket yang akan dikirimkan dari node asal ke node tujuan. Ukuran paket pada simulasi ini diatur *default* oleh Network Simulator 3.

h. Tipe Kanal

Tipe kanal dipilh wireless karena penelitian ini bertujuan untuk meneliti simulasi jaringan MANET yang identik dengan jaringan nirkabel.

BAB IV
HASIL DAN PEMBAHASAN

4.1 Implementasi Simulasi

Setelah menyelesaikan tahap perancangan dan menentukan parameter, tahap selanjutnya adalah implementasi dari semua yang telah dirancang pada tahap sebelumnya. Tahap pertama dari implementasi adalah penulisan *source code* program. Network Simulator 3 telah menyediakan beberapa modul yang dapat memudahkan penggunanya untuk membuat simulasi menggunakan Network Simulator 3. Pada penelitian ini, penulis menyusun *source code* dengan memanfaatkan beberapa contoh penelitian tentang simulasi black hole dalam jaringan MANET yang penulis dapatkan dari internet. Selanjutnya penulis menyesuaikan *source code* dengan kebutuhan penulis berdasarkan beberapa skenario yang penulis buat sebelumnya.

4.1.1 *Source Code* dan Penjelasan Program

a. Memanggil Library NS-3

Network Simulator 3 telah menyediakan beberapa modul-modul yang dapat digunakan penggunanya agar lebih mudah dan lebih cepat saat proses pemrograman simulasi. Modul yang disediakan tidak hanya dalam bentuk *source code*, tetapi bisa juga dalam bentuk file hasil kompilasi. Sehingga fungsi yang ada dalam kode file tersebut dapat dihubungkan (*include*) dengan kode program yang dibuat oleh pengguna Network Simulator 3. Pada program simulasi peelitian ini, penulis memasukkan modul-modul seperti dibawah ini.

```
1  #include "ns3/aodv-module.h"
2  #include "ns3/core-module.h"
3  #include "ns3/network-module.h"
4  #include "ns3/internet-module.h"
5  #include "ns3/applications-module.h"
6  #include "ns3/mobility-module.h"
7  #include "ns3/wifi-module.h"
8  #include "ns3/flow-monitor-module.h"
9  #include "ns3/mobility-module.h"
10 #include "myapp.h"
11 #include "ns3/netanim-module.h"
12 #include <fstream>
13 #include <iostream>
```

b. Fungsi Utama

Fungsi utama berisi berbagai perintah untuk membuat node, menentukan node black hole, pembuatan *file* csv, menentukan model pergerakan node, menghitung nilai *throughput*, *delay*, *packet loss* dan menjalankan fungsi simulasi secara keseluruhan. Fungsi utama diawali dengan *script*:

```
1  int main (int argc, char *argv[])
2  {
```

Setelah itu diikuti dengan *script-script* dibawah ini:

c. Fungsi Command Setup

Fungsi Command Setup berfungsi untuk menyimpan nilai parameter yang dikirimkan ketika penulis menjalankan perintah ./waf melalui terminal. *Script* yang terdapat warna merah pada nomor baris merupakan *script* tambahan yang penulis tulis untuk menambahkan parameter yang dibutuhkan penulis ketika menjalakan simulasi.

```
1  CommandLine cmd;
2  cmd.AddValue ("EnableMonitor", "Enable Flow Monitor",
3  enableFlowMonitor);
4  cmd.AddValue ("phyMode", "Wifi Phy mode", phyMode);
5  cmd.AddValue ("CSVfileName", "The name of the CSV output file
6  name", m_CSVfileName);
7  cmd.Parse (argc, argv);
```

d. *Script* Untuk Membuat Node

Script berikut merupakan *script* yang digunakan untuk membuat node dan menentukan node yang bertindak sebagai node biasa dan yang bertindak sebagai node black hole. *Script* yang terdapat warna merah pada nomor baris merupakan *script* hasil modifikasi penulis agar mempermudah saat mengubah jumlah node yang dibutuhkan dalam penelitian ini.

```
1  NS_LOG_INFO ("Create nodes.");
2  NodeContainer c; // ALL Nodes
3  NodeContainer not_malicious;
4  NodeContainer malicious;
5  c.Create(titik);
6
7  if(attackenable==false){
8    for(int i=0;i<titik;i++){
9      not_malicious.Add(c.Get(i));
10   }
11 }
12 else{
13   for(int i=0;i<titik;i++){
14     if(i == 2){
15       malicious.Add(c.Get(i));
16     }
17     else{
18       not_malicious.Add(c.Get(i));
19     }
20   }
21 }
```

e. Inisialisasi dan Pengaturan Wifi

Script ini digunakan untuk mengatur wifi dan jenis wifi yang digunakan sebagai media pengiriman dan penerimaan paket data. Kemudian program akan membuat *channel* yang digunakan sebagai penghubung antar node.

```
// Set up WiFi
WifiHelper wifi;

YansWifiPhyHelper wifiPhy =  YansWifiPhyHelper::Default ();
wifiPhy.SetPcapDataLinkType (YansWifiPhyHelper::DLT_IEEE802_11);

YansWifiChannelHelper wifiChannel ;
wifiChannel.SetPropagationDelay("ns3::ConstantSpeedPropagationDela
yModel");
wifiChannel.AddPropagationLoss("ns3::TwoRayGroundPropagationLossMo
del",
                                "SystemLoss", DoubleValue(1),
                                "HeightAboveZ", DoubleValue(1.5));

wifiPhy.SetChannel (wifiChannel.Create ());

// Add a non-QoS upper mac
NqosWifiMacHelper wifiMac = NqosWifiMacHelper::Default ();
wifiMac.SetType ("ns3::AdhocWifiMac");

// Set 802.11b standard
wifi.SetStandard (WIFI_PHY_STANDARD_80211b);

wifi.SetRemoteStationManager ("ns3::ConstantRateWifiManager",
                              "DataMode",StringValue(phyMode),
                              "ControlMode",StringValue(phyMode));
```

f. Membuat *Device*

Script ini digunakan untuk membuat *device* pada setiap node yang berfungsi untuk menghubungkan antara node dan *channel* yang dibuat sebelumnya

```
NetDeviceContainer devices;
devices = wifi.Install (wifiPhy, wifiMac, c);
```

g. Menentukan Routing Protocol

Selanjutnya tentukan *routing protocol* yang akan digunakan dalam simulasi ini. Pada penelitian ini penulis menggunakan *routing protocol* AODV, sehingga *script* yang dituliskan adalah seperti berikut:

```
//  Enable AODV
AodvHelper aodv;
AodvHelper malicious_aodv;

// Set up internet stack
InternetStackHelper internet;
internet.SetRoutingHelper (aodv);
internet.Install (not_malicious);

malicious_aodv.Set("IsMalicious",BooleanValue(true));
internet.SetRoutingHelper (malicious_aodv);
internet.Install (malicious);
```

h. Mengatur IP Address pada Device

Script ini digunakan untuk mengatur IP Address pada tiap-tiap node. Selain itu bisa juga digunakan untuk mengatur *range IP Address* yang akan digunakan tiap-tiap node. Tidak

ada alasan khusus penulis untuk mengatur *IP Address* simulasi ini dengan 10.1.2.0 dan netmask 255.255.255.0.

```
// Set up Addresses
Ipv4AddressHelper ipv4;
NS_LOG_INFO ("Assign IP Addresses.");
ipv4.SetBase ("10.1.2.0", "255.255.255.0");
Ipv4InterfaceContainer ifcont = ipv4.Assign (devices);
```

i. Membuat aplikasi dan Menentukan Node Tujuan

Script berikut ini digunakan untuk membuat aplikasi dan menginisiasi node yang akan menjadi node tujuan disini juga ditentukan node mana yang akan menjadi node asal atau node sumber dikirimkannya paket. Penulis mengatur agar node ke 0 menjadi node asalnya. *Script* yang terdapat warna merah pada nomor baris merupakan *script* modifikasi penulis.

```
NS_LOG_INFO ("Create Applications.");

for(int i = 0; i<titik; i++){
uint16_t sinkPort = titik;
Address sinkAddress (InetSocketAddress (ifcont.GetAddress (i),
sinkPort));
PacketSinkHelper packetSinkHelper ("ns3::UdpSocketFactory",
InetSocketAddress (Ipv4Address::GetAny (), sinkPort));
ApplicationContainer sinkApps = packetSinkHelper.Install (c.Get
(i));
sinkApps.Start (Seconds (0.));
sinkApps.Stop (Seconds (100.));

Ptr<Socket> ns3UdpSocket = Socket::CreateSocket (c.Get (0),
UdpSocketFactory::GetTypeId ());
```

j. Mengatur Koneksi UDP

Script ini berfungsi untuk mengatur besarnya paket yang akan dikirimkan, mengatur besaran *bandwidth* saat simulasi dan mengatur waktu mulai dan berakhirnya paket dikirimkan. *Script* yang terdapat warna merah pada nomor baris merupakan *script* modifikasi penulis.

```
// Create UDP application at n1
Ptr<MyApp> app = CreateObject<MyApp> ();
app->Setup (ns3UdpSocket, sinkAddress, 64, 0, DataRate
("2048bps"));
c.Get (i)->AddApplication (app);
app->SetStartTime (Seconds (0.));
app->SetStopTime (Seconds (100.));
}
```

k. Mengatur Mobility Node

Script ini digunakan untuk mengatur pergerakan node. Node dapat diatur pergerakannya menjadi statis atau pergerakannya menjadi dinamis. *Script* yang terdapat warna merah pada nomor baris merupakan *script* hasil modifikasi penulis agar mempermudah saat mengubah jumlah node yang dibutuhkan dalam penelitian ini.

```
// Set Mobility for all nodes
```

```
2
3   MobilityHelper mobility;
4   Ptr<ListPositionAllocator> positionAlloc = CreateObject
5       <ListPositionAllocator>();
6   for(int i=0;i<titik;i++){
7     positionAlloc ->Add(Vector(i, 0, 0));
8   }
9   mobility.SetPositionAllocator(positionAlloc);
10  mobility.SetMobilityModel("ns3::ConstantPositionMobilityModel");
11  mobility.Install(c);
```

l. Membuat *file* .xml

Selain meggunakan Network Simulator 3, penulis juga menggunakan aplikasi NetAnim untuk memvisualisasikan program simulasi yang dibuat. Untuk dapat melihat tampilan visual simulasi, maka diperlukan file berformat .xml. *Script* berikut ini yang harus di masukkan kedalam program, agar program yang dibuat dapat divisualisasikan menggunakan NetAnim.

```
1   AnimationInterface anim ("skripsiblackhole.xml");
```

m. Mengatur Tampilan Node pada NetAnim

Script berikut ini berfungsi untuk mengatur tampilan node yang akan ditampilkan di NetAnim. *Script* berikut mengatur jarak tiap-tiap node, warna dari node biasa, warna node blackhole, dan warna node sumber. *Script* dibawah ini merupakan *script* tambahan modifikasi penulis.

```
1   int kolom = 1;
2   int baris = 1;
3   for(int i=0;i<titik;i++){
4     AnimationInterface::SetConstantPosition (c.Get (i), kolom*100,
5   baris*100);
6     if((i+1)%10 == 0){
7       kolom = 0;
8       baris++;
9     }
10    kolom++;
11  }
12  if(attackenable==true){
13    for(uint32_t n=0; n<c.GetN(); n++){
14    if(n!=0 ){
15        // anim.UpdateNodeDescription (c.Get (n),"Node");
16        anim.UpdateNodeColor (c.Get (n), 255,0,0);
17      }
18      else if(n==0){
19          anim.UpdateNodeDescription (c.Get (n),"Source");
20          anim.UpdateNodeColor (c.Get (n), 0,255,0);
21        }
22      else{
23        anim.UpdateNodeDescription (c.Get (n),"Blackhole");
24        anim.UpdateNodeColor (c.Get (n), 0,0,0);
25      }
26    }
27  }
28  if(attackenable==false){
29    for(uint32_t n=0;n<c.GetN(); n++){
```

```
    anim.UpdateNodeColor(c.Get(n),255,0,0);
  }
}
```

n. Fungsi untuk Menghitung *Throughput, Delay* dan *Packet Loss*

Network Simulator 3 telah menyediakan modul untuk menghitung *throughput, delay packet loss* dan lain-lainnya. Untuk dapat menggunakan modul tersebut, maka diperlukan *script* berikut. *Script* dibawah ini merupakan *script* tambahan modifikasi penulis.

```
// Calculate Throughput using Flowmonitor

FlowMonitorHelper flowmon;
Ptr<FlowMonitor> monitor = flowmon.InstallAll();

Ptr<Ipv4FlowClassifier> classifier =
DynamicCast<Ipv4FlowClassifier> (flowmon.GetClassifier ());
std::map<FlowId, FlowMonitor::FlowStats> stats = monitor-
>GetFlowStats ();
for (std::map<FlowId, FlowMonitor::FlowStats>::const_iterator i =
stats.begin (); i != stats.end (); ++i)
  {
Ipv4FlowClassifier::FiveTuple t = classifier->FindFlow (i-
>first);
if ((t.sourceAddress=="10.1.2.1"))
 {
std::cout << "Flow " << i->first  << " (" << t.sourceAddress << "
-> " << t.destinationAddress << ")\n";
std::cout << "  Tx Bytes:   " << i->second.txBytes << "\n";
std::cout << "  Rx Bytes:   " << i->second.rxBytes << "\n";
std::cout << "  Throughput: " << i->second.rxBytes * 8.0 / (i-
>second.timeLastRxPacket.GetSeconds() - i-
>second.timeFirstTxPacket.GetSeconds())/1024/1024  << " Mbps\n";
std::cout << "  Delay:      " << i->second.delaySum << "\n"; //i-
>second.delaySum // Modifikasi
std::cout << "  Packet Loss:" << i->second.lostPackets << "\n";
// Modifikasi
```

o. Membuat *file* .csv

File .csv berfungsi untuk mencatatkan hasil keluaran dari program yang dibuat kedalam bentuk *file* csv. Sehingga hasil keluaran dapat dibuka menggunakan aplikasi pengolah angka seperti LibreOffice Calc atau Microsoft Office Excel. *Script* dibawah ini merupakan *script* tambahan yang penulis buat untuk memudahkan dokumentasi hasil simulasi.

```
std::ofstream out (m_CSVfileName.c_str (), std::ios::app);
out <<malicious.GetN()<<","<<
      titik <<","<<
      txpower << ","<<
      i->second.txBytes << ","<<
      i->second.rxBytes <<","<<
      i->second.lostPackets <<","<<
      i->second.rxBytes * 8.0 / (i-
>second.timeLastRxPacket.GetSeconds() - i-
>second.timeFirstTxPacket.GetSeconds())/1024/1024  << ","<<
      i->second.delaySum <<","<<
      t.sourceAddress <<","<<
      t.destinationAddress << ","<<
```

```
14 std::endl;
15 }
```

p. Memulai dan Menghentikan Simulasi

```
1 NS_LOG_INFO ("Run Simulation.");
2 Simulator::Stop (Seconds(100.0));
3 Simulator::Run ();
```

4.1.2 Menjalankan Simulasi dan Pengambilan Data

Sebelum menjalankan simulasi serangan black hole, Network Simulator 3 perlu ditambahkan *patch* blackhole tersendiri. Karena pada keadaan standarnya Network Simulator 3 tidak dapat mensimulasikan black hole. Untuk langkah instalasi *patch* black hole adalah sebagai berikut:

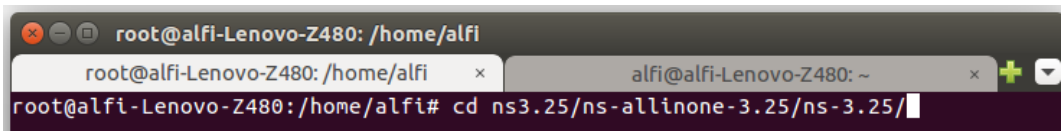
1. Unduh *patch* black hole pada tautan http://mohittahiliani.blogspot.co.id/2014/12/ns-3-blackhole-attack-simulation-in-ns-3.html sesuaikan dengan versi Network Simulator 3 yang di*install*.
2. Letakkan *file patch* black hole yang telah diunduh ke direktori Network Simulator 3.
3. Buka konsol terminal Ubuntu dan ketikkan perintah berikut

```
1 Patch -p1 < (versi patch yang diunduh)
```

4. Masuk ke direktori ns-allinone-3.25/ns-3.25 menggunakan konsol terminal kemudian ketikkan perintah berikut.

```
1 ./waf
```

Setelah *script* program selesai ditulis dan *patch* black hole terinstall, *script* tersebut diletakkan di direktori Scratch pada folder hasil instalasi Network Simulator 3 di sistem operasi Ubuntu. Untuk menjalakan simulasi, masuk ke direktori yang berisi file .waf yang berada di direktori ns-allinone-3.25/ns-3.25 sebagai *super user* seperti pada Gambar 4.1.



Gambar 4.1 Masuk ke Direktori NS-3

Setelah masuk ke direktori yang ditentukan, jalankan program simulasi dengan mengetikkan perintah seperti berikut.

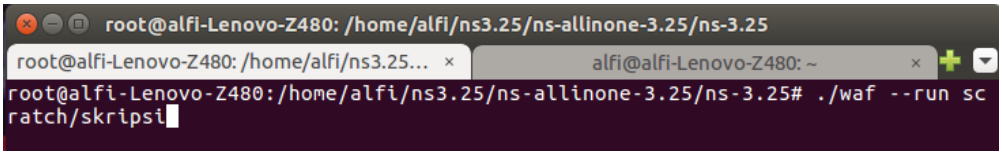
```
1 ./waf –run scratch/skripsi
```

Penjelasan:

./waf –-run : Perintah untuk menjalankan program simulasi.

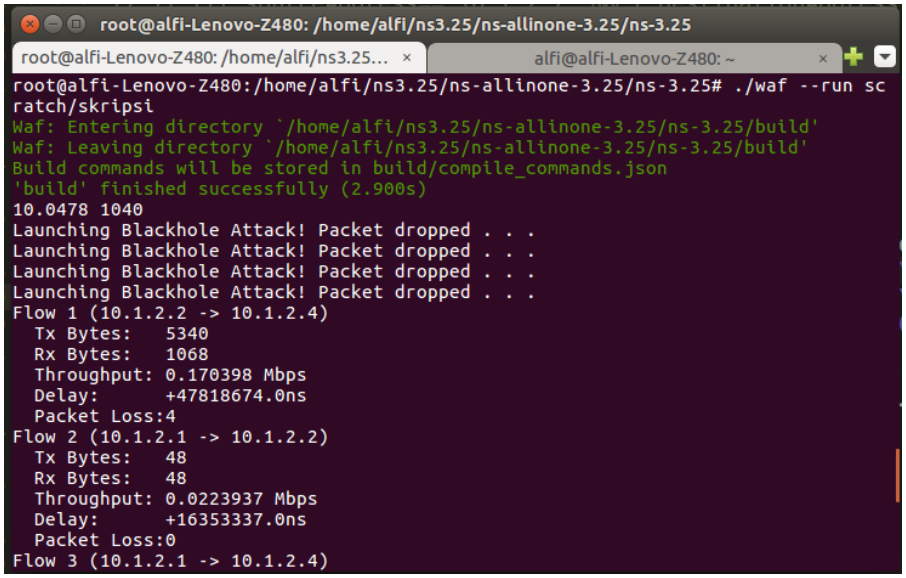
Scratch : Direktori tempat *file* program disimpan.

Skripsi : Nama *file* program simulasi.

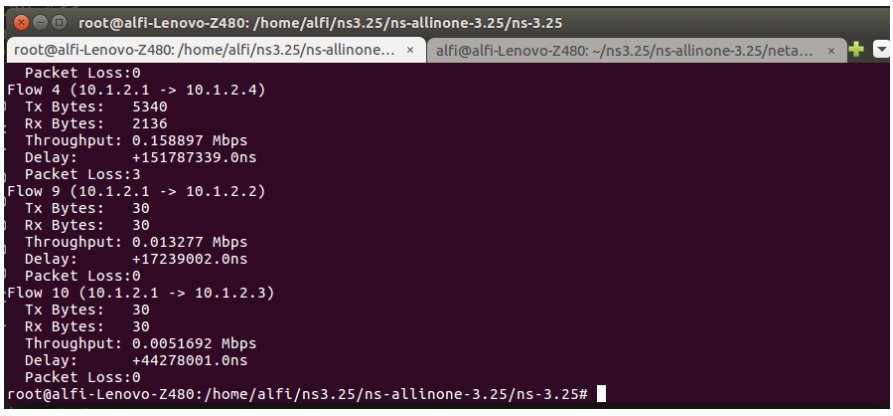


Gambar 4.2 Perintah Menjalankan Program Simulasi

Network Simulator 3 kemudian akan mem*build* program yang diperintahkan dan akan memunculkan hasil seperti Gambar 4.3 dan Gambar 4.4.

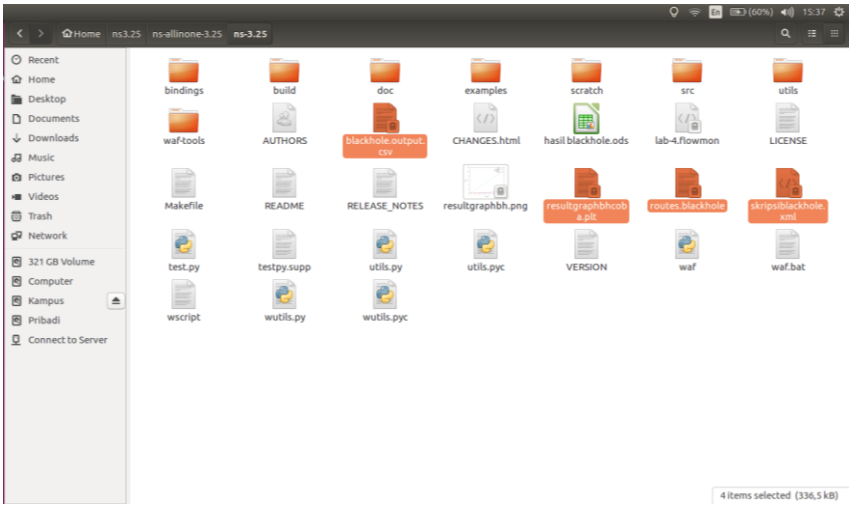


Gambar 4.3 Hasil Simulasi Berjalan



Gambar 4.4 Tampilan Selesai Simulasi

Setelah Network Simulator 3 selesai menjalankan program simulasi, akan muncul 4 buah *file* hasil keluaran simulasi yang dijalankan sebelumnya, yaitu *file* .csv, .plt, .xml dan *file* yang mencatatkan rute black hole. Contoh *file-file* hasil keluaran simulasi dapat dilihat dari Gambar 4.5.



Gambar 4.5 File-File hasil Keluaran Simulasi

File-file tersebut akan digunakan saat dilakukan proses analisis data pada langkah selanjutnya. Sedangkan untuk *file* .xml yang berfungsi untuk melihat visualisasi simulasi dalam bentuk animasi diperlukan aplikasi khusus yakni NetAnim yang sudah satu paket dengan Network Simulator 3. Contoh tampilan node yang akan ditampilkan aplikasi NetAnim akan dijelaskan penulis di bagian skenario simulasi pada bab ini.

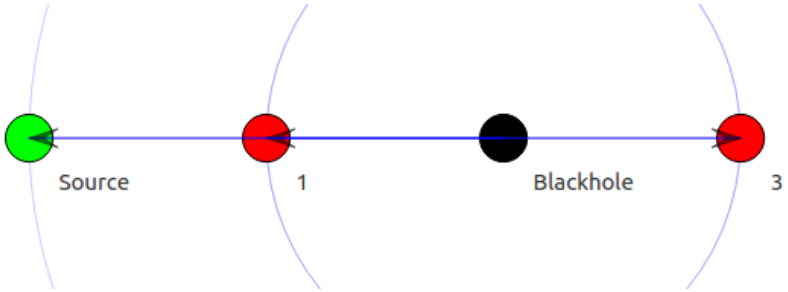
4.2 Skenario Simulasi

Skenario simulasi dibuat agar penulis dapat membandingkan perbedaan hasil dari simulasi yang dijalankan. Sehingga nantinya dapat diketahui apa yang mempengaruhi dan yang menyebabkan perbedaan hasil keluaran simulasi pada tiap-tiap skenario. Untuk itu penulis membuat beberapa skenario berbeda-beda seperti berikut ini.

4.2.1 Skenario 4 Node 1 Black Hole

Pada skenario simulasi yang pertama, dimulai dari 4 node dan 1 black hole. Skenario pertama adalah penulis membuat 4 buah node dan satu diantaranya adalah node black hole. Skenario ini adalah skenario default dari contoh simulasi blackhole yang penulis dapatkan dari internet. Penulis sedikit melakukan modifiksi pada file yang penulis dapatkan, yaitu penulis memberikan warna hijau sebagai penanda bahwa node tersebut merupakan node asal yang akan

mengirimkan paket dan memindahkan node black hole ke node kedua. Berikut adalah contoh tampilan animasi simulasi dari skenario pertama, seperti pada Gambar 4.6.

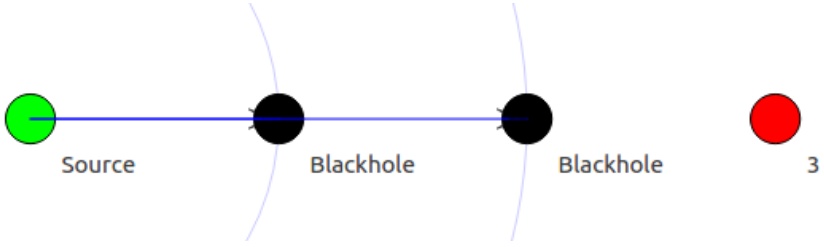


Gambar 4.6 Tampilan Node Skenario Pertama Pada NetAnim

Pada Gambar 4.6 diatas dapat dilihat bahwa node ke 0 di inisialisasi sebagai node sumber yang akan mengirimkan paket ke masing masing node. Node ke satu dan ketiga sebagai node yang akan menerima paket dari node ke 0, tetapi node ke dua berlaku sebagai node black hole atau node *malicious* yang akan mengganggu proses pengiriman node dari node sumber menuju node ke tiga. Hasil dari simulasi skenario ini dapat dilihat di tabel hasil skenario pertama pada lampiran.

4.2.2 Skenario 4 Node 2 Black Hole

Pada skenario kedua penulis masih menggunakan 4 node tetapi 2 diantaranya penulis atur sebagai node black hole. Untuk pengaturan penempatan posisi nodenya, penulis atur seperti pada Gambar 4.7.



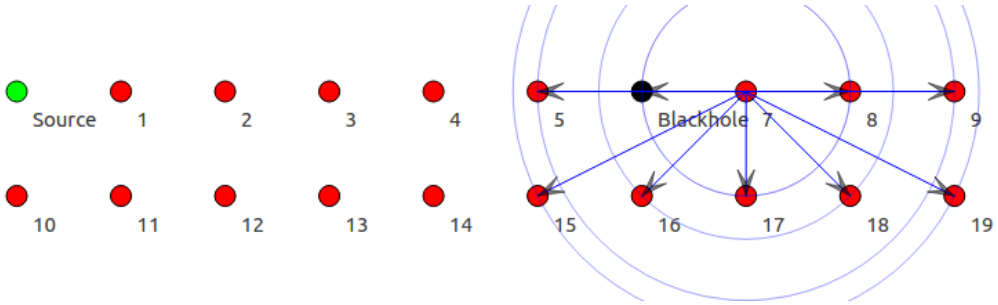
Gambar 4.7 Tampilan Node Skenario Kedua Pada NetAnim

Dari skenario simulasi pada Gambar 4.7, node black hole ditempatkan diantara node normal yang akan saling mengirim dan menerima paket yaitu pada node ke satu dan node ke dua. Proses pengiriman paket dari node asal dan node ketiga akan terhambat karena terdapat dua node black hole yang akan menghapus paket yang dikirimkan oleh node node asal, sehingga kemungkinan node ketiga untuk dapat menerima paket dari node sumber akan sangat

kecil kemungkinannya. Pada skenario kedua ini didapatkan hasil keluaran seperti tabel pada lampiran.

4.2.3 Skenario 20 Node 1 Black Hole

Pada skenario ketiga ini, penulis menambahkan jumlah node menjadi 20 node dengan 1 node blackhole. Skenario ini dibuat karena penulis ingin mengetahui seberapa besar perubahan hasil *throughput* dan *delay* simulasi jika jumlah node normal diperbanyak dari skenario sebelumya. Penulis masih mengatur node ke 0 sebagai node asal atau node sumber yang akan mengirim paket ke tiap-tiap node yang ada di jaringan, dan mengatur node ke enam yang menjadi node black hole atau node malicious yang akan menghapus paket yang akan dikirimkan oleh node sumber. Tampilan visual simulasi dari skenario ketiga ini dapat dilihat pada Gambar 4.8.

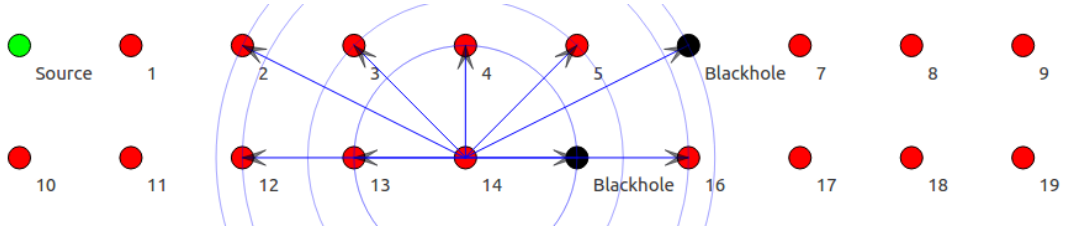


Gambar 4.8 Tampilan Node Skenario Ketiga Pada NetAnim

Pada skenario ini penulis ingin mengetahui besaran nilai *throughput*, *delay* dan *packet loss* karena seharusnya semakin banyak node dalam jaringan maka nilai *throughput* semakin kecil, nilai delay semakin tinggi dan semakin banyak paket yang tidak sampai ke node tujuan dibandingkan dengan skenario 4 node sebelumnya. Penulis menempatkan node black hole secara acak yang pada gambar diatas node black hole berada pada node ke 6. Jika dilihat dari topologi diatas seharusnya paket-paket yang kirimkan node sumber tidak akan sampai pada node yang berada dibelakang node black hole. Hasil simulasi pada skenario ini dapat dilihat di tabel hasil skenario ketiga pada lampiran.

4.2.4 Skenario 20 Node 2 Black Hole

Skenario keempat penulis masih menggunakan 20 node, tetapi penulis menempatkan 2 black hole dalam jaringan. Skenario ini dibuat untuk mengetahui perbedaan dengan skenario ketiga. Tampilan visual pada skenario keempat ini dapat dilihat pada Gambar 4.9.

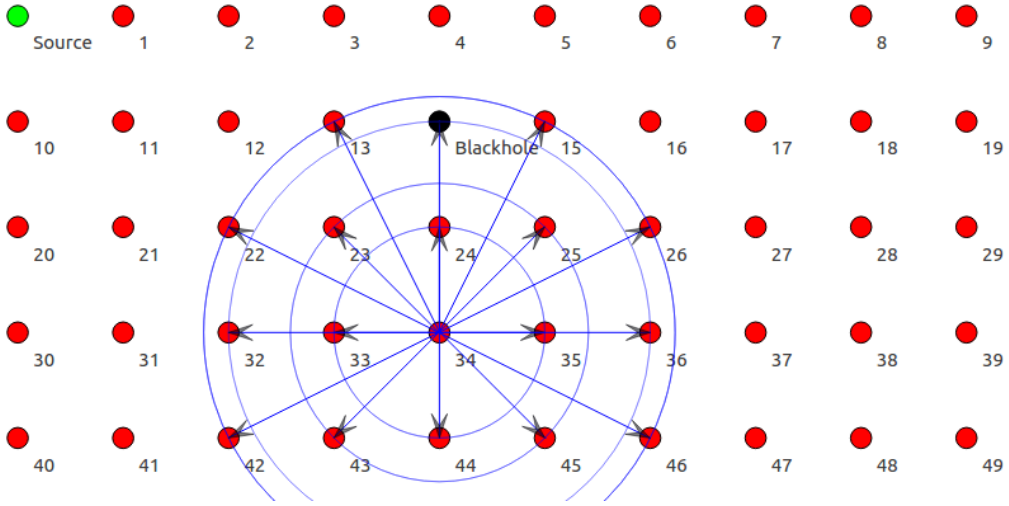


Gambar 4.9 Tampilan Node Skenario Keempat Pada NetAnim

Dari Gambar 4.9 diatas, dapat diketahui node black hole kedua berada dibawah node black hole pertama. Seharusnya pada skenario keempat ini akan lebih banyak paket yang hilang dibandingkan dengan skenario ketiga. Jika *packet loss* besar, maka seharusnya nilai *throughput* dan *delay* pada skenario ini lebih kecil dibandingkan dengan skenario ketiga. Hasil simulasi pada skenario ini dapat dilihat di tabel hasil skenario keempat pada lampiran.

4.2.5 Skenario 50 Node 1 Black Hole

Skenario simulasi kelima, penulis menambahkan lebih banyak lagi node normal ke dalam jaringan MANET sebanyak 50 buah node. Skenario ini penulis ingin membandingkan lagi nilai *throughput* dan *delay* yang akan dihasilkan jika terdapat 50 node dalam jaringan. Tampilan visual pada skenario kelima ini dapat dilihat pada Gambar 4.10.



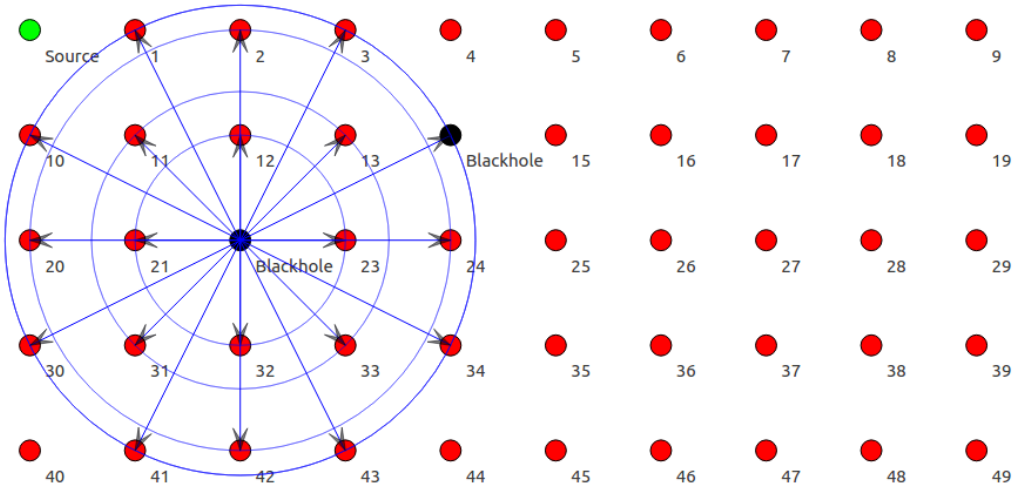
Gambar 4.10 Tampilan Node Skenario Kelima Pada NetAnim

Pada skenario ini penulis ingin mengetahui lebih jauh lagi bila node dalam jaringan lebih banyak dari skenario 20 node diatas. Penulis membuat 50 node dalam jaringan dan menempatkan 1 node black hole secara acak. Penulis berasumsi pada skenario ini nilai *throughput* lebih kecil dari skenario 20 node dan *delay* lebih besar dari pada skenario 20 node karena jumlah node yang lebih padat. Tetapi untuk *paket loss* bergantung dari *routing table*

node sumber. karena semakin banyak node dalam jaringan, maka semakin banyak rute yang dapat dipilih node sumber untuk mengirimkan paketnya. Hasil simulasi pada skenario ini dapat dilihat di tabel hasil skenario kelima pada lampiran.

4.2.6 Skenario 50 Node 2 Black Hole

Skenario terakhir yang dibuat penulis adalah 50 node dengan 2 node black hole. Skenario ini dibuat untuk membandingkan hasil *throughput*, *delay* dan *packet loss* pada skenario kelima. Tampilan visual pada skenario keenam dapat dilihat pada Gambar 4.11.



Gambar 4.11 Tampilan Node Skenario Keenam Pada NetAnim

Berdasarkan Gambar 4.11 diatas, penulis berasumsi jika dilihat dari letak kedua node black hole yang berada dekat dengan node sumber, maka akan banyak paket yang hilang dan membuat nilai *throughput* dan *delay* akan lebih kecil dari skenario sebelumnya, dan nilai *packet loss* akan lebih tinggi daripada skenario sebelumnya. Tetapi hal tersebut dapat terjadi yang sebaliknya bergantung dari *routing table* dari node sumber yang memiliki banyak pilihan rute untuk mengirimkan paketnya. Hasil simulasi pada skenario ini dapat dilihat di tabel hasil skenario keenam pada lampiran.

4.3 Analisis Data

Pada tahapan ini, akan dianalisa data hasil simulasi untuk diketahui perbandingannya pada setiap skenario yang dibuat sebelumnya. Penulis membagi data menjadi tiga kelompok, yakni kelompok 4 node, kelompok 20 node dan kelompok 50 node. Ketiga kelompok tersebut sama sama akan dibandingkan ketika mendapat 1 node black hole dan 2 node black hole, serta dibandingkan juga dengan keadaan jaringan yang tanpa adanya node black hole. Sebagai

perbandingan, tabel hasil simulasi yang tidak terdapat node black hole dalam jaringan dilampirkan oleh penulis di lampiran.

4.3.1 Analisis Data Simulasi Skenario 1 dan 2

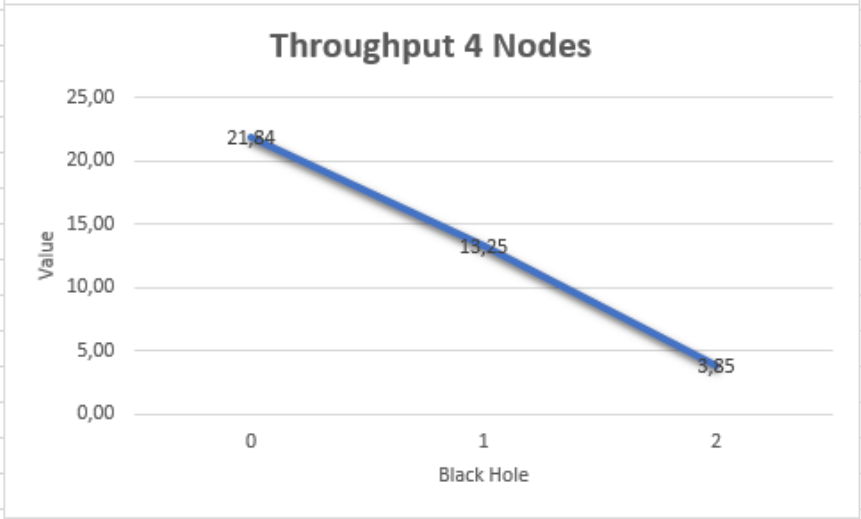
Pembahasan data hasil simulasi pada skenario yang dibuat akan dibagi dalam 3 bagian sesuai parameter yang diuji yaitu:

1. Throughput

Analisa *throughput* yang dilakukan berdasarkan rata-rata pada hasil skenario yang telah dilakukan. Hasil perhitungan *throughput* pada simulasi skenario 1 dan 2, serta simulasi tanpa node black hole dapat dilihat pada Tabel 4.1 dan grafik yang merepresentasikannya dapat dilihat pada Gambar 4.12.

Tabel 4.1 Rata-Rata *Throughput* Skenario 1 dan 2

Jumlah Node	Rata – Rata Throughput (Kbps)
0 Node Black Hole	21,84
1 Node Black Hole	13,25
2 Node Black Hole	3,85
Rata-Rata Keseluruhan	12,98



Gambar 4.12 Grafik *Troughput* Simulasi Skenario 1 dan 2

Berdasarkan grafik pada Gambar 4.12 diatas, dapat dilihat nilai rata-rata *throughput*, semakin bertambahnya node black hole, maka nilai *throughput* semakin turun. Rata-rata penurunan *throughput* pada skenario 4 node ini sebesar 12,98 Kbps. Hal ini disebabkan karena node black hole yang menghapus paket data yang dikirimkan dari node asal sehingga semakin

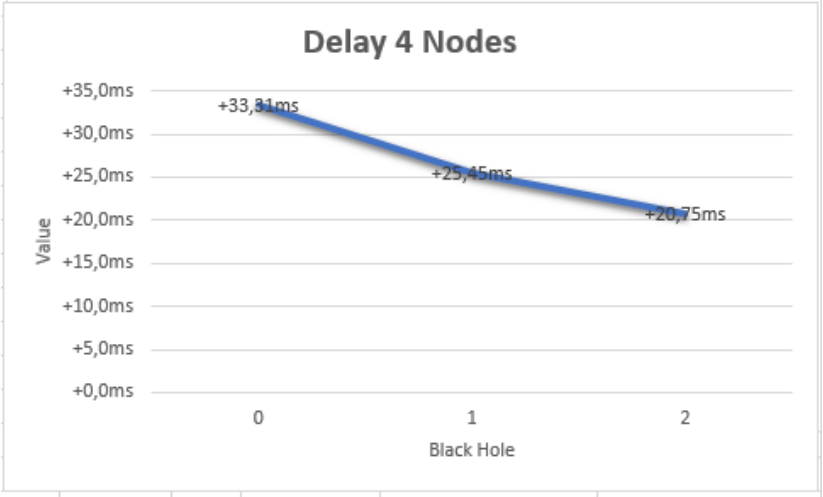
sedikit paket data yang bertransmisi dalam jaringan. Nilai yang dihasilkan simulasi juga dapat dipengaruhi karena letak black hole yang berada dekat dengan node sumber, sehingga banyak paket data yang terperangkap ke node black hole.

2. Delay

Analisa *delay* pada skenario 1 dan 2 juga dilihat dari nilai rata-rata *delay* yang dihasilkan saat simulasi. Penulis juga mencantumkan rata-rata *delay* tanpa node black hole. Rata-rata *delay* yang dihasilkan dapat dapat dilihat pada Tabel 4.2 dan Gambar 4.13.

Tabel 4.2 Rata-Rata *Delay* Skenario 1 dan 2

Jumlah Node	Rata – Rata Delay
0 Black Hole	+33,31ms
1 Black Hole	+25,45ms
2 Black Hole	+20,75ms
Rata–Rata Keseluruhan	+26,5ms



Gambar 4.13 Grafik *Delay* Skenario 1 dan 2

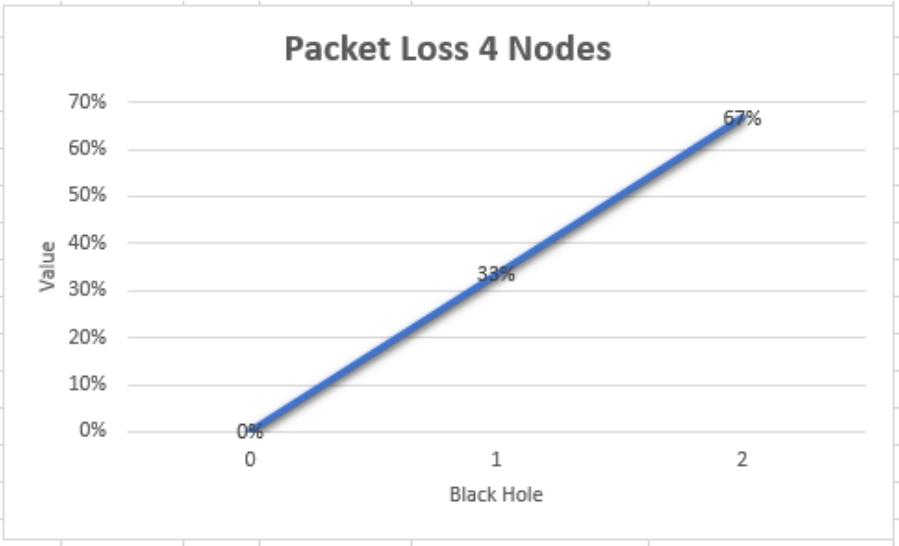
Berdasarkan grafik pada Gambar 4.13 diatas dapat dilihat bahwa *delay* yang terjadi semakin menurun dengan semakin banyaknya node black hole yang ada dalam jaringan. Rata-rata penurunan *delay* sebesar +26,5ms. Hal ini terjadi karena pada skenario ini, semakin banyak node black hole maka semakin besar kemungkinan paket yang hilang karena terperangkap node black hole, sehingga tidak ada paket yang bertransmisi dalam jaringan, membuat tidak ada nilai *delay* yang dihitung dan membuat nilai *delay* semakin menurun.

3. Packet Loss

Analisa *packet loss* dapat dilihat dari persentase hilangnya paket selama simulasi di skenario 1 dan 2 ini dapat dilihat pada Tabel 4.3 dan grafik yang merepresentasikan pada Gambar 4.14.

Tabel 4.3 *Packet Loss* Skenario 1 dan 2

Jumlah Node	Packet Loss
0 Black Hole	0%
1 Black Hole	33%
2 Black Hole	67%
Rata–Rata keseluruhan	33%



Gambar 4.14 Grafik *Packet Loss* Skenario 1 dan 2

Dari grafik *packet loss* pada Gambar 4.14 yang dihasilkan, persentase terjadinya packet loss dalam jaringan skenario 1 dan 2 semakin meningkat, berbanding lurus dengan peningkatan jumlah node black hole dalam jaringan. Jika dilihat topologinya pada Gambar 4.7 tidak ada jalur lain yang dapat dipilih node sumber untuk mengirimkan paketnya selain melalui node black hole.

4.3.2 Analisis Data Simulasi Skenario 3 dan 4

Sama seperti dengan Analisa data skenario 1 dan 2, Analisa data skenario 3 dan 4 juga akan dianalisa *throughput, delay* dan *packet loss.* Analisa skenario 3 dan 4 dilakukan karena

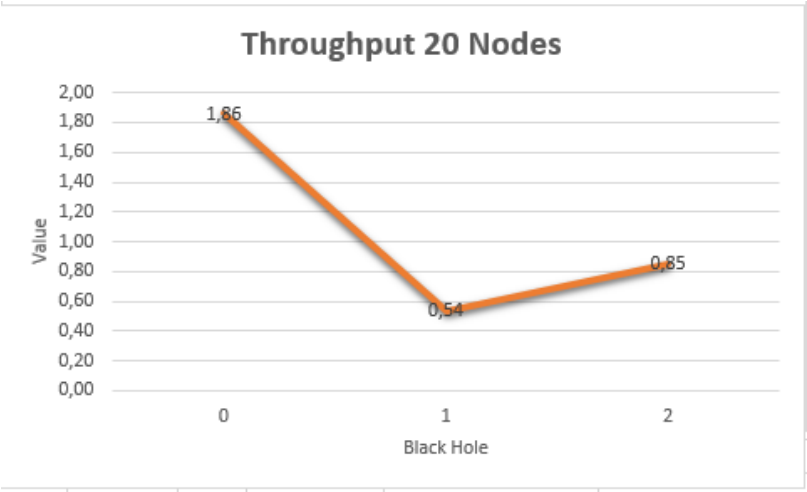
untuk mengetahui perbedaan pengaruh node black hole pada jaringan antara 4 node dan 20 node.

1. Throughput

Analisa *throughput* pada skenario 3 dan 4 dapat dilihat dari rata-rata *throughput* yang telah dihitung dari simulasi yang dijalankan pada skenario 3 dan 4. Hasil perhitungan rata-rata *throughput* dapat dilihat pada Tabel 4.4 dan grafik pada Gambar 4.15.

Tabel 4.4 Rata-Rata *Throughput* Skenario 3 dan 4

Jumlah Node	Rata – Rata Throughput
0 Black Hole	1,86
1 Black Hole	0,54
2 Black Hole	0,85
Rata-Rata Keseluruhan	1,08



Gambar 4.15 Grafik *Troughput* Skenario 3 dan 4

Dari grafik pada Gambar 4.15 dapat dilihat nilai rata-rata *throughput* turun dan naik. Pada skenario ke 3 yang mana terdapat 1 node black hole dan 20 node, nilai *throughput* mengalami penurunan dibandingkan jaringan yang tidak terdapat node black holenya. Tetapi terjadi peningkatan nilai *throughput* pada skenario 2 node black hole, hal ini dapat terjadi karena beberapa faktor, salah satunya pada simulasi skenario 3 letak node black hole yang mempengaruhi *routing table* yang membuat node sumber mengirimkan paket melewati node black hole sebelum sampai ke node tujuan. Sehingga banyak paket yang terperangkap ke node black hole, terutama pada node-node tujuan yang dekat dengan node sumber yang memiliki

nilai throughput lebih besar daripada node yang letaknya jauh dari node sumber, sehingga sedikit yang bertransmisi pada jaringan tersebut dan membuat rata-rata throughput pada skenario ke 3 menjadi rendah.

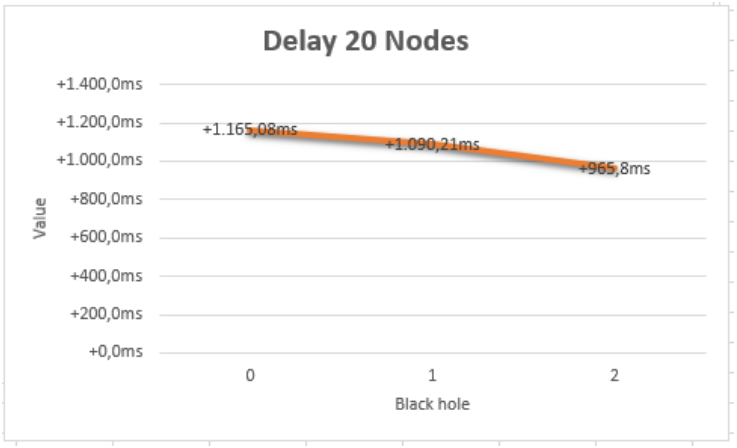
Dibandingkan dengan skenario yang ke 4, walaupun memiliki node black hole yang lebih banyak dari skenario ke 3, tetapi letak node black hole membuat paket yang hilang terjadi pada node-node yang letaknya jauh dari node sumber yang nilai throughputnya tidak besar, sehingga membuat rata-rata nilai throughput pada skenario 4 lebih tinggi jika dibandingkan dengan rata-rata throughput skenario ke 3. Rata-rata perubahan nilai *throughput* pada skenario 4 node ini sebesar 1,08 Kbps

2. Delay

Analisa delay pada skenario 3 dan 4 dibandingkan dengan melihat hasil perhitungan rata-rata delay dari skenario 3 dan 4. Hasil perhitungan rata-rata delay pada skenario 3 dan 4 dapat dilihat pada Tabel 4.5.

Tabel 4.5 Rata-Rata *Delay* Skenario 3 dan 4

Jumlah Node	Rata – Rata Delay
0 Black Hole	+1.165,08ms
1 Black Hole	+1.090,21ms
2 Black Hole	+965,8ms
Rata – rata Keseluruhan	+1.073,7ms



Gambar 4.16 Grafik *Delay* Skenario 3 dan 4

Dari grafik yang ditampilkan Gambar 4.16, dapat dilihat bahwa rata-rata delay yang dihasilkan pada skenario 3 dan 4 semakin menurun. Karena semakin banyak node black hole,

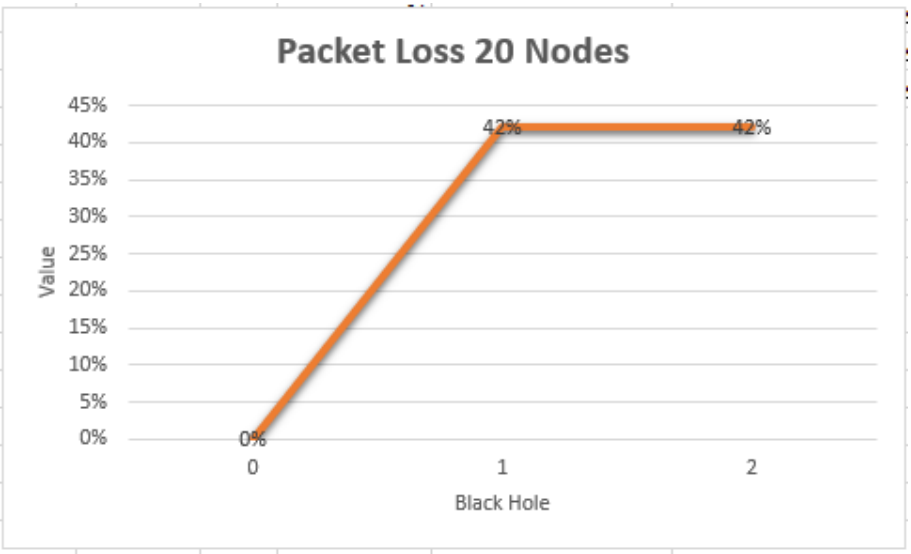
semakin banyak paket yang terperangkap dan tidak sampai ke tujuan, sehingga sedikit terjadi proses transmisi paket dalam jaringan yang mengakibatkan tidak ada delay yang terjadi pada simulasi jaringan di skenario 3 dan 4. Rata-rata penurunan delay yang terjadi pada skenario 3 dan 4 ini adalah +1.073,7ms

3. Packet Loss

Analisa packet loss dapat dilihat dari persentase hilangnya paket selama simulasi di skenario 3 dan 4 ini dapat dilihat pada Tabel 4.6 dan grafik yang merepresentasikan ada Gambar 4.17.

Tabel 4.6 *Packet Loss* Skenario 3 dan 4

Jumlah Node	Packet Loss
0 Black Hole	0%
1 Black Hole	42%
2 Black Hole	42%
Rata – Rata keseluruhan	28%



Gambar 4.17 Grafik *Packet Loss* Skenario 3 dan 4

Pada grafik pada Gambar 4.17 diatas menunjukkan bahwa packet loss pada simulasi skenario 3 dan 4 sama-sama menunjukkan angka 42%. Tetapi jika dilihat dari tabel hasil simulasi skenario 3 dan 4 yang penulis lampirkan, paket yang hilang atau tidak sampai ke node tujuan, berbeda antara skenario 3 dengan skenario 4. Perbedaan itu dipengaruhi karena letak node black hole yang dapat dilihat pada Gambar 4.8 dan Gambar 4.9.

4.3.3 Analisis Data Simulasi Skenario 5 dan 6

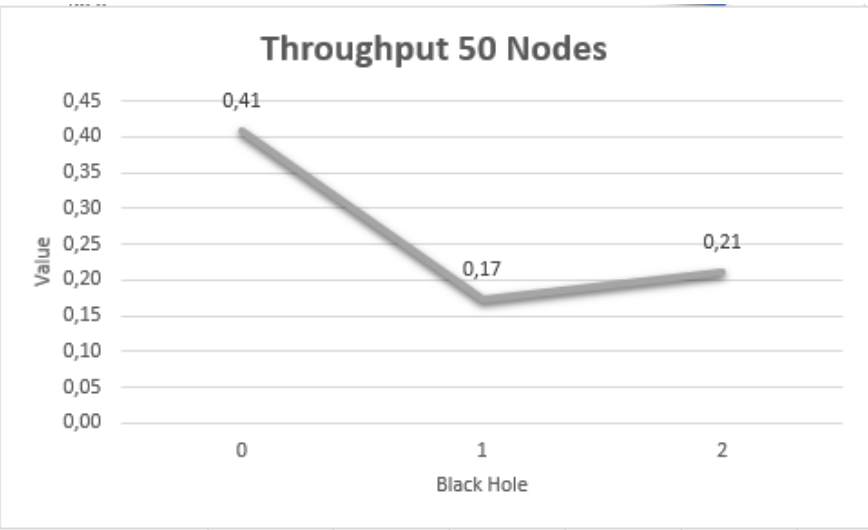
Analisa data skenario 5 dan 6, juga akan dianalisa *throughput*, *delay* dan *packet loss*. Analisa skenario 5 dan 6 dilakukan karena untuk mengetahui perbedaan pengaruh node black hole pada jaringan antara 4 node, 20 node dan 50 node.

1. Throughput

Analisa *throughput* pada skenario 5 dan 6 dilihat dari rata-rata *throughput* yang telah dihitung dari simulasi jaringan yang dijalankan pada skenario 5 dan 6. Hasil perhitungan rata-rata *throughput* dapat dilihat pada Tabel 4.7 dan grafik pada Gambar 4.18.

Tabel 4.7 Rata-Rata *Throughput* Skenario 5 dan 6

Jumlah Node	Throughput
0 Black Hole	0,41
1 Black Hole	0,17
2 Black Hole	0,21
Rata – Rata keseluruhan	0,26



Gambar 4.18 Grafik *Througput* Skenario 5 dan 6

Dari grafik pada Gambar 4.18 dapat dilihat nilai rata-rata *throughput* turun dan naik sama seperti dengan skenario 3 dan 4. Pada skenario ke 5 yang mana terdapat 1 node black hole dan 50 node, nilai *throughput* lebih rendah dibandingkan dengan nilai *throughput* jaringan keadaan normal dan *throughput* skenario 6. Jika dilihat dari tabel hasil simulasi skenario 5 dan 6 yang dilampirkan, node black hole pada skenario 5 lebih banyak menghapus paket data yang ada di

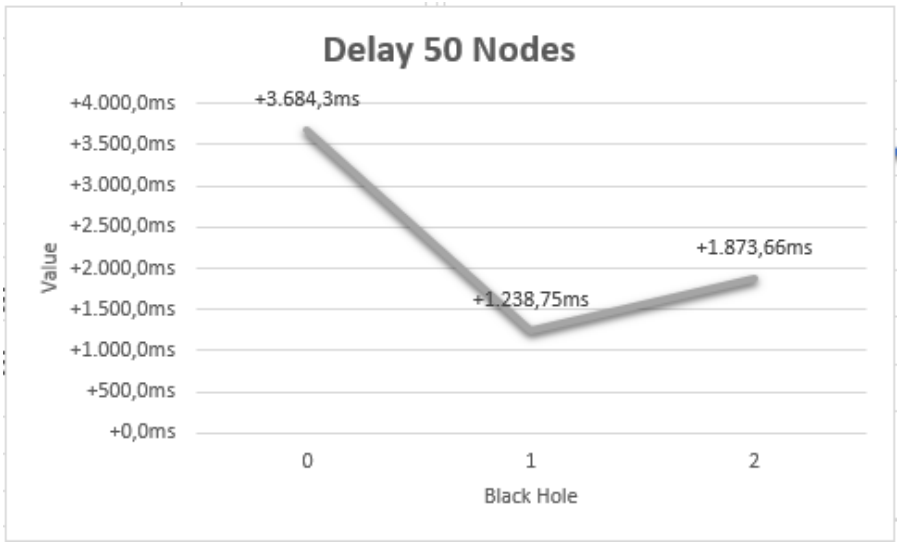
jaringan, dibandingkan dengan skenario 6, sehingga rata-rata *throughput* yang dihasilkan lebih rendah daripada rata-rata *throughput* dari skenario 6. Rata-rata perubahan nilai *throughput* pada skenario 5 dan 6 node ini sebesar 0,26 Kbps

2. Delay

Analisa *delay* pada skenario 5 dan 6 dibandingkan dengan melihat hasil perhitungan rata-rata *delay* dari skenario 3 dan 4. Hasil perhitungan rata-rata *delay* pada skenario 5 dan 6 dapat dilihat pada Tabel 4.8.

Tabel 4.8 Rata-Rata *Delay* Skenario 5 dan 6

Jumlah Node	Delay
0 Black Hole	+3.684,3ms
1 Black Hole	+1.238,75ms
2 Black Hole	+1.873,66ms
Rata – Rata keseluruhan	+2.265,57ms



Gambar 4.19 Grafik *Delay* Skenario 5 dan 6

Dari grafik *delay* yang ditampilkan Gambar 4.19 dapat dilihat bahwa rata-rata *delay* yang dihasilkan pada skenario 5 dan 6 menunjukkan grafik yang menurun menjadi +1.238,75ms, tetapi pada skenario 6 *delay* kembali naik menjadi +1.873,66ms. Jika dilihat di tabel hasil simulasi, black hole pada skenario 5 lebih banyak menghapus paket yang bertransmisi dalam jaringan, terutama paket-paket yang mempunyai tujuan di node-node yang jauh dari node sumber yang pada keadaan normal menghasilkan nilai *delay* yang tinggi, sehingga

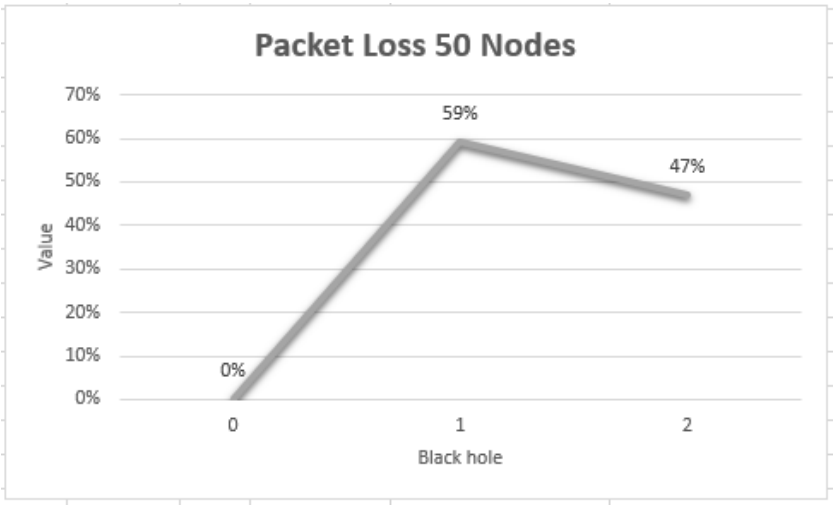
menghasilkan rata-rata *delay* yang rendah. Sedangkan pada skenario 6, walaupun memiliki node black hole yang lebih banyak dari skenario 5, tetapi paket yang hilang dalam jaringan saat bertransmisi lebih sedikit daripada skenario 5, sehingga membuat rata-rata *delay* yang dihasilkan lebih tinggi daripada skenario 5. Rata-rata perubahan *delay* yang terjadi pada skenario 5 dan 6 ini adalah +2.265,57ms

3. Packet Loss

Analisa *packet loss* dapat dilihat dari persentase hilangnya paket selama simulasi di skenario 5 dan 6 ini dapat dilihat pada Tabel 4.9 dan grafik yang merepresentasikan pada Gambar 4.20.

Tabel 4.9 *Packet Loss* Skenario 5 dan 6

Jumlah Node	Packet Loss
0 Black Hole	0%
1 Black Hole	59%
2 Black Hole	47%
Rata – Rata keseluruhan	35%

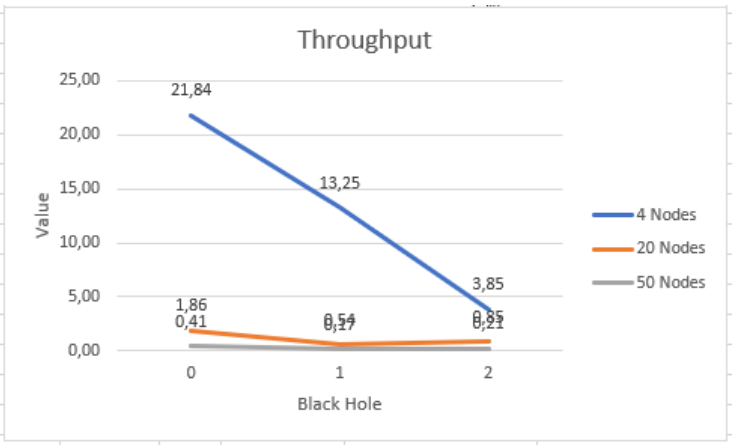


Gambar 4.20 Grafik *Packet Loss* Skenario 5 dan 6

Pada grafik pada Gambar 4.20 diatas menunjukkan bahwa *packet loss* pada simulasi skenario 5, *packet loss* mencapai 59% tetapi pada skenario 6 *packet loss* lebih rendah yaitu 47%. Jika dilihat di tabel hasil simulasi yang dilampirkan, paket yang hilang di skenario 5, lebih banyak berada pada node yang jauh dari node sumber. Perbedaan itu dipengaruhi karena letak node black hole yang dapat dilihat pada Gambar 4.10 dan Gambar 4.11.

4.3.4 Analisis Pengaruh Black Hole Terhadap Parameter Yang Diuji

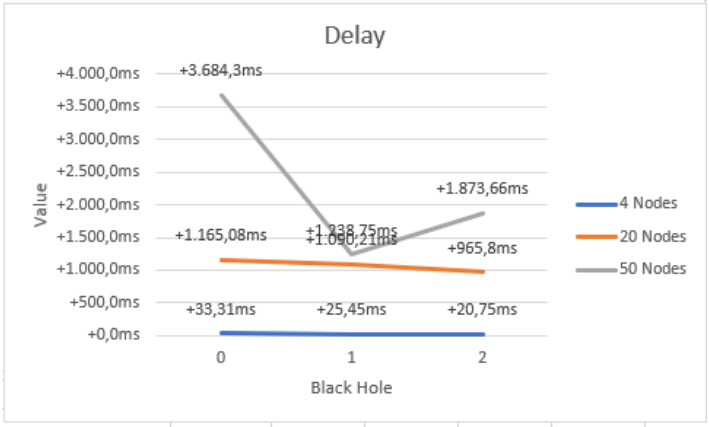
1. Throughput



Gambar 4.21 *Throughput* Simulasi Black Hole

Banyaknya jumlah node dalam jaringan mempengaruhi besaran *throughput* dalam jaringan tersebut. Semakin padat jumlah node dalam jaringan, maka semakin kecil *throughput* yang dihasilkan. Untuk pengaruhnya dengan node black hole, dari grafik pada Gambar 4.21 dapat dilihat semakin banyak node black hole dalam jaringan tidak selalu akan menurunkan *throughput* seketika, seperti pada skenario simulasi 20 node dan skenario simulasi 50 node. Black hole mempengaruhi *throughput* tergantung dari *routing table* pada node sumber untuk menuju node tujuan dan letak black hole itu sendiri. Jika letak black hole berada dekat dengan node sumber, maka kemungkinan black hole untuk menghapus paket yang dikirimkan node sumber menjadi lebih besar dan membuat rata-rata *throughput* menjadi lebih kecil karena tidak ada nilai *throughput* yang dihitung pada paket yang hilang. Analisa ini berbeda dengan kesimpulan penelitian Neelam Janak Kumar Patel, yang menyebutkan jaringan AODV normal tanpa serangan black hole, menghasilkan *throughput* yang tinggi. Ketika ada 1 serangan black hole, *throughput* berkurang, untuk 3 dan 5 serangan black hole, *throughput* jauh lebih berkurang. Sedangkan kesimpulan dari penelitian Istas Pratomo dan M Hizrian Hizburrahman menyebutkan, untuk pengukuran dengan metode *flow control* default, penurunan *throughput* disebabkan oleh ukuran jaringan yang semakin besar. Hal ini dikarenakan *throughput* dihitung dalam satuan flits/cycle, dengan ukuran paket, ukuran buffer dan packet injection rate yang tetap apabila ukuran jaringan diperbesar, maka ukuran waktu yang diperlukan oleh sebuah flit untuk mencapai tujuan akan bertambah, sehingga menyebabkan *throughput* menurun. Selain itu metode pengiriman yang hanya mengizinkan mengirimkan 1 flit sebelum menerima ack menjadi salah satu faktor yag menurunkan *throughput* apabila ukuran jaringan diperbesar.

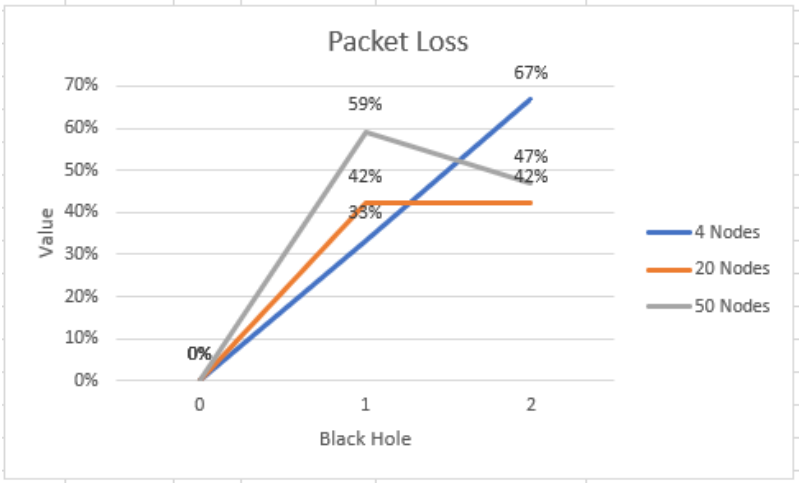
2. Delay



Gambar 4.22 *Delay* Simulasi Black hole

Dari grafik *delay* pada Gambar 4.22 diatas, dapat disimpulkan *delay* berkebalikan dengan *throughput.* Dalam keadaan normal, semakin banyak node dalam jaringan, maka akan menghasilkan nilai delay yang tinggi. Karena sifat dari *routing protocol* AODV yang akan mengirimkan RREQ terlebih dahulu untuk mencari rute ke node tujuan sebelum mengirimkan paket. Untuk pengaruhnya dengan node black hole, bergantung dari letak black hole dalam jaringan. jika letak black hole mampu mempengaruhi *routing table* node sumber untuk mengirimkan banyak paket melalui dirinya, maka nilai *delay* akan semakin rendah dari keadaan normal karena nilai *delay* pada paket yang hilang tidak dapat dihitung. Jika black hole tidak mampu membuat *routing table* node sumber melewatkan paket-paketnya melalui dirinya seperti pada skenario 6, maka paket tersebut akan sampai ke node tujuan dan membuat nilai rata-rata *delay* pada skenario 6 akan lebih tinggi daripada skenario 5. Analisa ini sesuai dengan kesimpulan penelitian Neelam Janak Kumar Patel yang menyebutkan jaringan AODV normal tanpa serangan black hole, *delay* terlihat seperti seharusnya di jaringan nirkabel, ketika ada 1 serangan black hole, *delay* menurun disebabkan oleh serangan, untuk 3 dan 5 serangan black hole, *delay* tidak ada nilai lagi yang terlihat oleh serangan black hole. Sedangkan kesimpulan dari penelitian Istas Pratomo dan M Hizrian Hizburrahman menyebutkan, untuk metode *flow control* DyML dan Ack memiliki karakteristik yang hampir sama, dimana terjadi peningkatan *delay* seiring dengan peningkatan ukuran jaringan dan pada ukuran jaringan tertentu dicapai *delay* maksimum sebelum *delay* yang terjadi pada jaringan menurun seiring dengan peningkatan dari ukuran jaringan. Sedangkan jaringan yang menggunakan metode *flow control* default, *delay* mengalami peningkatan ketika ukuran jaringan diperkecil. Karena pada *flow control* default, *delay* berbanding lurus dengan ukuran jaringan.

3. Packet Loss



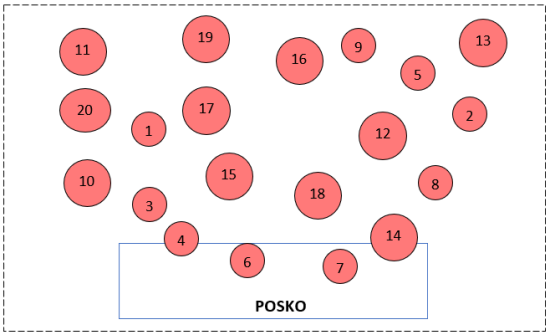
Gambar 4.23 *Packet Loss* Simulasi Black Hole

Dari grafik *packet loss* ada Gambar 4.23 diatas dapat dilihat bahwa banyaknya jumlah node black hole tidak berbanding lurus dengan persentase packet loss dalam jaringan. Persentase packet loss akan semakin besar bergantung dari topologi jaringan dan *routing table* dari node sumber itu sendiri, seperti pada simulasi skenario 1 dan 2. Walaupun node black hole lebih banyak, kalau *routing table* node sumber tidak melewatkan paketnya melalui node black hole untuk sampai ke node tujuan, maka tidak ada packet loss yang tercatat seperti pada simulasi skenario 3 dan 4. Dan semakin padat node dalam jaringan, maka semakin banyak pilihan rute bagi node sumber untuk mengirimkan paketnya menuju node tujuan, sehingga walaupun dalam jaringan lebih banyak black hole, paket yang hilang bisa saja lebih sedikit daripada jaringan yang memiliki lebih sedikit node black hole seperti yang terjadi pada simulasi skenario 5 dan 6. Analisa ini berbeda dengan kesimpulan penelitian Neelam Janak Kumar Patel, yang menyebutkan untuk jaringan AODV normal tanpa serangan black hole, paket yang hilang hanya sedikit, ketika ada 1 serangan black hole, paket yang hilang meningkat, untuk 3 dan 5 serangan black hole, paket yang hilang jauh lebih meningkat.

4.4 Pemanfaatan Hasil Simulasi dan Skenario Pencegahan Serangan Black hole.

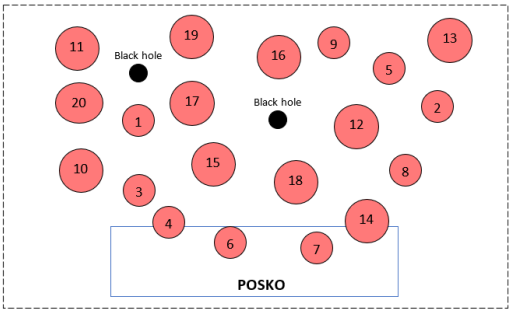
Seperti pada penjelasan diawal laporan bahwa MANET dapat digunakan pada daerah yang sedang dilanda bencana alam, MANET dapat digunakan sebagai jalur komunikasi sementara para relawan. Sebagai gambaran contoh dari penelitian ini, dibuatlah sebuah skenario bencana alam yang menghancurkan infrastuktur jaringan di wilayah tersebut. Kemudian para relawan memanfaatkan teknologi MANET untuk saling berkomunikasi selama proses pencarian korban

dan evakuasi. Dimisalkan ada 20 relawan dan masing – masing relawan membawa perangkat yang telah saling berkomunikasi menggunakan jaringan MANET. Sebelum mengirimkan paket data, masing-masing perangkat akan saling mengirimkan RREQ (Route Request) dan RREP (Route Reply) ke perangkat lainnya untuk mencari node tujuan paket data tersebut, dan rute yang akan dilaluinya. RREQ berisi ID node sumber, ID node tujuan, time-to-live (TTL) dan rute ke tujuan. Setelah RREQ sampai ke node tujuan, node tujuan akan memberi balasan dengan mengirimkan RREP yang berisi ID node sumber, ID node tujuan, time-to-live (TTL), sequence number dan Hop (Rahebi & Mehdi, 2009). Setelah RREP sampai ke node sumber, node tersebut akan mengupdate routing tablenya dan mulai mengirimkan paket data berupa informasi yang dikirimkan oleh relawan.



Gambar 4.24 Skenario Jaringan MANET

Setelah beberapa hari melakukan pencarian dan evakuasi, mulai muncul gangguan-gangguan saat saling berkomunikasi. Banyak informasi yang dikirimkan relawan, sering tidak diterima oleh relawan lainya. Ketika dilakukan pengecekkan pada *routing table* salah satu perangkat relawan, ternyata ada node mencurigakan dalam jaringan tersebut. Setelah di identifikasi ternyata ada serangan black hole pada jaringan diwilayah tersebut.



Gambar 4.25 Skenario Serangan Black Hole dalam Jaringan MANET

Setelah diketahuinya serangan black hole, maka dilakukan upaya pencegahan agar paket yang dikirimkan oleh relawan tidak melewati node black hole. Cara pencegahan jaringan terkena black hole menurut penelitian Setio Adiwicaksono pada tahun 2017, ada 4 langkah yang perlu dilakukan. Yang pertama membuat daftar blacklist. Daftar ini digunakan untuk mengatur rute pengiriman agar tidak melewati node yang mempunyai *IP Address* yang terdaftar di daftar blacklist. Langkah kedua adalah pendaftaran node berbahaya kedalam blacklist. Ketika node berbahaya sudah diinstansiasikan sebagai node berbahaya atau node black hole, maka secara otomatis *IP Address* node black hole tersebut akan dimasukkan ke daftar blacklist. Sehingga pendeteksian perangkat relawan tentang adanya node black hole atau tidak adalah dari pencocokan alamat node tersebut pada daftar berbahaya blacklist. Langkah ketiga yaitu pencegahan blackhole yang berada di dalam zona. Langkah ketiga ini dilakukan dengan cara mengecek *IP Address* tetangganya apakah ada didaftar blacklist atau tidak. Jika ada, maka node tersebut tidak akan ditambahkan pada daftar tetangga pada perangkat relawan. Dan langkah terakhir adalah pencegahan black hole yag berada diluar zona. Node sumber tidak mengetahui alamat rute jika destinasi berada diluar zona. Maka perlu ada pencegahan jika terdapat black hole diluar zona. Cara pencegahannya adalah ketika node sumber memberikan route request ke semua *peripheral* node nya, maka *peripheral* node akan melakukan persis seperti yang node sumber lakukan, lalu akan menambahkan alamat rute miliknya ke paket route request nya. Pencegahan dilakukan jika alamat node yang akan disambungkan maupun node yang akan menyambungkan terdaftar pada blacklist, maka tidak akan ada penambahan link pada daftar link (linklist) dari maupun menuju node tersebut (Adiwicaksono, 2017).

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Setelah menyelesaikan beberapa tahapan penelitian mulai dari mempelajari beberapa referensi dan penelitian sebelumnya, serta hasil analisa dari penelitian yang dilakukan oleh penulis, maka dapat diambil kesimpulan bahwa:

1. Berdasarkan hasil uji coba menggunakan beberapa parameter QoS, menunjukan bahwa black hole dapat menurunkan kualitas jaringan.
2. Penurunan kualitas jaringan terhadap black hole bergantung pada topologi jaringan yang sedang terbentuk dan letak black hole itu sendiri.
3. Semakin padat jumlah node dalam jaringan, maka *Throughput* yang dihasilkan akan semakin menurun. Karena besar ukuran *bandwidth* harus dibagi dengan lebih banyak node dalam jaringan. Tetapi banyak black hole tidak mempengaruhi penurunan *Throughput.*
4. Semakin banyak node black hole tidak menjamin akan semakin banyak paket data yang hilang, karena proses pengiriman paket bergantung rute yang ada di *routing table* pada node sumber.
5. Semakin padat node dalam jaringan, mampu mengurangi dampak black hole, karena node sumber akan mempunyai banyak pilihan rute pada *routing table* yang dapat digunakan untuk mengirimkan paketnya agar sampai ke node tujuan.

5.2 Saran

Saran yang diberikan untuk pengembangan sistem ini adalah:

1. Perlu adanya mekanisme pada saat simulasi untuk memunculkan *routing table* pada node sumber agar diketahui node-node yang dilalui paket yang dikirim node sumber sebelum sampai ke node tujuan.
2. Perlu adanya pengenalan sifat-sifat node black hole yang ada pada dunia nyata sehingga perilaku node black hole saat simulasi bisa sesuai dengan sifat-sifat asli black hole node.
3. Perlu pengujian dengan lebih banyak node black hole dalam jaringan agar bisa mendapatkan data yang lebih akurat lagi.

DAFTAR PUSTAKA

Adiwicaksono, S. (2017). DETEKSI MALICIOUS NODE PADA ZONE ROUTING PROTOCOL DI JARINGAN MOBILE ADHOC.

Ali Pangera, A. (2008). *Menjadi Administrator Jaringan Nirkabel*. (F. Sigit Suryantoro, Ed.). Yogyakarta: ANDI.

Asrullah. (2015). PERANCANGAN DAN EVALUASI KINERJA MANET MENGGUNAKAN SIMULATOR JARINGAN NS-3 UNTUK KAPAL NELAYAN DI PERAIRAN LAUT ACEH. Retrieved March 6, 2018, from http://etd.unsyiah.ac.id/baca/index.php?id=11181&page=21

Chitkara, M., & Ahmad, M. W. (2014). Review on MANET: Characteristics, Challenges, Imperatives and Routing Protocols. *International Journal of Computer Science and Mobile Computing*, *32*(2), 432–437. Retrieved from http://ijcsmc.com/docs/papers/February2014/V3I2201499a3.pdf

Ekaputra, M. Y. (2016). PEMANFAATAN TEKNOLOGI MOBILE AD-HOC NETWORK (MANET) DAN SIMULASINYA MENGGUNAKAN NETWORK SIMULATOR 3 (NS-3) (STUDI KASUS KOMUNITAS TRAIL ADVENTURE BALIKPAPAN), *3*.

Irawan, D. (2011). Simulasi Model Jaringan Mobile Ad-Hoc (Manet) Dengan Ns-3. *Badan Pengkajian Dan Penerapan Teknologi, Jakarta. Jurnal Konferensi Nasional Sistem Dan Informatika 2011; Bali, November 12, 2011.*, 335–339.

Jiatmiko, N. (2015). Simulasi Model Jaringan Mobile Ad-Hoc Network (Manet) Dengan Ns3 Untuk Membandingkan Performa Routing Protokol, (7), 104.

Khozaimi, A. (2013). Catatan khozaimi: MANET : Karakteristik. Retrieved August 22, 2017, from http://khozaimi.blogspot.co.id/2013/05/manet-karakteristik.html

Nsnam. (2011). What is NS 3. Retrieved November 19, 2017, from https://www.nsnam.org/overview/what-is-ns-3/

Nusanet. (2016). Standar Protokol Jaringan Wireless IEEE 802.11. Retrieved November 19, 2017, from https://www.nusa.net.id/blog/article/standar-protokol-jaringan-wireless-ieee-802-11/

Patel, N. J. K., & Tripathi, K. (2017). Analysis of Black Hole Attack in MANET Based on Simulation through NS3.26, *5*(5), 194–205.

Pratama, A. nur, Azaim, M. H., & Fauzi, V. (2015). Routing Protocol (AODV, DSR, DSDV). Retrieved November 19, 2017, from

http://menulicious.student.telkomuniversity.ac.id/routing-protocol-aodv-dsr-dsdv/

Pratomo, I., & Hizburrahman, M. H. (2015). Pendeteksian Dan Pencegahan Serangan Black Hole & Grey Hole Pada Manet, *13*(4), 47–53. Retrieved from javajournal.its.ac.id/index.php/java/article/download/39/37

Rahebi, S., & Mehdi, A. (2009). WBRR : A Weight Based Reliable Routing Method in Mobile Ad Hoc Network, (January).

Sanjaya, A. (2015). Pengertian Wireless Jenis Teknologi Nirkabel (WPAN, WWAN, WLAN) MANET, WMN dan Ad Hoc dan Infrastruktur. Retrieved November 20, 2017, from http://www.landasanteori.com/2015/10/pengertian-wireless-jenis-teknologi.html

Sasson, Y., Cavin, D., & Schiper, A. (2003). Probabilistic Broadcast for Flooding in Wireless Mobile Ad hoc Networks.

LAMPIRAN

1. Source code program simulasi black hole

```
1   #include "ns3/aodv-module.h"
2   #include "ns3/core-module.h"
3   #include "ns3/network-module.h"
4   #include "ns3/internet-module.h"
5   #include "ns3/applications-module.h"
6   #include "ns3/mobility-module.h"
7   #include "ns3/wifi-module.h"
8   #include "ns3/flow-monitor-module.h"
9   #include "ns3/mobility-module.h"
10  #include "myapp.h"
11  #include "ns3/netanim-module.h"
12  #include <fstream>
13  #include <iostream>
14
15  NS_LOG_COMPONENT_DEFINE ("SkripsiBlackhole");
16
17  using namespace ns3;
18
19    int txpower = 50;
20    int titik = 50;
21    uint32_t bytesTotal = 0;
22    uint32_t packetsReceived = 0;
23    std::string m_CSVfileName ("blackhole.output.csv");
24    int m_nSinks;
25    std::string m_protocolName ("AODV");
26    double m_txp;
27    uint32_t m_protocol;
28    bool attackenable = true;
29
30  void
31  ReceivePacket(Ptr<const Packet> p, const Address & addr)
32  {
33    std::cout << Simulator::Now ().GetSeconds () << "\t" << p-
34  >GetSize() <<"\n";
35  }
36
37  int main (int argc, char *argv[])
38   {
39    bool enableFlowMonitor = false;
40    std::string phyMode ("DsssRate1Mbps");
41
42    CommandLine cmd;
43    cmd.AddValue ("EnableMonitor", "Enable Flow Monitor",
44  enableFlowMonitor);
45    cmd.AddValue ("phyMode", "Wifi Phy mode", phyMode);
46    cmd.AddValue ("CSVfileName", "The name of the CSV output file
47  name", m_CSVfileName);
48    cmd.Parse (argc, argv);
49
50  // Explicitly create the nodes required by the topology (shown
51  above).
52
53    NS_LOG_INFO ("Create nodes.");
```

```
54    NodeContainer c; // ALL Nodes
55    NodeContainer not_malicious;
56    NodeContainer malicious;
57    c.Create(titik);
58
59    if(attackenable==false){
60      for(int i=0;i<titik;i++){
61        not_malicious.Add(c.Get(i));
62      }
63    }
64    else{
65      for(int i=0;i<titik;i++){
66        if(i==14 || i==22){  //i==1 || i==2 || i==6 || i==14 ||
67  i==15 || i==22
68          malicious.Add(c.Get(i));
69        }
70        else{
71          not_malicious.Add(c.Get(i));
72        }
73      }
74    }
75
76    // Set up WiFi
77    WifiHelper wifi;
78
79    YansWifiPhyHelper wifiPhy =  YansWifiPhyHelper::Default ();
80    wifiPhy.SetPcapDataLinkType
81  (YansWifiPhyHelper::DLT_IEEE802_11);
82
83    YansWifiChannelHelper wifiChannel ;
84    wifiChannel.SetPropagationDelay
85  ("ns3::ConstantSpeedPropagationDelayModel");
86    wifiChannel.AddPropagationLoss
87  ("ns3::TwoRayGroundPropagationLossModel",
88                                     "SystemLoss", DoubleValue(1),
89                                     "HeightAboveZ",
90  DoubleValue(1.5));
91
92    // For range near 250m
93    wifiPhy.Set ("TxPowerStart", DoubleValue(txpower));
94    wifiPhy.Set ("TxPowerEnd", DoubleValue(txpower));
95    wifiPhy.Set ("TxPowerLevels", UintegerValue(1));
96    wifiPhy.Set ("TxGain", DoubleValue(0));
97    wifiPhy.Set ("RxGain", DoubleValue(0));
98    wifiPhy.Set ("EnergyDetectionThreshold", DoubleValue(-61.8));
99    wifiPhy.Set ("CcaMode1Threshold", DoubleValue(-64.8));
100
101   wifiPhy.SetChannel (wifiChannel.Create ());
102
103   // Add a non-QoS upper mac
104   NqosWifiMacHelper wifiMac = NqosWifiMacHelper::Default ();
105   wifiMac.SetType ("ns3::AdhocWifiMac");
106
107   // Set 802.11b standard
108   wifi.SetStandard (WIFI_PHY_STANDARD_80211b);
109
110   wifi.SetRemoteStationManager ("ns3::ConstantRateWifiManager",
111                                 "DataMode",StringValue(phyMode),
112
113 "ControlMode",StringValue(phyMode));
```

```
114
115     NetDeviceContainer devices;
116     devices = wifi.Install (wifiPhy, wifiMac, c);
117
118   //  Enable AODV
119     AodvHelper aodv;
120     AodvHelper malicious_aodv;
121
122   // Set up internet stack
123     InternetStackHelper internet;
124     internet.SetRoutingHelper (aodv);
125     internet.Install (not_malicious);
126
127     malicious_aodv.Set("IsMalicious",BooleanValue(true)); //
128   putting *false* instead of *true* would disable the malicious
129   behavior of the node //true
131     internet.SetRoutingHelper (malicious_aodv);
132     internet.Install (malicious);
133
134   // Set up Addresses
135     Ipv4AddressHelper ipv4;
136     NS_LOG_INFO ("Assign IP Addresses.");
137     ipv4.SetBase ("10.1.2.0", "255.255.255.0");
138     Ipv4InterfaceContainer ifcont = ipv4.Assign (devices);
139
140   NS_LOG_INFO ("Create Applications.");
141
142   // UDP connection from N1 to N3
143   for(int i = 1; i<titik; i++){
144
145     uint16_t sinkPort = titik;
146     Address sinkAddress (InetSocketAddress (ifcont.GetAddress (i),
147   sinkPort)); // interface of n3
148     PacketSinkHelper packetSinkHelper ("ns3::UdpSocketFactory",
149   InetSocketAddress (Ipv4Address::GetAny (), sinkPort));
150     ApplicationContainer sinkApps = packetSinkHelper.Install (c.Get
151   (i)); //n3 as sink
152     sinkApps.Start (Seconds (0.));
153     sinkApps.Stop (Seconds (100.));
154
155     Ptr<Socket> ns3UdpSocket = Socket::CreateSocket (c.Get (0),
156   UdpSocketFactory::GetTypeId ()); //source at n1
157
158     // Create UDP application at n1
159     Ptr<MyApp> app = CreateObject<MyApp> ();
160     app->Setup (ns3UdpSocket, sinkAddress, 64, 0, DataRate
161   ("2048bps"));//(ns3UdpSocket, sinkAddress, 64, 5, DataRate
162   ("2048bps"));
163     c.Get (i)->AddApplication (app);
164     app->SetStartTime (Seconds (0));//40
165     app->SetStopTime (Seconds (100.));
166   }
167
168   // Set Mobility for all nodes
169
170   MobilityHelper mobility;
171     Ptr<ListPositionAllocator> positionAlloc = CreateObject
172   <ListPositionAllocator>();
173     for(int i=0;i<titik;i++){
174       positionAlloc ->Add(Vector(i, 0, 0));
```

```
  }
  mobility.SetPositionAllocator(positionAlloc);

mobility.SetMobilityModel("ns3::ConstantPositionMobilityModel");
//ns3::ConstantPositionMobilityModel
  mobility.Install(c);

  AnimationInterface anim ("skripsiblackhole.xml"); // Mandatory

  // int setengah = titik/2;
  int kolom = 1;
  int baris = 1;
  for(int i=0;i<titik;i++){
    AnimationInterface::SetConstantPosition (c.Get (i),
kolom*100, baris*100);
    if((i+1)%10 == 0){
      kolom = 0;
      baris++;
    }
    kolom++;
  }

  if(attackenable==true){
    for(uint32_t n=0; n<c.GetN(); n++){
      if( n!=0 && n!=14 && n!=22){ //n!=1 && n!=2 && n!=6 &&
n!=14 && n!=15 && n!=22
        // anim.UpdateNodeDescription (c.Get (n),"Node");
        anim.UpdateNodeColor (c.Get (n), 255,0,0);
      }
      else if(n==0){
        anim.UpdateNodeDescription (c.Get (n),"Source");
        anim.UpdateNodeColor (c.Get (n), 0,255,0);
      }
      else{
        anim.UpdateNodeDescription (c.Get (n),"Blackhole");
        anim.UpdateNodeColor (c.Get (n), 0,0,0);
      }
    }
  }
  if(attackenable==false){
    for(uint32_t n=0;n<c.GetN(); n++){
      anim.UpdateNodeColor(c.Get(n),255,0,0);
    }
  }

  Ptr<OutputStreamWrapper> routingStream =
Create<OutputStreamWrapper> ("routes.blackhole", std::ios::out);
  aodv.PrintRoutingTableAllAt (Seconds (45), routingStream); //45

  // Trace Received Packets


Config::ConnectWithoutContext("/NodeList/*/ApplicationList/*/$ns3
::PacketSink/Rx", (MakeCallback (&ReceivePacket)));

  // Calculate Throughput using Flowmonitor

  FlowMonitorHelper flowmon;
  Ptr<FlowMonitor> monitor = flowmon.InstallAll();

```

```
325   // Now, do the actual simulation.
236
237   NS_LOG_INFO ("Run Simulation.");
238   Simulator::Stop (Seconds(100.0));
239   Simulator::Run ();
240
241   monitor->CheckForLostPackets ();
242
243   //modifikasi-----------------------------------------------------
244 ---
245   double th=0.0;
246   uint32_t LostPacketsum = 0;
247   uint32_t txPacketsum = 0;
248   uint32_t rxPacketsum = 0;
249   uint32_t DropPacketsum = 0;
250   double Delaysum =  0;
251   //----------------------------------------------
252   Ptr<Ipv4FlowClassifier> classifier =
253 DynamicCast<Ipv4FlowClassifier> (flowmon.GetClassifier ());
254   std::map<FlowId, FlowMonitor::FlowStats> stats = monitor-
255 >GetFlowStats ();
256   for (std::map<FlowId, FlowMonitor::FlowStats>::const_iterator i
257 = stats.begin (); i != stats.end (); ++i)
258     {
259     Ipv4FlowClassifier::FiveTuple t = classifier->FindFlow (i-
260 >first);
261       if ((t.sourceAddress=="10.1.2.1"))
262       {
263         //modifikasi-----------------------------------------------
264 -----------------------
265             txPacketsum += i->second.txPackets;
266                   rxPacketsum += i->second.rxPackets;
267                   LostPacketsum += i->second.lostPackets;
268                   DropPacketsum += i-
269 >second.packetsDropped.size();
270             Delaysum += i->second.delaySum.GetSeconds();
271         //-------------------------------------------------------
272 -----------------------
273           std::cout << "Flow " << i->first  << " (" <<
274 t.sourceAddress << " -> " << t.destinationAddress << ")\n";
275           std::cout << "  Tx Bytes:   " << i->second.txBytes <<
276 "\n";
277           std::cout << "  Rx Bytes:   " << i->second.rxBytes <<
278 "\n";
279           std::cout << "  Throughput: " << i->second.rxBytes *
280 8.0 / (i->second.timeLastRxPacket.GetSeconds() - i-
281 >second.timeFirstTxPacket.GetSeconds())/1024/1024  << " Mbps\n";
282         //modifikasi-----------------------------------------------
283 ------
284           std::cout << "  Delay:      " << i->second.delaySum <<
285 "\n"; //i->second.delaySum
286           std::cout << "  Packet Loss:" << i->second.lostPackets
287 << "\n";
288         //-------------------------------------------------------
289 ------
290
291   std::ofstream out (m_CSVfileName.c_str (), std::ios::app);
292   out <<malicious.GetN()<<","<<
293         titik <<","<<
294         txpower << ","<<
```

```
295         i->second.txBytes << ","<<
296         i->second.rxBytes <<","<<
297         i->second.lostPackets <<","<<
298         i->second.rxBytes * 8.0 / (i-
299 >second.timeLastRxPacket.GetSeconds() - i-
300 >second.timeFirstTxPacket.GetSeconds())/1024/1024  << ","<<
301         i->second.delaySum <<","<<
302         t.sourceAddress <<","<<
303         t.destinationAddress << ","<<
304 std::endl;
305 out.close ();
306        }
307       }
308
309   monitor->SerializeToXmlFile("lab-4.flowmon", true, true);
310
311   //resultgraph
312   std::ofstream ofs ("resultgraphbhcoba.plt",std::ofstream::out);
313
314   ofs << "set terminal png" << std::endl;
315   ofs << "set output 'resultgraphbh.png'" << std::endl;
316   ofs << "set title ''" << std::endl;
317   ofs << "set xlabel 'Nodes' " << std::endl;
318   ofs << "set ylabel 'Value' " << std::endl;
319   ofs << "plot "<<" '-'" <<"title "<<"'PDR'  with linespoints, '-
320 ' title 'overhead' with lines, '-' title 'Throughput' with lines,
321 '-' title 'Delay' with lines" << std::endl;
322   ofs << "1 " <<0<<std::endl;
323   ofs << titik <<" " <<((rxPacketsum*100) /
324 txPacketsum)/100.<<std::endl;
325   ofs << "e"<<std::endl;
326   ofs <<"1 " <<0<<std::endl;
327   ofs << titik <<" " <<((txPacketsum ) /
328 Delaysum)/10.0<<std::endl;
329   ofs << "e"<<std::endl;
330   ofs <<"1 " <<0<< std::endl;
331   ofs << titik <<" " << th*0.2<<std::endl;
332   ofs << "e"<<std::endl;
333   ofs << "1 " <<0<<std::endl;
334   ofs << titik <<" " <<Delaysum<<std::endl;
335   ofs << "e"<<std::endl;
336
337   ofs.close();
338   //-----------------------------------------------------------
339 }
```

2. Tabel hasil simulasi 4 node tanpa black hole

Packet Send	Packet Receive	Packet Loss	Throughput	Delay	Source	Destination
92	92	0	0,0243748	+28796332,0ns	10.1.2.1	10.1.2.2
92	92	0	0,0209303	+33535332,0ns	10.1.2.1	10.1.2.3
92	92	0	0,0186672	+37600999,0ns	10.1.2.1	10.1.2.4

3. Tabel hasil simulasi 20 node tanpa black hole

Packet Send	Packet Receive	Packet Loss	Throughput	Delay	Source	Destination
92	92	0	0,00420737	+166827250,0ns	10.1.2.1	10.1.2.2
92	92	0	0,000606783	+1156762581,0ns	10.1.2.1	10.1.2.3
92	92	0	0,0041053	+170975251,0ns	10.1.2.1	10.1.2.4
92	92	0	0,00402003	+174601587,0ns	10.1.2.1	10.1.2.5
92	92	0	0,000249733	+2810623000,0ns	10.1.2.1	10.1.2.6
92	92	0	0,0040634	+172738252,0ns	10.1.2.1	10.1.2.7
92	92	0	0,00398254	+176245255,0ns	10.1.2.1	10.1.2.8
92	92	0	0,000346306	+2026829879,0ns	10.1.2.1	10.1.2.9
92	92	0	0,00442193	+158732579,0ns	10.1.2.1	10.1.2.10
92	92	0	0,00426258	+164666584,0ns	10.1.2.1	10.1.2.11
92	92	0	0,000341598	+2054766873,0ns	10.1.2.1	10.1.2.12
92	92	0	0,000604951	+1160266101,0ns	10.1.2.1	10.1.2.13
92	92	0	0,000605787	+1158664302,0ns	10.1.2.1	10.1.2.14
92	92	0	0,00061009	+1150492941,0ns	10.1.2.1	10.1.2.15
92	92	0	0,000345551	+2031259647,0ns	10.1.2.1	10.1.2.16
92	92	0	0,000613026	+1144983301,0ns	10.1.2.1	10.1.2.17
92	92	0	0,000611059	+1148669208,0ns	10.1.2.1	10.1.2.18
92	92	0	0,000226825	+3094477321,0ns	10.1.2.1	10.1.2.19
92	92	0	0,000348534	+2013875140,0ns	10.1.2.1	10.1.2.20

4. Tabel hasil simulasi 50 node tanpa black hole

Packet Send	Packet Receive	Packet Loss	Throughput	Delay	Source	Destination
92	92	0	0,000570578	+1230164566,0ns	10.1.2.1	10.1.2.2
92	92	0	0,00406173	+172809343,0ns	10.1.2.1	10.1.2.3
92	92	0	0,00421189	+166648342,0ns	10.1.2.1	10.1.2.4
92	92	0	0,000249583	+2812308011,0ns	10.1.2.1	10.1.2.5
92	92	0	0,000249029	+2818561682,0ns	10.1.2.1	10.1.2.6
92	92	0	0,000248854	+2820545350,0ns	10.1.2.1	10.1.2.7
92	92	0	0,000249198	+2816657681,0ns	10.1.2.1	10.1.2.8
92	92	0	0,000214086	+3278601782,0ns	10.1.2.1	10.1.2.9
92	92	0	0,000249977	+2807878341,0ns	10.1.2.1	10.1.2.10

Packet Send	Packet Receive	Packet Loss	Throughput	Delay	Source	Destination
92	92	0	0,00056767	+1236464830,0ns	10.1.2.1	10.1.2.11
92	92	0	0,000577408	+1215611507,0ns	10.1.2.1	10.1.2.12
92	92	0	0,000574793	+1221142111,0ns	10.1.2.1	10.1.2.13
92	92	0	0,000568471	+1234723443,0ns	10.1.2.1	10.1.2.14
92	92	0	0,000184316	+3808162124,0ns	10.1.2.1	10.1.2.15
92	92	0	0,000183958	+3815560711,0ns	10.1.2.1	10.1.2.16
92	92	0	0,00018405	+3813656983,0ns	10.1.2.1	10.1.2.17
92	92	0	0,00021251	+3302920826,0ns	10.1.2.1	10.1.2.18
92	92	0	0,000555515	+1263520470,0ns	10.1.2.1	10.1.2.19
92	92	0	0,000565309	+1241628497,0ns	10.1.2.1	10.1.2.20
92	92	0	0,000582888	+1204182786,0ns	10.1.2.1	10.1.2.21
92	92	0	0,00014619	+4801316744,0ns	10.1.2.1	10.1.2.22
92	92	0	0,000145697	+4817570022,0ns	10.1.2.1	10.1.2.23
92	92	0	0,000146003	+4807469951,0ns	10.1.2.1	10.1.2.24
92	92	0	0,000165528	+4240407057,0ns	10.1.2.1	10.1.2.25
92	92	0	0,000145753	+4815727283,0ns	10.1.2.1	10.1.2.26
92	92	0	0,000165883	+4231331176,0ns	10.1.2.1	10.1.2.27
92	92	0	0,000145946	+4809353581,0ns	10.1.2.1	10.1.2.28
92	92	0	0,00021355	+3286831134,0ns	10.1.2.1	10.1.2.29
92	92	0	0,000165143	+4250272141,0ns	10.1.2.1	10.1.2.30
92	92	0	0,000166423	+4217586907,0ns	10.1.2.1	10.1.2.31
92	92	0	0,000165741	+4234936877,0ns	10.1.2.1	10.1.2.32
92	92	0	0,00016507	+4252156418,0ns	10.1.2.1	10.1.2.33
92	92	0	0,000213093	+3293889119,0ns	10.1.2.1	10.1.2.34
92	92	0	0,000165674	+4236659598,0ns	10.1.2.1	10.1.2.35
92	92	0	0,0001208	+5810443831,0ns	10.1.2.1	10.1.2.36
92	92	0	0,00021419	+3277016877,0ns	10.1.2.1	10.1.2.37
92	92	0	0,000164916	+4256142325,0ns	10.1.2.1	10.1.2.38
92	92	0	0,000120904	+5805486546,0ns	10.1.2.1	10.1.2.39
92	92	0	0,000166738	+4209629193,0ns	10.1.2.1	10.1.2.40
92	92	0	0,000212402	+3304604160,0ns	10.1.2.1	10.1.2.41

Packet Send	Packet Receive	Packet Loss	Throughput	Delay	Source	Destination
92	92	0	0,00014	+5013587625,0ns	10.1.2.1	10.1.2.42
92	92	0	0,000103071	+6809941074,0ns	10.1.2.1	10.1.2.43
92	92	0	0,000136535	+5140834766,0ns	10.1.2.1	10.1.2.44
92	92	0	0,000166104	+4225694735,0ns	10.1.2.1	10.1.2.45
92	92	0	0,000137816	+5093070230,0ns	10.1.2.1	10.1.2.46
92	92	0	0,000103042	+6811804970,0ns	10.1.2.1	10.1.2.47
92	92	0	0,000114412	+6134857673,0ns	10.1.2.1	10.1.2.48
92	92	0	0,000166813	+4207724528,0ns	10.1.2.1	10.1.2.49
92	92	0	8,93872E-05	+7852401596,0ns	10.1.2.1	10.1.2.50

5. Tabel hasil simulasi skenario pertama

Packet Send	Packet Receive	Packet Loss	Throughput	Delay	Source	Destination
92	92	0	0,0238778	+29395664,0ns	10.1.2.1	10.1.2.2
92	92	0	0,0149452	+46965331,0ns	10.1.2.1	10.1.2.3
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.4

6. Tabel hasil simulasi skenario kedua

Packet Send	Packet Receive	Packet Loss	Throughput	Delay	Source	Destination
92	92	0	0,0112779	+62236996,0ns	10.1.2.1	10.1.2.2
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.3
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.4

7. Tabel hasil simulasi skenario ketiga

Packet Send	Packet Receive	Packet Loss	Throughput	Delay	Source	Destination
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.2
92	92	0	0,00237094	+296045148,0ns	10.1.2.1	10.1.2.3
92	92	0	0,00228059	+307772643,0ns	10.1.2.1	10.1.2.4
92	92	0	0,00235454	+298107149,0ns	10.1.2.1	10.1.2.5
92	92	0	0,000336992	+2082851573,0ns	10.1.2.1	10.1.2.6
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.7
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.8

92	92	0	0,000249739	+2810553341,0ns	10.1.2.1	10.1.2.9
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.10
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.11
92	92	0	0,000593042	+1183565552,0ns	10.1.2.1	10.1.2.12
92	92	0	0,00062394	+1124955248,0ns	10.1.2.1	10.1.2.13
92	92	0	0,000603219	+1163596900,0ns	10.1.2.1	10.1.2.14
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.15
92	92	0	0,000183885	+3817081911,0ns	10.1.2.1	10.1.2.16
92	92	0	0,000183795	+3818945639,0ns	10.1.2.1	10.1.2.17
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.18
92	92	0	0,000184199	+3810566066,0ns	10.1.2.1	10.1.2.19
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.20

8. Tabel hasil simulasi skenario keempat

Packet Send	Packet Receive	Packet Loss	Throughput	Delay	Source	Destination
92	92	0	0,00333248	+210625192,0ns	10.1.2.1	10.1.2.2
92	92	0	0,00349429	+200871964,0ns	10.1.2.1	10.1.2.3
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.4
92	92	0	0,00340263	+206282686,0ns	10.1.2.1	10.1.2.5
92	92	0	0,000250571	+2801217666,0ns	10.1.2.1	10.1.2.6
92	92	0	0,00025043	+2802801334,0ns	10.1.2.1	10.1.2.7
92	92	0	0,000249756	+2810365009,0ns	10.1.2.1	10.1.2.8
92	92	0	0,000250102	+2806473339,0ns	10.1.2.1	10.1.2.9
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.10
92	92	0	0,00327604	+214253862,0ns	10.1.2.1	10.1.2.11
92	92	0	0,00054305	+1292522596,0ns	10.1.2.1	10.1.2.12
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.13
92	92	0	0,000184648	+3801317053,0ns	10.1.2.1	10.1.2.14
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.15
92	92	0	0,000583241	+1203455652,0ns	10.1.2.1	10.1.2.16
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.17
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.18
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.19

Packet Send	Packet Receive	Packet Loss	Throughput	Delay	Source	Destination
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.20

9. Tabel hasil simulasi skenario kelima

Packet Send	Packet Receive	Packet Loss	Throughput	Delay	Source	Destination
92	92	0	0,000250571	+2801216332,0ns	10.1.2.1	10.1.2.2
92	92	0	0,00025043	+2802797332,0ns	10.1.2.1	10.1.2.3
92	92	0	0,000249898	+2808765667,0ns	10.1.2.1	10.1.2.4
92	92	0	0,000250274	+2804539333,0ns	10.1.2.1	10.1.2.5
92	92	0	0,00293038	+239526682,0ns	10.1.2.1	10.1.2.6
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.7
92	92	0	0,000570295	+1230773589,0ns	10.1.2.1	10.1.2.8
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.9
92	92	0	0,000567787	+1236209742,0ns	10.1.2.1	10.1.2.10
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.11
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.12
92	92	0	0,000184648	+3801316744,0ns	10.1.2.1	10.1.2.13
92	92	0	0,000557199	+1259702011,0ns	10.1.2.1	10.1.2.14
92	92	0	0,000184561	+3803098864,0ns	10.1.2.1	10.1.2.15
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.16
92	92	0	0,000184276	+3808975039,0ns	10.1.2.1	10.1.2.17
92	92	0	0,000184178	+3811019425,0ns	10.1.2.1	10.1.2.18
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.19
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.20
92	92	0	0,00056566	+1240858488,0ns	10.1.2.1	10.1.2.21
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.22
92	92	0	0,000308904	+2272242843,0ns	10.1.2.1	10.1.2.23
92	92	0	0,000145983	+4808134041,0ns	10.1.2.1	10.1.2.24
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.25
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.26
92	92	0	0,000145933	+4809777352,0ns	10.1.2.1	10.1.2.27
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.28

Packet Send	Packet Receive	Packet Loss	Throughput	Delay	Source	Destination
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.29
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.30
92	92	0	0,00031161	+2252508011,0ns	10.1.2.1	10.1.2.31
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.32
92	92	0	0,000120861	+5807551897,0ns	10.1.2.1	10.1.2.33
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.34
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.35
92	92	0	0,000120828	+5809135043,0ns	10.1.2.1	10.1.2.36
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.37
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.38
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.39
92	92	0	0,000213317	+3290426077,0ns	10.1.2.1	10.1.2.40
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.41
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.42
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.43
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.44
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.45
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.46
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.47
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.48
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.49
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.50

10. Tabel hasil simulasi skenario keenam

Packet Send	Packet Receive	Packet Loss	Throughput	Delay	Source	Destination
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.2
92	92	0	0,000491626	+1427719790,0ns	10.1.2.1	10.1.2.3
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.4
92	92	0	0,00247493	+283605625,0ns	10.1.2.1	10.1.2.5
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.6
92	92	0	0,000249408	+2814278010,0ns	10.1.2.1	10.1.2.7

Packet Send	Packet Receive	Packet Loss	Throughput	Delay	Source	Destination
92	92	0	0,00215885	+325129327,0ns	10.1.2.1	10.1.2.8
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.9
92	92	0	0,000249752	+2810407005,0ns	10.1.2.1	10.1.2.10
92	92	0	0,000250571	+2801216332,0ns	10.1.2.1	10.1.2.11
92	92	0	0,000542309	+1294289321,0ns	10.1.2.1	10.1.2.12
92	92	0	0,000184648	+3801316744,0ns	10.1.2.1	10.1.2.13
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.14
92	92	0	0,000184566	+3802998864,0ns	10.1.2.1	10.1.2.15
92	92	0	0,000184487	+3804621939,0ns	10.1.2.1	10.1.2.16
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.17
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.18
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.19
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.20
92	92	0	0,000529723	+1325041333,0ns	10.1.2.1	10.1.2.21
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.22
92	92	0	0,000287418	+2442101612,0ns	10.1.2.1	10.1.2.23
92	92	0	0,00014619	+4801317201,0ns	10.1.2.1	10.1.2.24
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.25
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.26
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.27
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.28
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.29
92	92	0	0,000146056	+4805729628,0ns	10.1.2.1	10.1.2.30
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.31
92	92	0	0,000216807	+3237457582,0ns	10.1.2.1	10.1.2.32
92	92	0	0,000120991	+5801317201,0ns	10.1.2.1	10.1.2.33
92	92	0	0,000216534	+3241539727,0ns	10.1.2.1	10.1.2.34
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.35
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.36
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.37
92	92	0	0,000211659	+3316201069,0ns	10.1.2.1	10.1.2.38

Packet Send	Packet Receive	Packet Loss	Throughput	Delay	Source	Destination
92	92	0	0,000211762	+3314595680,0ns	10.1.2.1	10.1.2.39
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.40
92	92	0	0,000211356	+3320959616,0ns	10.1.2.1	10.1.2.41
92	92	0	0,000103201	+6801317374,0ns	10.1.2.1	10.1.2.42
92	92	0	0,000167814	+4182622398,0ns	10.1.2.1	10.1.2.43
92	92	0	0,00010317	+6803380416,0ns	10.1.2.1	10.1.2.44
92	92	0	0,000167543	+4189391853,0ns	10.1.2.1	10.1.2.45
92	92	0	0,000103107	+6807510488,0ns	10.1.2.1	10.1.2.46
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.47
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.48
92	0	1	0	+0,0ns	10.1.2.1	10.1.2.49
92	92	0	0,000165036	+4253036072,0ns	10.1.2.1	10.1.2.50