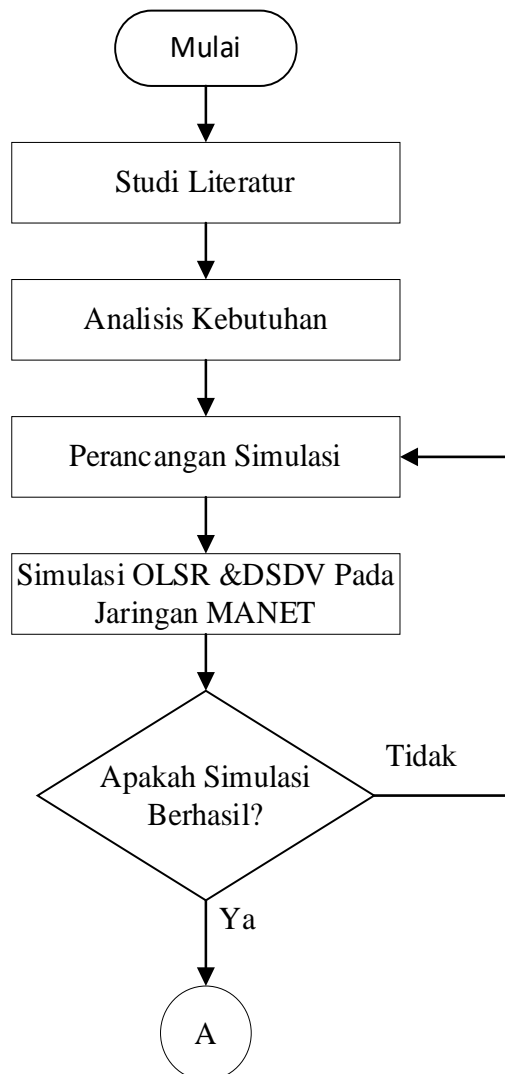
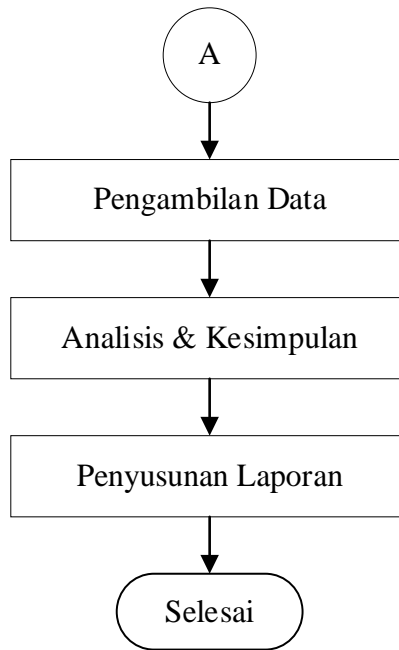


BAB 3 METODE PENELITIAN

3.1 ALUR PENELITIAN

Gambar 3.1 menggambarkan alur penelitian yang dilakukan dalam penelitian ini, dimulai dengan studi literatur dan diakhiri dengan perancangan sistem.





Gambar 3.1 Diagram alur penelitian

Penelitian ini diawali dengan melakukan studi literatur terhadap beberapa penelitian terkait MANET yang menggunakan protokol *routing* OLSR dan DSDV dengan pergerakan *node*.

Dengan membandingkan jurnal terkait dapat digunakan untuk membantu peneliti untuk mempersempit fokus penelitian dan menentukan judul yang tepat. serta dapat memahami konsep dasar dari topik tersebut.

Selanjutnya adalah menganalisis kebutuhan *hardware* dan *software* yang digunakan dan dibutuhkan pada penelitian ini.

Setelah melakukan analisis *hardware* dan *software* yang digunakan dalam penelitian ini. Langkah selanjutnya merupakan perancangan simulasi jaringan untuk penelitian ini. Dalam perancangan simulasi ini peneliti melakukan instalasi aplikasi *network simulator 3* untuk *software* yang akan digunakan pada penelitian ini. Setelah instalasi berhasil dilakukan peneliti melakukan perancangan parameter jaringan.

Setelah perancangan parameter jaringan, langkah selanjutnya adalah melakukan perancangan topologi dengan jumlah *node* pada topologi untuk penelitian ini adalah 20, 30, 40 dan 50 *node*, serta pergerakan *node* yang akan digunakan pada perancangan topologi simulasi merupakan pergerakan *node*

random walk. Lalu setelah perancangan simulasi berhasil dilakukan. Selanjutnya, adalah menjalankan simulasi yang sudah dibuat dan dilakukan pengujian.

Ketika pengujian berhasil dijalankan tanpa ada masalah maka selanjutnya merupakan pengambilan data. Data yang akan diambil merupakan data QOS meliputi PDR, *throughput*, *delay* dan *packet loss*. Setelah semua data terkumpul dan berhasil diambil, langkah selanjutnya adalah menganalisis data tersebut.

Analisis dilakukan dengan cara membandingkan data yang diperoleh saat menggunakan *routing* protokol OLSR dan DSDV menggunakan pergerakan *node random walk* dengan variasi jumlah node yang sudah ditentukan. yang ditampilkan dalam bentuk grafik, kesimpulan akan dilakukan ketika analisis sudah selesai dilakukan. Analisa dan kesimpulan dilakukan dengan menganalisis data QOS meliputi PDR, *throughput*, *delay* dan *packet loss*.

3.1.1 STUDI LITERATUR

Studi literatur merupakan tahap fundamental dalam proses penelitian, di mana peneliti mencari, mengumpulkan, membaca, dan memahami teori-teori yang berkaitan dengan topik penelitian. Sumber-sumber teori yang digunakan dapat berupa jurnal, buku, dan situs web yang relevan. Tujuan dari fase studi literatur adalah untuk membantu mengidentifikasi permasalahan penelitian, merumuskan pertanyaan penelitian, menentukan metode pengumpulan data, serta meningkatkan pemahaman tentang topik penelitian yang sedang dibahas. Dengan demikian, tahap studi literatur memegang peran yang sangat penting dalam mendukung kelancaran jalannya penelitian.

3.1.2 Analisis Kebutuhan *Hardware* dan *Software*

3.1.2.1 *Hardware*

Hardware yang digunakan untuk menjalankan simulasi pada penelitian ini menggunakan satu unit laptop dengan *spesifikasi* sebagai berikut:

1. *Processor* : AMD A4-9125 (2.3 GHz; up to 2.6 GHz; 1 MB cache)
2. *RAM* : 4GB DDR4
3. *SSD* : 256 GB SATA

3.1.2.2 Software

Software yang digunakan untuk menjalankan simulasi pada penelitian ini sebagai berikut:

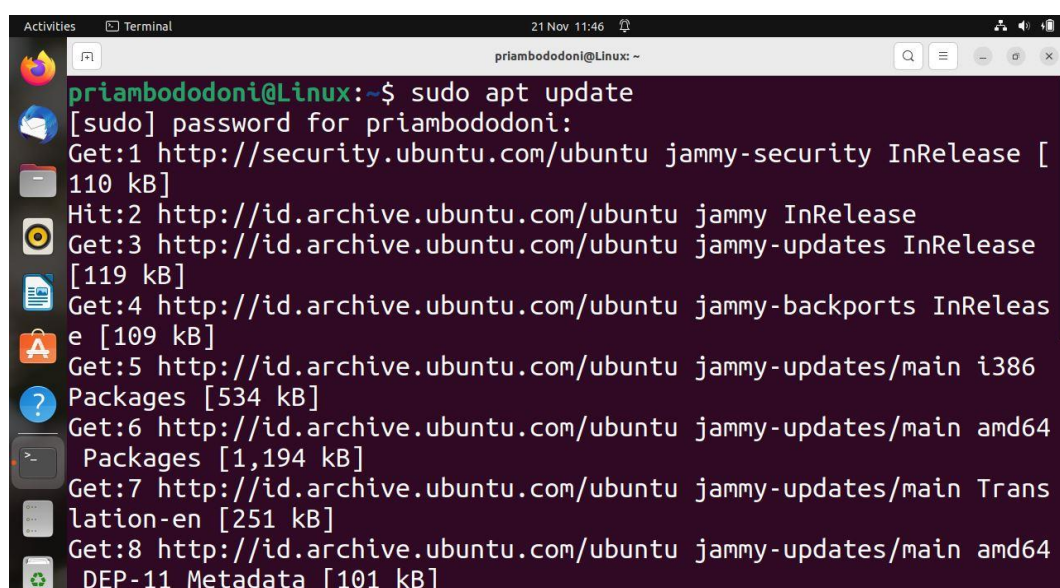
1. *Virtual Machine* : Oracle VM VirtualBox 7.0.10
2. *Sistem Operasi* : Ubuntu-22.04.3-Desktop-AMD64
3. *Simulation Tools* : Network Simulator 3.37 (NS3)

3.1.3 Perancangan Simulasi dan Jaringan

Penelitian ini menggunakan software NS-3.37 untuk melakukan perancangan jaringannya.

3.1.3.1 Instalasi Network Simulator 3

Penelitian ini akan menggunakan software network simulator 3 dengan versi NS3.37. Sebelum instalasi diperlukan untuk mengunduh software-nya pada website nsnam dan ekstrak file unduhan software dan letakkan pada folder home di ubuntu. Untuk melakukan instalasi kita perlu men-setting beberapa command untuk update pada terminal ubuntu. Instalasi software perlu beberapa langkah yang perlu diperhatikan agar program dapat menjalankan simulasi. Berikut tahapan-tahapan untuk menginstal NS3.37:



```
priambododoni@Linux:~$ sudo apt update
[sudo] password for priambododoni:
Get:1 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Hit:2 http://id.archive.ubuntu.com/ubuntu jammy InRelease
Get:3 http://id.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:4 http://id.archive.ubuntu.com/ubuntu jammy-backports InRelease [109 kB]
Get:5 http://id.archive.ubuntu.com/ubuntu jammy-updates/main i386 Packages [534 kB]
Get:6 http://id.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [1,194 kB]
Get:7 http://id.archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [251 kB]
Get:8 http://id.archive.ubuntu.com/ubuntu jammy-updates/main amd64 DEP-11 Metadata [101 kB]
```

Gambar 3.2 Instalasi tahap pertama

Pada gambar 3.2 merupakan gambar langkah pertama untuk instalasi ns3 yaitu memberikan perintah *"sudo apt update"*. Dimana perintah *"sudo apt update"* pada *ubuntu* ini digunakan untuk memperbarui daftar paket yang tersedia dari repository yang terkonfigurasi di system. Dalam instalasi NS-3 perintah ini membantu memastikan bahwa system memiliki informasi paket yang paling baru sebelum melakukan instalasi.

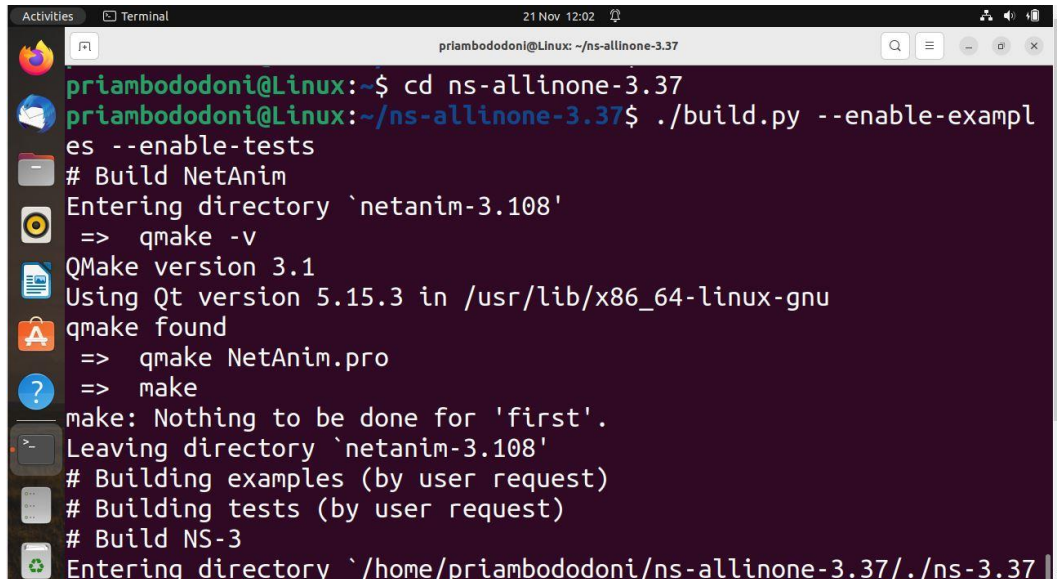
```

priambododoni@Linux:~$ sudo apt install build-essential autoconf l
libxmu-dev g++ python3 python3-dev pkg-config sqlite3 cmake python3
-setuptools git qtbase5-dev qtchooser qt5-qmake qtbase5-dev-tools
gir1.2-gocanvas-2.0 python3-gi python3-gi-cairo python3-pygraphviz
gir1.2-gtk-3.0 ipython3 openmpi-bin openmpi-common openmpi-doc l
ibopenmpi-dev autoconf cvs bzip2 unrar gsl-bin libgsl-dev libgslcblas
0 wireshark tcpdump sqlite sqlite3 libsqlite3-dev llvm-dev automa
ke python3-pip libxml2 libxml2-dev libboost-all-dev
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
autoconf is already the newest version (2.71-2).
automake is already the newest version (1:1.16.5-1.3).
build-essential is already the newest version (12.9ubuntu3).
g++ is already the newest version (4:11.2.0-1ubuntu1).
libxmu-dev is already the newest version (2:1.1.3-3).
pkg-config is already the newest version (0.29.2-1ubuntu3).

```

Gambar 3.3 Instalasi tahap kedua

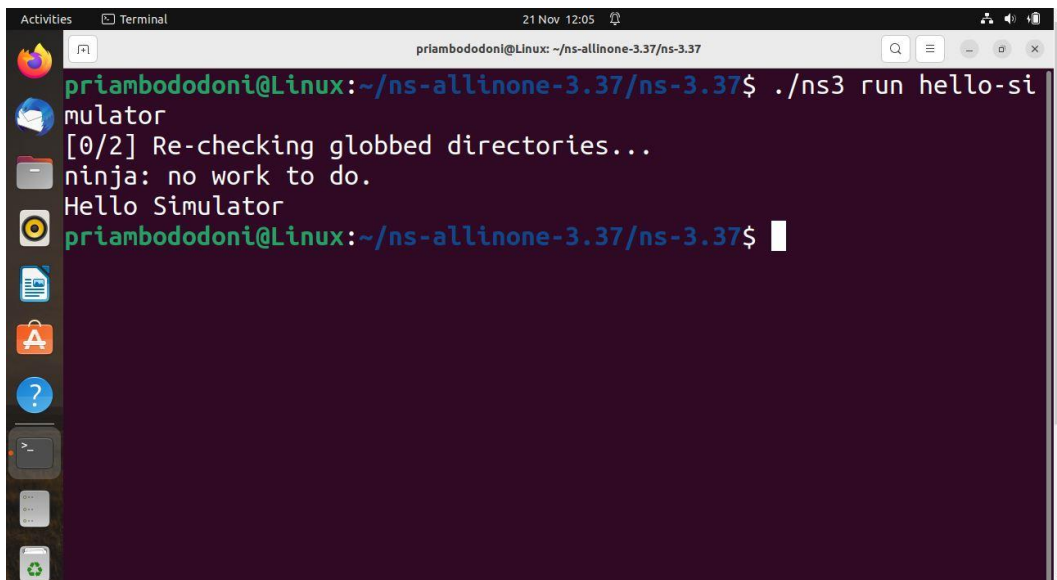
Pada gambar 3.3 merupakan gambar instalasi langkah ke dua dimana memberi perintah pada terminal *ubuntu* yaitu *"sudo apt install build-essential autoconf automake libxmu-dev g++ python3 python3-dev pkg-config sqlite3 cmake python3-setuptools git qtbase5-dev qtchooser qt5-qmake qtbase5-dev-tools gir1.2-gocanvas-2.0 python3-gi python3-gi-cairo python3-pygraphviz gir1.2-gtk-3.0 ipython3 openmpi-bin openmpi-common openmpi-doc libopenmpi-dev autoconf cvs bzip2 unrar gsl-bin libgsl-dev libgslcblas0 wireshark tcpdump sqlite sqlite3 libsqlite3-dev libxml2 libxml2-dev libboost-all-dev"*. Fungsi dari *command* adalah untuk memberikan daftar paket yang perlu di *instal* dimana paket - paket tersebut diperlukan agar bisa menjalankan program pada *network simulator 3*. Setelah selesai *command* dijalankan paket – paket yang diperlukan *software network simulator 3* untuk menjalankan program secara otomatis terinstal dan bisa menjalankan program simulasi.



```
priambododoni@Linux: ~$ cd ns-allinone-3.37
priambododoni@Linux: ~/ns-allinone-3.37$ ./build.py --enable-examples --enable-tests
# Build NetAnim
Entering directory `netanim-3.108'
=> qmake -v
QMake version 3.1
Using Qt version 5.15.3 in /usr/lib/x86_64-linux-gnu
qmake found
=> qmake NetAnim.pro
=> make
make: Nothing to be done for 'first'.
Leaving directory `netanim-3.108'
# Building examples (by user request)
# Building tests (by user request)
# Build NS-3
Entering directory `~/home/priambododoni/ns-allinone-3.37/./ns-3.37'
```

Gambar 3.4 Instalasi tahap ketiga

Pada gambar 3.4 adalah gambar instalasi langkah ketiga dimana perlu masuk ke folder NS3 yang ada pada folder *Home* dengan menjalankan perintah “*cd ns-allinone-3.37*”. Setelah masuk, perintah selanjutnya adalah “*./build.py --enable-examples --enable-tests*” untuk menginstal folder - folder pada ns3.37.



```
priambododoni@Linux: ~/ns-allinone-3.37/ns-3.37$ ./ns3 run hello-simulator
[0/2] Re-checking globbed directories...
ninja: no work to do.
Hello Simulator
priambododoni@Linux: ~/ns-allinone-3.37/ns-3.37$
```

Gambar 3.5 Instalasi tahap keempat

Pada gambar 3.5 adalah gambar instalasi langkah ke empat dimana perlu masuk ke folder NS3.37 dengan perintah “*cd ns-3.37*”. selanjutnya untuk mencoba apakah instalasi sudah berhasil atau belum. Maka perlu menjalankan perintah “*./ns3 -run hello-simulator*” untuk menjalankan *simulator* bernama *hello*

simulator. Jika muncul “*Hello Simulator*” maka artinya instalasi yang dilakukan sudah berhasil. dan aplikasi sudah bisa digunakan untuk menjalankan program simulasi.

3.1.3.2 Script dan penjelasan simulasi

1. Fungsi utama

Gambar 3.6 merupakan fungsi utama untuk menjalankan kelas eksperimen, pembuatan *file csv* dan *command* untuk menjalankan simulasi program. Penamaan kelas eksperimen pada *script* dengan nama *RoutingExperiment*. Pembuatan *file csv* bertujuan untuk menghasilkan *file output .csv*.

```
int
main (int argc, char *argv[])
{
    RoutingExperiment experiment;
    std::string CSVfileName = experiment.CommandSetup (argc, argv);
    std::ofstream out (CSVfileName.c_str ());
    out << "SimulationSecond,"
    <<"ReceiveRate," <<
    "PacketsReceived," <<
    "NumberOfSinks," <<
    "RoutingProtocol," <<
    "TransmissionPower"
    <<
    std::endl;
    out.close ();
    int nSinks = 10;
    double txp = 15;
    experiment.Run (nSinks, txp, CSVfileName);
}
```

Gambar 3.6 Script fungsi utama

2. Inisialisasi kelas utama *routingexperiment*

Gambar 3.7 merupakan *script* untuk kelas utama yang berisi *variabel* utama dan fungsi-fungsi yang dijalankan saat simulasi berjalan. Pada *script* dituliskan penamaan *file csv* dengan nama “*OLSR_DSDV.csv*”.

```
class RoutingExperiment
{
public:
    RoutingExperiment ();
    void Run (int nSinks, double txp, std::string CSVfileName);
    std::string CommandSetup (int argc, char **argv);
private:Ptr<Socket> SetupPacketReceive (Ipv4Address addr,
Ptr<Node> node);
    void ReceivePacket (Ptr<Socket> socket);
    void CheckThroughput ();
    uint32_t port;
    uint32_t bytesTotal;
    uint32_t packetsReceived;
    std::string m_CSVfileName;
    int m_nSinks;
    std::string m_protocolName;
    double m_txp;
    uint32_t m_protocol;
};
RoutingExperiment::RoutingExperiment ()
: port (9),
  bytesTotal (0),
  packetsReceived (0),
  m_CSVfileName ("OLSR_DSDV.csv"),
  m_protocol (0) // Default to OLSR
{}
```

Gambar 3.7 Script kelas utama

3. Receive packet

Gambar 3.8 merupakan *script* untuk mencetak paket yang terkirim saat simulasi dijalankan.

```
static inline std::string
PrintReceivedPacket (Ptr<Socket> socket, Ptr<Packet> packet,
Address senderAddress)
{
std::ostringstream oss;
oss << Simulator::Now ().GetSeconds () << " " << socket->GetNode
()->GetId ();
if (InetSocketAddress::IsMatchingType (senderAddress))
{
InetSocketAddress addr = InetSocketAddress::ConvertFrom
(senderAddress);
oss << " received one packet from " << addr.GetIpv4 ();
}
else
{
oss << " received one packet!";
}
return oss.str ();
}
```

Gambar 3.8 Script receive packet

4. Set node

Gambar 3.9 merupakan *script* untuk menentukan *node* awal dan menentukan kapan *node* akan mulai mengirimkan paket dan berhenti mengirimkan paket.

```
for (int i = 0; i < nSinks; i++)
{
```

```

Ptr<Socket> sink = SetupPacketReceive (adhocInterfaces.GetAddress
(i), adhocNodes.Get (i));
AddressValue remoteAddress (InetSocketAddress
(adhocInterfaces.GetAddress (i), port));
onoff1.SetAttribute ("Remote", remoteAddress);
Ptr<UniformRandomVariabel> var =
CreateObject<UniformRandomVariabel> ();
ApplicationContainer temp = onoff1.Install (adhocNodes.Get (i +
nSinks));
temp.Start (Seconds (var->GetValue (100.0, 101.0)));
temp.Stop (Seconds (TotalTime));
}

```

Gambar 3.9 Script Set Node

5. Pembuatan *node*

Gambar 3.10 merupakan *script* untuk membuat *node* terlebih dahulu sesuai konsep NS3.

```

NodeContainer adhocNodes;
adhocNodes.Create (nWifis);

```

Gambar 3.10 Script pembuatan *node*

6. Set *IP Address*

Gambar 3.11 berfungsi untuk membuat range *ip address* yang akan diberikan pada setiap *node*.

```

Ipv4AddressHelper addressAdhoc;
addressAdhoc.SetBase ("10.1.1.0", "255.255.255.0");
Ipv4InterfaceContainer adhocInterfaces; adhocInterfaces =
addressAdhoc.Assign (adhocDevices);

```

Gambar 3.11 Script *IP address*

7. *Flowmonitor*

Gambar 3.12 merupakan *script* untuk melakukan perhitungan untuk mendapatkan nilai *delay* dan *packet loss*, jumlah paket dikirim dan

diterima, dan juga *packet loss* menggunakan module *flowmonitor* yang ada pada NS3, hasil dari program tersebut disimpan dalam file *.flowmon*.

```
Ptr<Flowmonitor> flowmon;  
FlowmonitorHelper flowmonHelper;  
flowmon = flowmonHelper.InstallAll();  
flowmon->SerializeToXmlFile ((tr_name + ".flowmon").c_str (), false,  
false);
```

Gambar 3.12 Script flowmonitor

8. *AnimationInterface*

Gambar 3.13 merupakan *script* untuk menampilkan topologi *node* pada jaringan manet yang digunakan. *Script* tersebut digunakan untuk mendapatkan hasil *file output* bertipe *.xml* dimana *file* tersebut bisa dijalankan menggunakan fitur *NetAnim* yang berada pada *software network simulator 3*.

```
AnimationInterface anim("manet-routing-compare.xml");
```

Gambar 3.13 Script AnimationInterface

9. Simulasi dimulai dan berhenti

Gambar 3.14 merupakan *script* untuk memulai simulasi dan simulasi berhenti.

```
Simulator::Stop (Seconds (TotalTime));  
Simulator::Run ();  
AnimationInterface anim("manet-routing-compare.xml");  
...  
Simulator::Destroy ();
```

Gambar 3.14 Script simulasi dimulai dan berhenti

3.1.3.3 Parameter jaringan

Peneliti menggunakan parameter jaringan MANET yang akan digunakan untuk melakukan pengujian sistem yang akan dijalankan pada *software network*

simulator 3.37. sistem akan dirancang menggunakan parameter – parameter yang sudah ditentukan sesuai seperti pada tabel 3.1.

Tabel 3.1 Parameter jaringan

Parameter	Nilai
Protokol	OLSR dan DSDV
Jumlah <i>Node</i>	20, 30, 40, dan 50
Transport Protokol	UDP
Kecepatan Transmisi	11Mbps
Paket Data	64 <i>byte</i>
Pergerakan <i>Node</i>	<i>Random walk</i>
Luas Area	500x500
Waktu Simulasi	200s

Penelitian akan menggunakan *software network simulator 3.37* dengan sistem operasi *ubuntu 22.04.3*. pada jaringan MANET ini *routing* protokol yang digunakan pada penelitian ini akan menggunakan *routing* protokol *proactive* yaitu OLSR dan DSDV. Dengan variasi jumlah *node* yaitu 20, 30, 40, dan 50 *node*. Menggunakan *transport protocol* jenis *User Datagram Protocol (UDP)*. Kecepatan transmisi data sebesar 11Mbps yang digunakan untuk mentransmisikan data secara nirkabel menggunakan standar *Wi-Fi 802.11b*. Simulasi menggunakan pergerakan *node* yaitu *random walk*. Dengan ukuran paket data sebesar 64*byte*, dengan luas area yang digunakan sebesar 500x500 meter dan waktu simulasi yang digunakan selama 200s.

1. Konfigurasi Parameter

Gambar 3.15 merupakan *script* untuk beberapa *konfigurasi* seperti jumlah *node* yang digunakan. Lalu konfigurasi kecepatan *transmisi* data sebesar 11Mbps yang digunakan untuk mentransmisikan data secara nirkabel menggunakan standar *Wi-Fi 802.11b*. Kemudian konfigurasi ukuran paket data sebesar 64 *byte*, waktu simulasi selama 200s, kemudian kecepatan *node* sebesar 20 m/s, tidak ada waktu *pause* pada *node* yang digunakan, dan *data rate* sebesar 2084 bps. Pengaturan penggunaan jumlah *node* yang digunakan dengan mengubah nilai *nWifis*.

```

RoutingExperiment::Run (int nSinks, double txp, std::string CSVfileName)
{
Packet::EnablePrinting ();
m_nSinks = nSinks;
m_txp = txp;
m_CSVfileName = CSVfileName;
int nWifis = 20;
double TotalTime = 200.0;
std::string rate ("2048bps");
std::string phyMode ("DsssRate11Mbps");
std::string tr_name ("OLSR_DSDV");
int nodeSpeed = 20; // in m/s
int nodePause = 0; // in s
Config::SetDefault ("ns3::OnOffApplication::PacketSize", StringValue
("64"));
Config::SetDefault ("ns3::OnOffApplication::DataRate", StringValue
(rate));

```

Gambar 3.15 Script konfigurasi parameter

2. Konfigurasi *Routing Protocol*

Gambar 3.16 adalah *script* untuk mengkonfigurasi *routing* protokol yang akan digunakan yaitu OLSR dan DSDV.

```

Ipv4ListRoutingHelper list;
InternetStackHelper internet;
OlsrHelper olsr;
DsdvHelper dsdv;
switch (m_protocol)
{
case 0:
list.Add (olsr, 100);
m_protocolName = "OLSR";
break;
case 1:
list.Add (dsdv, 100);
m_protocolName = "DSDV";
break;

```

```
default:
NS_FATAL_ERROR ("No such protocol: " << m_protocol);
}
```

Gambar 3.16 Script konfigurasi *routing* protokol

3. Konfigurasi Luas Area

Gambar 3.17 merupakan *script* untuk mengkonfigurasi luas area yang digunakan adalah 500x500 meter.

```
ObjectFaktry pos;
pos.SetTypeId ("ns3::RandomRectanglePositionAllocator");
pos.Set ("X", StringValue
("ns3::UniformRandomVariabel[Min=0.0|Max=500.0]"));
pos.Set ("Y", StringValue
("ns3::UniformRandomVariabel[Min=0.0|Max=500.0]"));
```

Gambar 3.17 Script konfigurasi luas area

4. Konfigurasi Pergerakan *Node*

Gambar 3.18 merupakan *script* untuk mengkonfigurasi jenis pergerakan *node* yang digunakan yaitu pergerakan *random walk*.

```
mobilityAdhoc.SetMobilityModel("ns3::RandomWalk2dMobilityModel",
"Mode", StringValue("Time"),
"Time", StringValue("1s"),
"Speed", StringValue(ssSpeed.str()),
"Bounds", StringValue("0|500|0|500"));
mobilityAdhoc.SetPositionAllocator (taPositionAlloc);
mobilityAdhoc.Install (adhocNodes);
```

Gambar 3.18 Script konfigurasi *random walk*

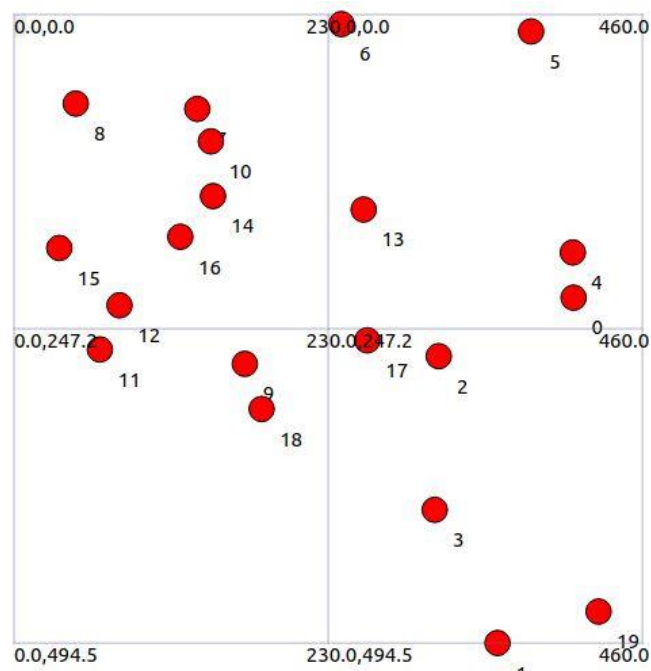
3.1.3.4 Perancangan Topologi

Dalam perancangan topologi dalam penelitian ini, menggunakan skenario dengan jumlah *node* sebanyak 20, 30, 40, dan 50. Serta *routing* protokol yang akan digunakan adalah OLSR dan DSDV. Dengan pergerakan *node* yang akan

digunakan adalah pergerakan *random walk*. Proses perancangan topologi ini akan dilakukan dengan memanfaatkan perangkat *NetAnim* yang tersedia dalam perangkat lunak *network simulator* versi 3.37 yang akan menampilkan tampilan dari topologi yang sudah dibuat. Berikut adalah gambaran perancangan topologi dengan jumlah *node* 20, 30, 40, dan 50:

1. Skenario menggunakan 20 *node*

Gambar 3.19 adalah topologi jaringan MANET menggunakan 20 *node* dengan protokol *routing* OLSR dan DSDV dengan pergerakan *random walk* menggunakan perangkat *NetAnim* pada NS3. Lingkaran merah pada gambar merupakan *node* yang digunakan pada topologi jaringan MANET. Angka – angka pada masing – masing sudut menunjukkan luas area yang digunakan yaitu 500x500. *File* yang digunakan dalam ns3 untuk menunjukkan gambar topologi jaringan yang digunakan merupakan *file .xml* yang dijalankan menggunakan aplikasi *NetAnim* yang tersedia pada *software ns3*.

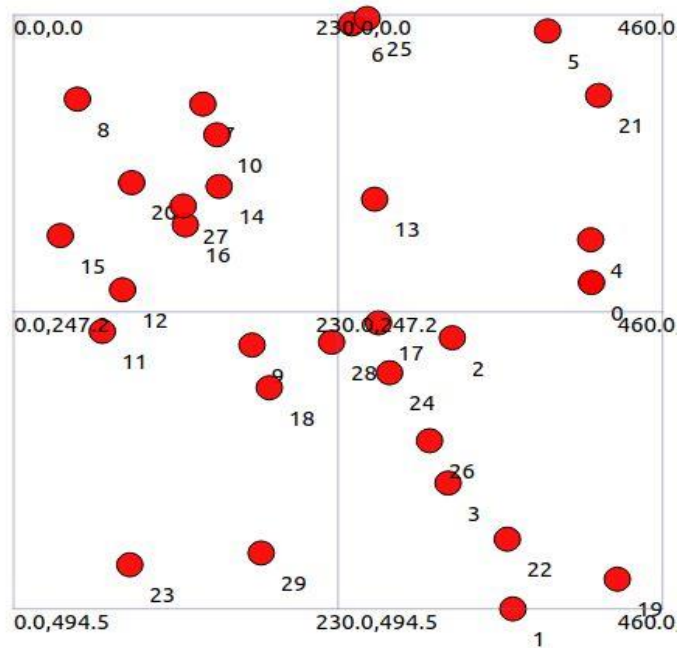


Gambar 3.19 Topologi 20 *node*

2. Skenario menggunakan 30 *node*

Gambar 3.20 adalah topologi jaringan MANET menggunakan 30 *node* dengan protokol *routing* OLSR dan DSDV dengan pergerakan *random walk*

menggunakan perangkat *NetAnim* pada NS3. Lingkaran merah pada gambar merupakan *node* yang digunakan pada topologi jaringan MANET. Angka – angka pada masing – masing sudut menunjukkan luas area yang digunakan yaitu 500x500. Skenario 30 *node* menunjukkan kepadatan pada topologi jaringan. *File* yang digunakan dalam NS3 untuk menunjukkan gambar topologi jaringan yang digunakan merupakan *file .xml* yang dijalankan menggunakan aplikasi *NetAnim* yang tersedia pada *software ns3*.

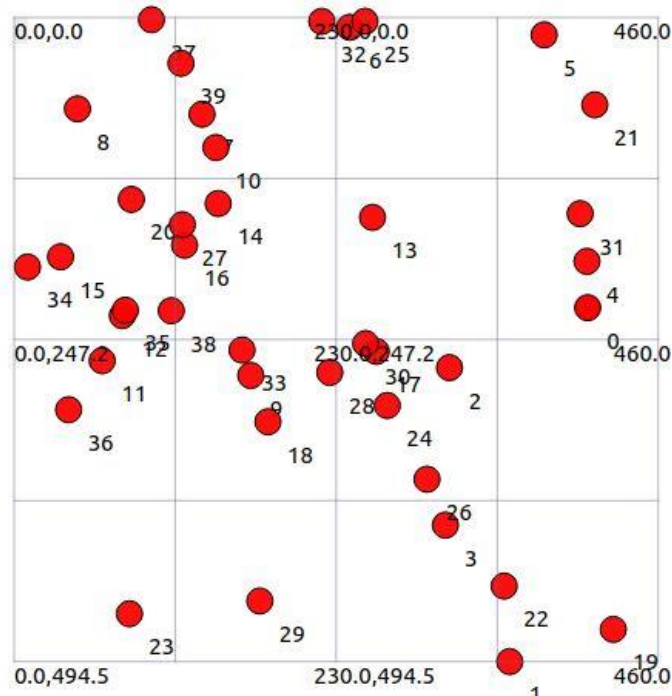


Gambar 3.20 Topologi 30 *node*

3. Skenario menggunakan 40 *node*

Gambar 3.21 adalah topologi jaringan MANET menggunakan 40 *node* dengan protokol *routing* OLSR dan DSDV dengan pergerakan *random walk* menggunakan perangkat *NetAnim* pada NS3. Lingkaran merah pada gambar merupakan *node* yang digunakan pada topologi jaringan MANET. Angka – angka pada masing – masing sudut menunjukkan luas area yang digunakan yaitu 500x500. *File* yang digunakan dalam ns3 untuk menunjukkan gambar topologi jaringan yang digunakan merupakan *file .xml* yang dijalankan menggunakan aplikasi *NetAnim* yang tersedia pada *software ns3*. Topologi jaringan pada skenario 40 *node* menunjukkan kepadatan pada topologi jaringan. Kepadatan terjadi karena

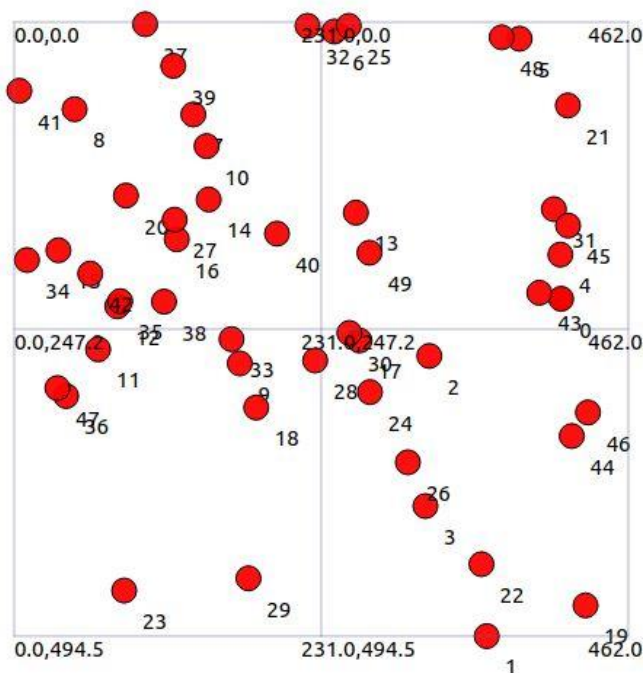
penambahan *node* dan luas area yang digunakan masih sama dengan topologi jaringan sebelumnya.



Gambar 3.21 Topologi 40 node

4. Skenario menggunakan 50 node

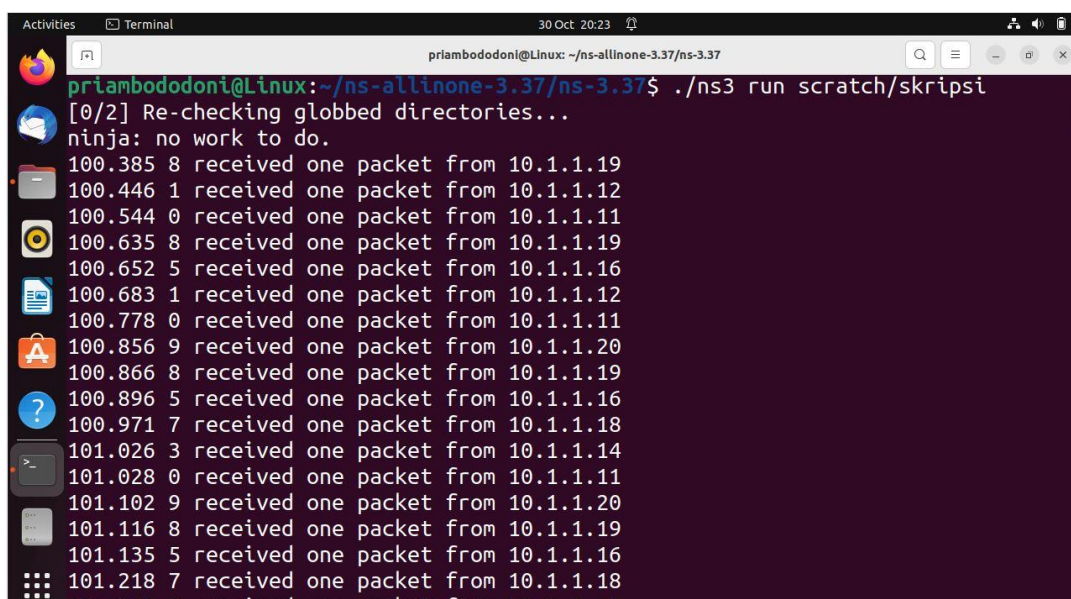
Gambar 3.22 merupakan topologi jaringan MANET menggunakan 50 *node* dengan protokol *routing* OLSR dan DSDV dengan pergerakan *random walk* menggunakan perangkat *NetAnim* pada NS3. Lingkaran merah pada gambar merupakan *node* yang digunakan pada topologi jaringan MANET. Angka – angka pada masing – masing sudut menunjukkan luas area yang digunakan yaitu 500x500. *File* yang digunakan dalam ns3 untuk menunjukkan gambar topologi jaringan yang digunakan merupakan *file .xml* yang dijalankan menggunakan aplikasi *NetAnim* yang tersedia pada *software* NS3. Topologi jaringan pada skenario 50 *node* menunjukkan kepadatan pada topologi jaringan. Kepadatan terjadi karena penambahan *node* dan luas area yang digunakan masih sama dengan topologi jaringan sebelumnya. Posisi *node* pada topologi jaringan manet ini terlihat sangat acak karena pergerakan *node* yang digunakan dimana pada topologi jaringan manet ini menggunakan pergerakan *random walk* dimana posisi *node* pada *random walk* bergerak secara acak dan posisi *node* dapat berubah selama simulasi.



Gambar 3.22 Topologi 50 node

3.1.3.5 Running Simulasi

Simulasi dijalankan menggunakan *network simulator 3* dengan versi NS3.37. Setelah menentukan parameter jaringan. Lalu, menentukan skenario pengujian dan perancangan topologi. Selanjutnya adalah menjalankan simulasi. Simulasi dilakukan dengan menggunakan terminal pada *ubuntu*.



Gambar 3.23 Menjalankan simulasi

Gambar 3.23 merupakan gambar saat menjalankan simulasi. Menjalankan simulasi dilakukan dengan memberikan perintah `./ns3 run scratch/skripsi`. *Scratch* merupakan folder yang digunakan untuk menjalankan program pada NS3. Sedangkan, skripsi merupakan nama dari program yang akan dijalankan.

3.1.4 Pengambilan Data

Setelah melakukan simulasi dengan beberapa parameter, pergerakan *node* dan *routing* protokol yang sudah ditentukan. Ada beberapa data yang akan diambil antara lain data simulasi *routing* protokol OLSR dan DSDV dengan jumlah *node* 20,30,40, dan 50. Dengan pergerakan yang digunakan merupakan *random walk*. Data yang diambil akan menggunakan parameter QoS antara lain PDR, *throughput*, *delay*, dan *packet loss*.

Hasil yang didapat pada saat setelah melakukan simulasi pada NS3, nantinya akan menghasilkan 3 *file output* antara lain *file .xml*, *file .flowmon*, dan *file .csv*. Tiga *file* tersebut akan digunakan untuk mendapatkan data QoS antara lain PDR, *throughput*, *delay*, dan *packet loss*.

1. File .csv

File ini merupakan salah satu *file* yang didapatkan dari simulasi yang dijalankan. *File* ini menampilkan beberapa data yang dapat digunakan dalam mengumpulkan data QoS khususnya nilai *throughput*. Pada gambar 3.24 adalah sebuah *file* dalam format *.csv* yang berisi informasi terkait parameter - parameter yang digunakan dalam simulasi.

SimulationSecond	ReceiveRate	PacketsReceived	NumberOfSinks	RoutingProtocol	TransmissionPower
0	0	0	10	OLSR	15
1	0	0	10	OLSR	15
2	0	0	10	OLSR	15
3	0	0	10	OLSR	15
4	0	0	10	OLSR	15
5	0	0	10	OLSR	15
6	0	0	10	OLSR	15
7	0	0	10	OLSR	15
8	0	0	10	OLSR	15
9	0	0	10	OLSR	15
10	0	0	10	OLSR	15
11	0	0	10	OLSR	15
12	0	0	10	OLSR	15
13	0	0	10	OLSR	15
14	0	0	10	OLSR	15

Gambar 3.24 File .csv skenario 20 node OLSR

Pada *file .csv* memiliki beberapa Informasi mencakup waktu simulasi selama 200 detik, ukuran data yang diterima, jumlah paket yang diterima, jumlah jalur atau koneksi yang digunakan (dalam hal ini 10 koneksi), jenis protokol *routing* yang digunakan, dan daya transmisi sebesar 15dBm. *File .csv* ini digunakan untuk menghitung nilai parameter *throughput*. Untuk mencari nilai *throughput*, perlu diketahui jumlah total data yang diterima serta durasi waktu simulasi.

2. *File .flowmon*

Flow monitor (flowmon) merupakan modul yang digunakan untuk mengumpulkan dan menganalisis statistik lalu lintas jaringan selama simulasi. *File .flowmon* ini merupakan salah satu *file* yang dihasilkan setelah kita melakukan simulasi. Pada gambar 3.25 adalah tampilan dari *file .flowmon* yang memuat beberapa data, seperti jumlah paket yang diterima, jumlah paket yang hilang, jumlah paket yang dikirim, dan *delay*.

Flow Id:1 =====	Flow Id:2 =====	Flow Id:3 =====
UDP 10.1.1.13/49153---->10.1.1.3/9	UDP 10.1.1.16/49153---->10.1.1.6/9	UDP 10.1.1.15/49153---->10.1.1.5/9
Tx bitrate:2.92921kbps Rx bitrate:1.69234kbps Mean delay:26.6358ms Packet Loss ratio:44.2748%	Tx bitrate:1.1574kbps Rx bitrate:0.243261kbps Mean delay:2.17914ms Packet Loss ratio:80.137%	Tx bitrate:0.919537kbps Rx bitrate:0.374061kbps Mean delay:13.3465ms Packet Loss ratio:57.6577%
timeFirstTxPacket= 1.00412e+11ns timeFirstRxPacket= 1.04677e+11ns timeLastTxPacket= 1.99912e+11ns timeLastRxPacket= 1.9992e+11ns delaySum= 5.83324e+09ns jitterSum= 4.29389e+09ns lastDelay= 5.83324e+09ns txBytes= 36432 rxBytes= 20148 txPackets= 396 rxPackets= 219 lostPackets= 174 timesForwarded= 443	timeFirstTxPacket= 1.00536e+11ns timeFirstRxPacket= 1.04546e+11ns timeLastTxPacket= 1.95286e+11ns timeLastRxPacket= 1.92287e+11ns delaySum= 6.31952e+07ns jitterSum= 2.5675e+07ns lastDelay= 6.31952e+07ns txBytes= 13708 rxBytes= 2668 txPackets= 149 rxPackets= 29 lostPackets= 117 timesForwarded= 67	timeFirstTxPacket= 1.00629e+11ns timeFirstRxPacket= 1.04404e+11ns timeLastTxPacket= 1.99879e+11ns timeLastRxPacket= 1.96881e+11ns delaySum= 6.27287e+08ns jitterSum= 3.50858e+08ns lastDelay= 6.27287e+08ns txBytes= 11408 rxBytes= 4324 txPackets= 124 rxPackets= 47 lostPackets= 64 timesForwarded= 109

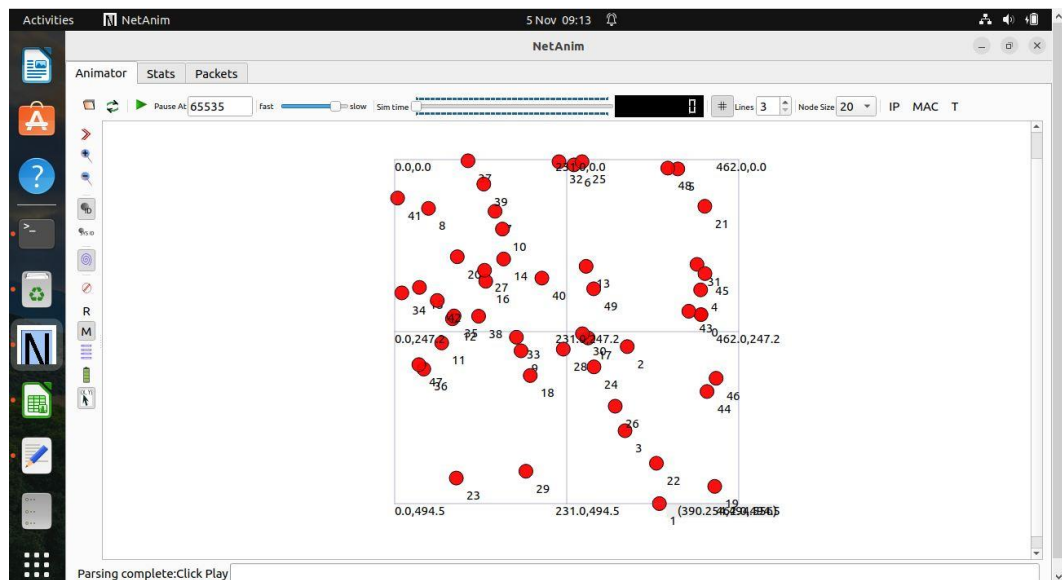
Gambar 3.25 *File .flowmon*

File ini digunakan untuk menghitung nilai *packet loss*, PDR, dan *delay*. Untuk menentukan nilai *packet loss*, informasi yang relevan dapat ditemukan dalam kolom "*lost packet*" dimana *file* tersebut menunjukkan jumlah paket *lost*. Untuk menghitung PDR, perlu diketahui jumlah paket yang diterima dan jumlah paket yang dikirim dimana pada *file* tersebut ditunjukkan dengan nama *Tx bitrate* dan *Rx bitrate*. Sedangkan nilai *delay* dapat dilihat pada kolom "*mean delay*" yang

mencantumkan rata-rata *delay*. Dalam *flowmonitor* atau file *.flowmon* ini, *delay* dan *packet loss* telah dihitung otomatis. Oleh karena itu, untuk menghitung rata-rata *delay* dan *packet loss*, cukup menggunakan informasi dari 10 koneksi atau *flow id* yang tersedia.

3. File *.xml*

File *.xml* merupakan file yang digunakan untuk memvisualisasikan simulasi yang dilakukan. File ini berfungsi untuk menyimpan dan memvisualisasikan simulasi yang dijalankan. File *.xml* digunakan untuk menentukan topologi jaringan yang digunakan. Pada gambar 3.26 adalah tampilan dari file *.xml* yang digunakan untuk menampilkan simulasi dalam bentuk animasi.



Gambar 3.26 File *.xml*

Pada gambar 3.26 merupakan tampilan simulasi dengan skenario 20 *node* yang menggunakan *routing* protokol OLSR. Dimensi atau luas area yang digunakan dalam simulasi tersebut adalah 500x500 meter. File ini dapat dibuka menggunakan *network animator* atau *NetAnim*, sebuah aplikasi animasi jaringan yang terintegrasi dengan NS3.

3.1.5 Skenario Pengujian

Perancangan skenario pengujian pada penelitian ini adalah dengan variasi jumlah *node* yang digunakan adalah 20, 30, 40, dan 50 serta menggunakan

routing protokol OLSR dan DSDV dengan pergerakan *node random walk* pada jaringan MANET. Pengujian akan dilakukan dengan melakukan simulasi menggunakan *routing* protokol OLSR dengan variasi jumlah *node* yang digunakan yaitu 20, 30, 40, dan 50 *node*. Kemudian, pengujian dengan *routing* protokol DSDV dengan variasi jumlah *node* yang sama dengan sebelumnya. Setelah melakukan pengujian. Peneliti mendapatkan 3 *file* yang dihasilkan dalam pengujian. *File* tersebut dapat digunakan untuk mendapatkan data atau nilai QOS untuk dapat dianalisis dan diambil kesimpulan. Pada tabel 3.2 merupakan perancangan skenario pengujian yang akan dilakukan dalam penelitian ini.

Tabel 3.2 Skenario pengujian

Skenario	Protokol <i>Routing</i>	Jumlah <i>Node</i>	Pergerakan <i>Node</i>
1	OLSR	20	<i>RandomWalk</i>
2	DSDV	20	<i>RandomWalk</i>
3	OLSR	30	<i>RandomWalk</i>
4	DSDV	30	<i>RandomWalk</i>
5	OLSR	40	<i>RandomWalk</i>
6	DSDV	40	<i>RandomWalk</i>
7	OLSR	50	<i>RandomWalk</i>
8	DSDV	50	<i>RandomWalk</i>

3.1.6 Analisis dan Kesimpulan

Analisis dan kesimpulan akan dijelaskan dibab selanjutnya. Analisis dan kesimpulan dilakukan setelah melakukan pengambilan data. Analisis dilakukan dengan membandingkan *routing* protokol OLSR dan DSDV dengan pergerakan *random walk* dan variasi jumlah *node* yaitu 20, 30, 40, dan 50 *node* yang sudah ditentukan.