DEFENDING BLACK HOLE ATTACK IN MANET

FAHMID AL RIFAT
SUBMITTED BY : 1705087

OLDMEAD SECONDARY SCHOOL
SCIENCE CLASS

REFERENCE

CONCEPT AND ALGORITHM

ALGORITHM:

2014 - Prevention of Black Hole Attack in AODV Routing Algorithm of MANET Using Trust Based Computing
-Ashish Sharma , Dinesh Bhuriya , Upendra Singh , Sushma Singh
-International Journal of Computer Science and Information Technologies

2018 -A Secure and Trust based Approach to Mitigate Blackhole Attack on AODV based Manet
-mh Kamel , Ibrahim alameri , ameer N Onaizah

OVERALL:

2018 - Routing AODV Defending Black Hole Attack through NS3 in Manet
-Anupam Mishra , Rajeev Paulus , Aditi Agrawa
-International Journal of Computer Applications

SIMULATION:

2018 - Evaluation of MANET Routing Protocols under Black Hole Attack Using AODV and OLSR in NS3
-Abdellah Nabou , My Driss Laanaoui , Mohammed Ouzzif

REFERENCE - LINKS

CONCEPT AND ALGORITHM

ALGORITHM:

2014 - Microsoft Word - 84. adhoc ntetwork peper (psu.edu)
2018 - https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8054219

OVERALL:

2018 - https://www.researchgate.net/profile/Rajeev-Paulus/publication/327224082_Routing_AODV_Defending_Black_Hole_Attack_through_NS3_in_Manet/links/5b9eed56299bf13e6037c364/Routing-AODV-Defending-Black-Hole-Attack-through-NS3-in-Manet.pdf

SIMULATION:

2018 - https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8629603

MANET

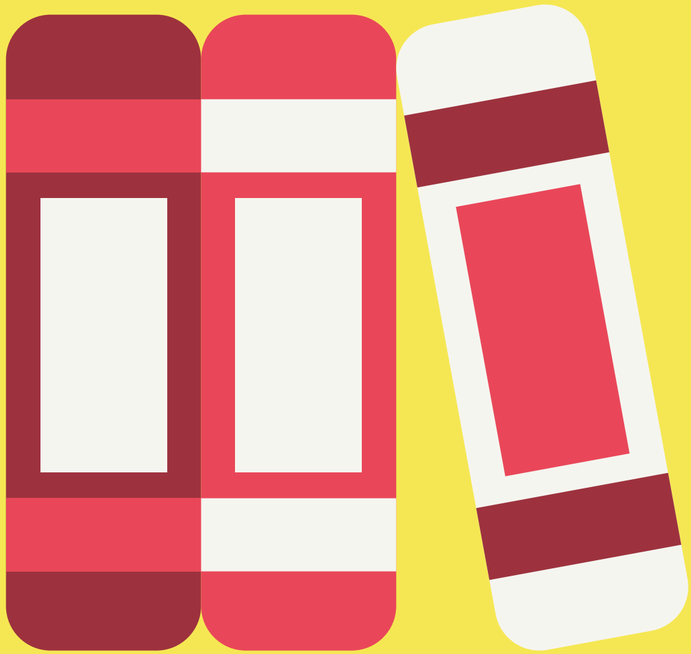
A **MOBILE AD HOC** NETWORK (MANET) IS A CONTINUOUSLY **SELF-CONFIGURING,** SELF-ORGANIZING, I**NFRASTRUCTURE-LESS** NETWORK OF MOBILE DEVICES CONNECTED WITHOUT WIRE

AODV

AODV (AD-HOC ON-DEMAND DISTANCE VECTOR) IS A LOOP-FREE ROUTING PROTOCOL FOR AD-HOC NETWORKS. IT IS DESIGNED TO BE **SELF-STARTING** IN AN ENVIRONMENT OF MOBILE NODES, WITHSTANDING A VARIETY OF NETWORK BEHAVIORS SUCH AS NODE **MOBILITY, LINK** FAILURES AND PACKET LOSSES

BLACK HOLE ATTACK

BLACK-HOLE ATTACKS OCCUR WHEN A ROUTER **DELETES ALL MESSAGES** IT IS SUPPOSED TO FORWARD



TODAY 'S OBJECTIVE

01 . Aodv implementation in Manet

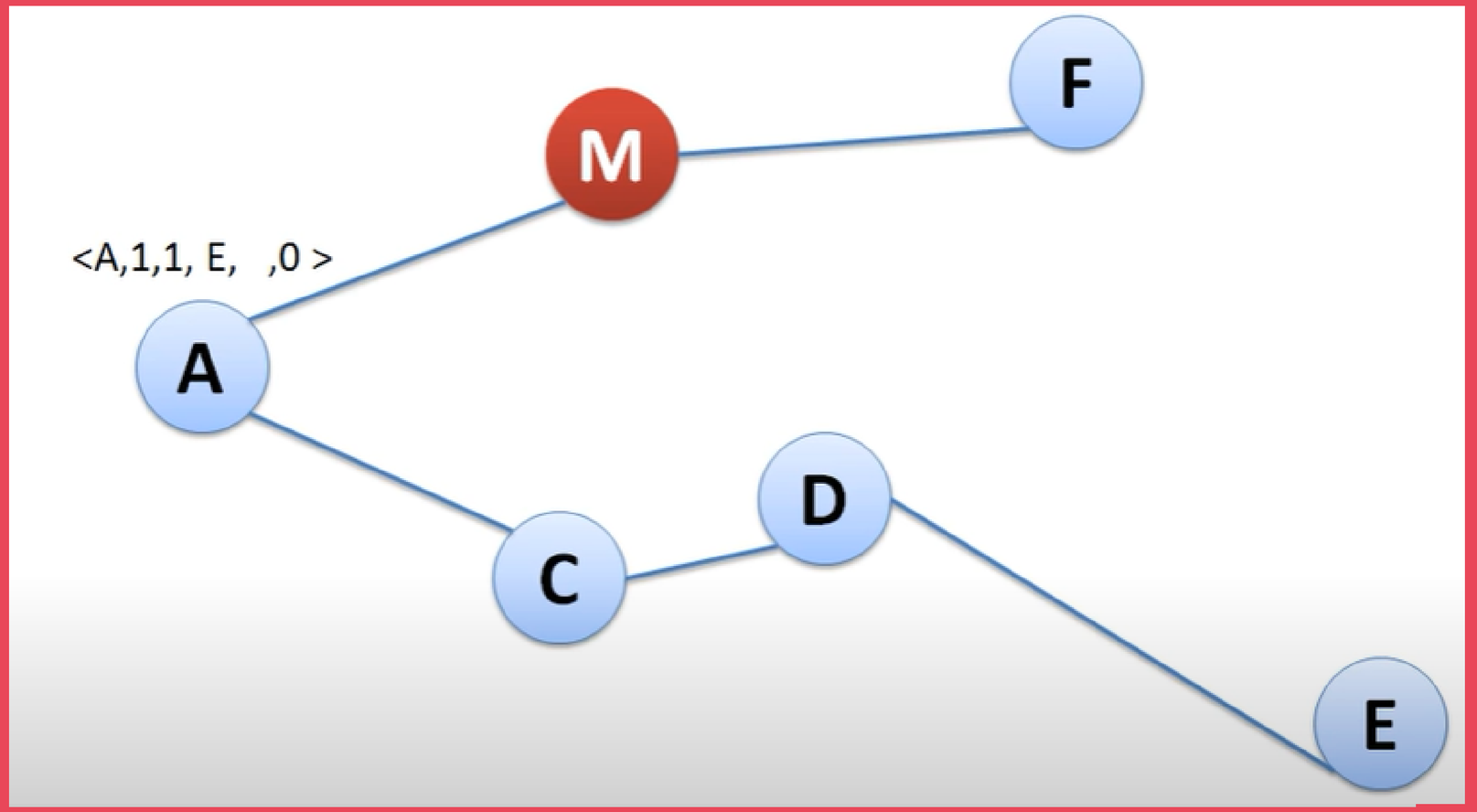
02. Calculate Performence Metrices

03. show data flow

OLDMEAD SECONDARY SCHOOL

SCIENCE CLASS

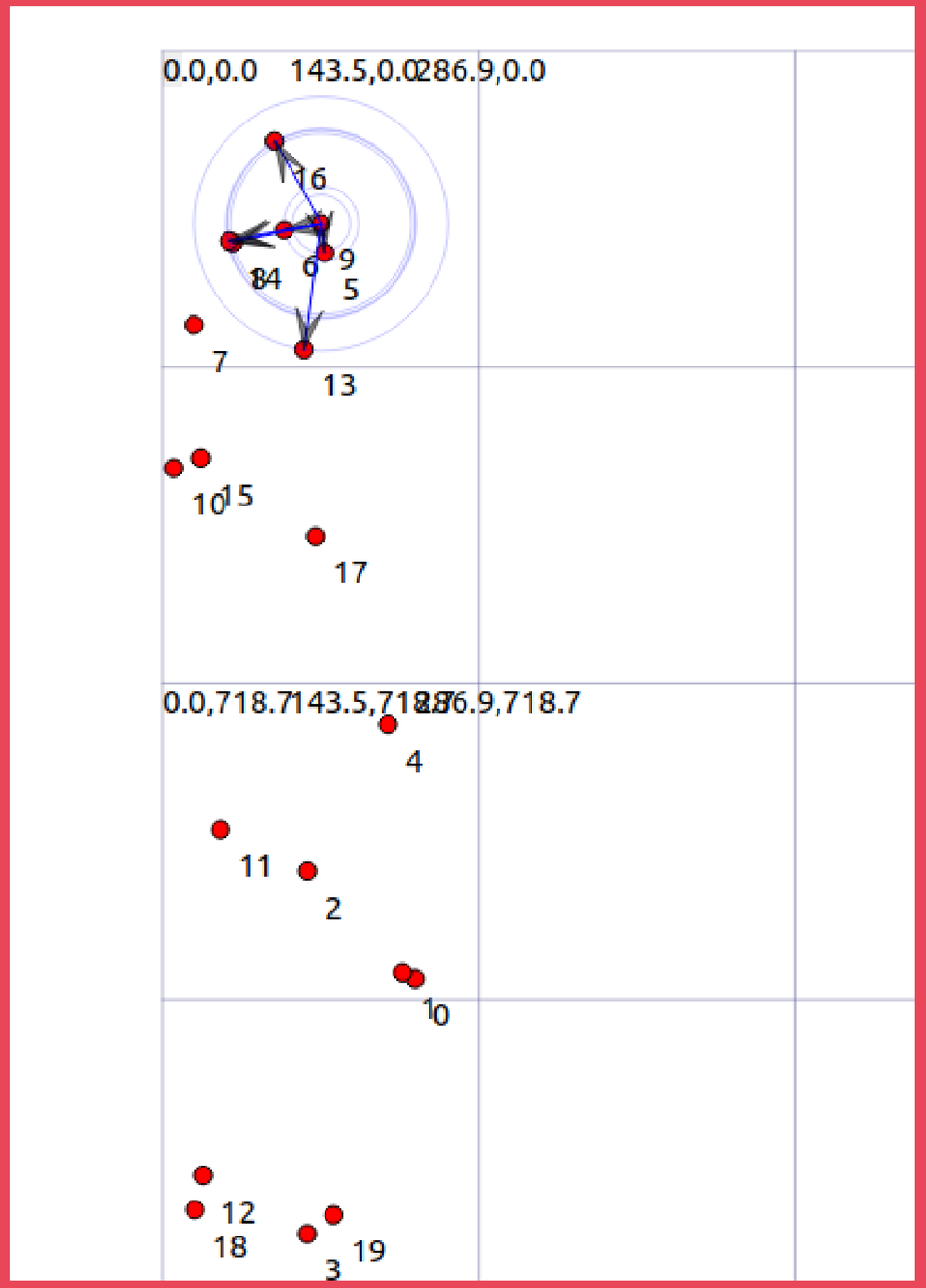
AODV IN MANET (BLACK WHOLE ATTACK)



OLDMEAD SECONDARY SCHOOL

SCIENCE CLASS

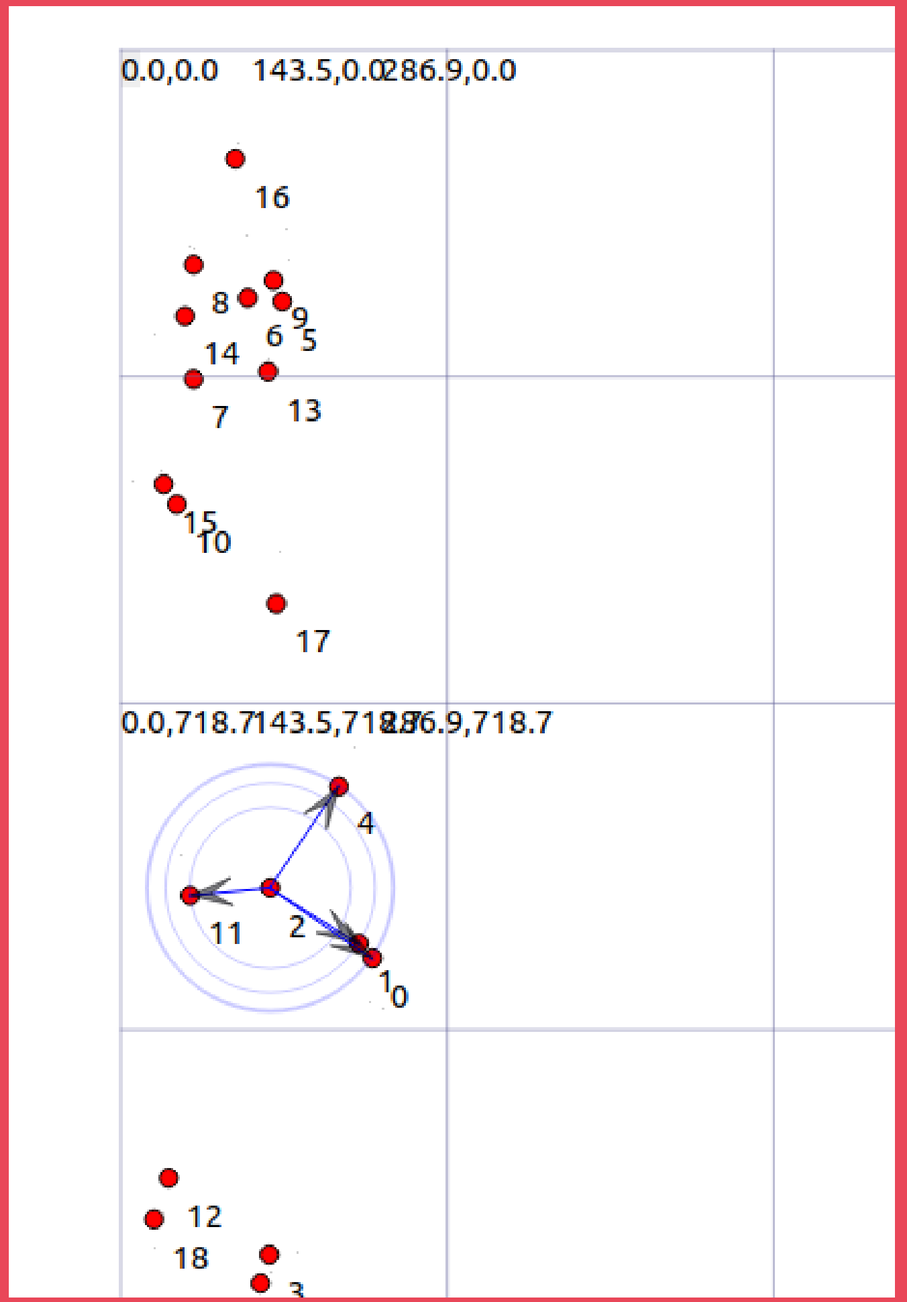
AODV -NETANIM



OLDMEAD SECONDARY SCHOOL

SCIENCE CLASS

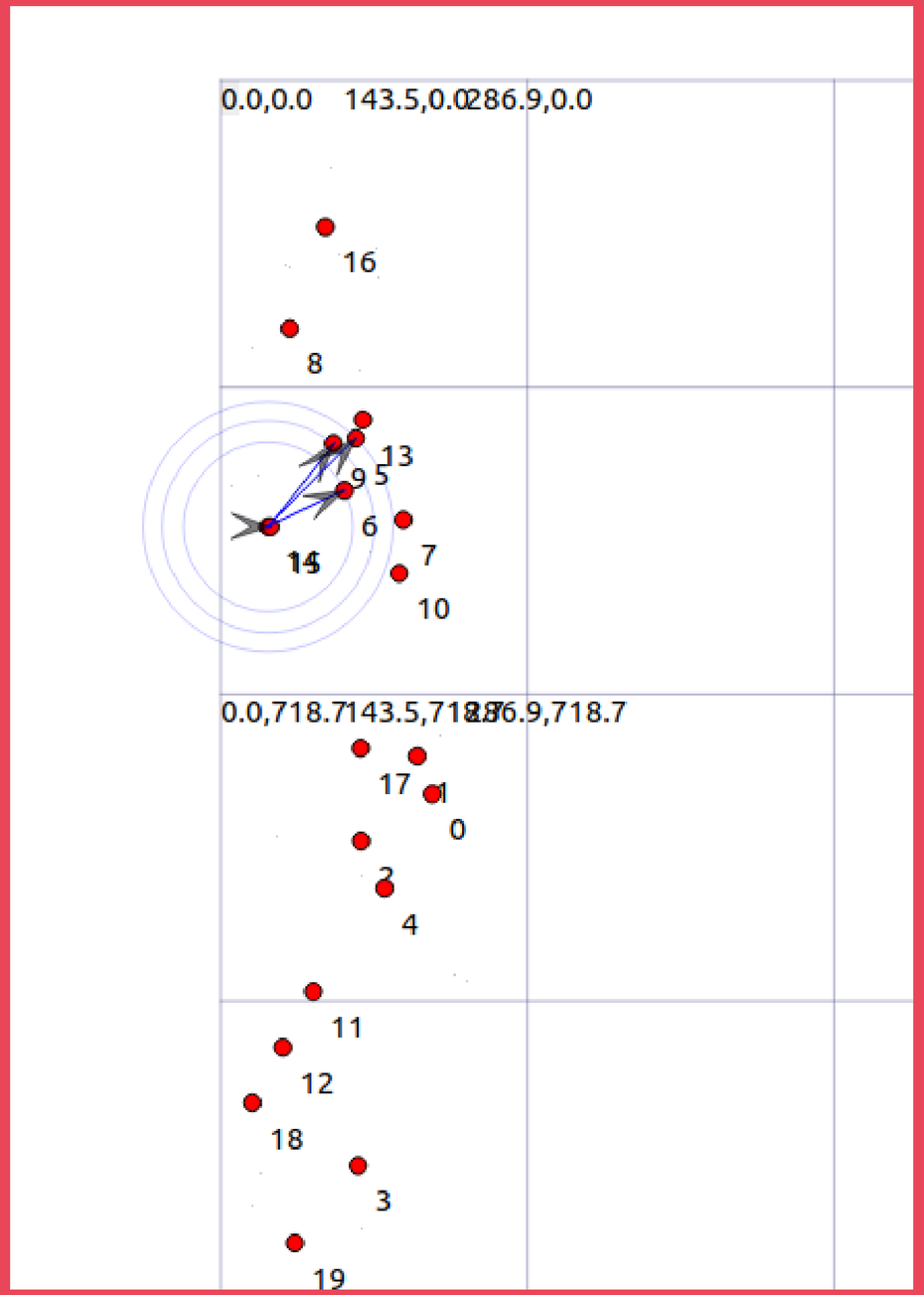
AODV -NETANIM



OLDMEAD SECONDARY SCHOOL

SCIENCE CLASS

AODV -NETANIM



OLDMEAD SECONDARY SCHOOL

SCIENCE CLASS

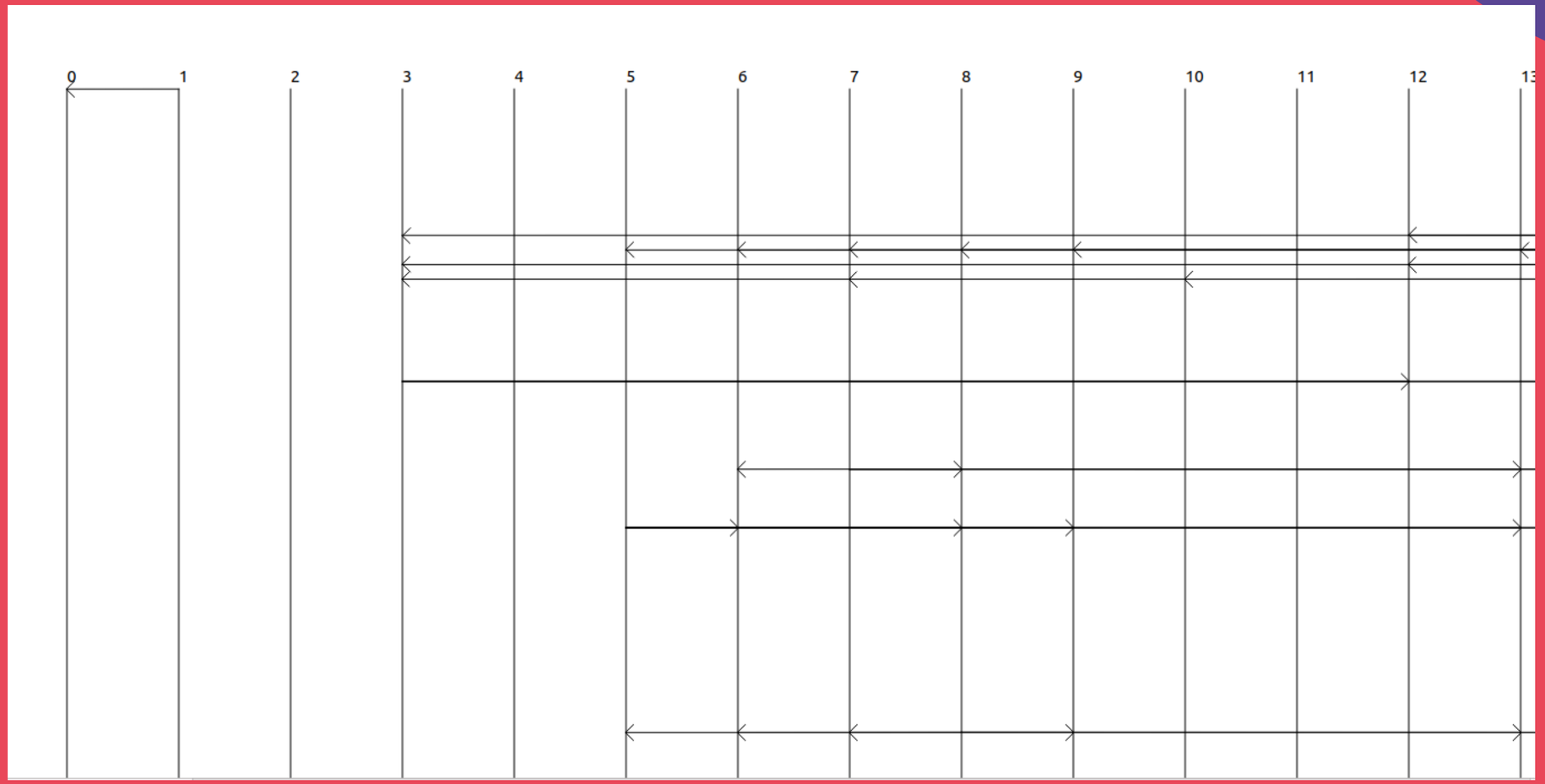
AODV-NETANIM-NODE DETAILS

Node:0 IP: 10.1.1.1 127.0.0.1 IPv6: ::1 MAC: 00:00:00:00:00:01	Node:1 IP: 10.1.1.2 127.0.0.1 IPv6: ::1 MAC: 00:00:00:00:00:02	Node:2 IP: 127.0.0.1 10.1.1.3 IPv6: ::1 MAC: 00:00:00:00:00:03	Node:3 IP: 10.1.1.4 127.0.0.1 IPv6: ::1 MAC: 00:00:00:00:00:04	Node:4 IP: 10.1.1.5 127.0.0.1 IPv6: ::1 MAC: 00:00:00:00:00:05	Node:5 IP: 10.1.1.6 127.0.0.1 IPv6: ::1 MAC: 00:00:00:00:00:06	Node:6 IP: 10.1.1.7 127.0.0.1 IPv6: ::1 MAC: 00:00:00:00:00:07
Node:7 IP: 10.1.1.8 127.0.0.1 IPv6: ::1 MAC: 00:00:00:00:00:08	Node:8 IP: 10.1.1.9 127.0.0.1 IPv6: ::1 MAC: 00:00:00:00:00:09	Node:9 IP: 10.1.1.10 127.0.0.1 IPv6: ::1 MAC: 00:00:00:00:00:0a	Node:10 IP: 10.1.1.11 127.0.0.1 IPv6: ::1 MAC: 00:00:00:00:00:0b	Node:11 IP: 10.1.1.12 127.0.0.1 IPv6: ::1 MAC: 00:00:00:00:00:0c	Node:12 IP: 10.1.1.13 127.0.0.1 IPv6: ::1 MAC: 00:00:00:00:00:0d	Node:13 IP: 10.1.1.14 127.0.0.1 IPv6: ::1 MAC: 00:00:00:00:00:0e
Node:14 IP: 10.1.1.15 127.0.0.1 IPv6: ::1 MAC: 00:00:00:00:00:0f	Node:15 IP: 10.1.1.16 127.0.0.1 IPv6: ::1 MAC: 00:00:00:00:00:10	Node:16 IP: 10.1.1.17 127.0.0.1 IPv6: ::1 MAC: 00:00:00:00:00:11	Node:17 IP: 10.1.1.18 127.0.0.1 IPv6: ::1 MAC: 00:00:00:00:00:12	Node:18 IP: 10.1.1.19 127.0.0.1 IPv6: ::1 MAC: 00:00:00:00:00:13	Node:19 IP: 10.1.1.20 127.0.0.1 IPv6: ::1 MAC: 00:00:00:00:00:14	

OLDMEAD SECONDARY SCHOOL

SCIENCE CLASS

AODV-NETANIM-STATS -SINK+APP..



OLDMEAD SECONDARY SCHOOL

SCIENCE CLASS

RESEARCH PAPER PEREMETERS

Table.1. Simulation Parameters

Parameters	Values
Network size	700 m * 700 m
Number of Nodes	10-70
Max. speed/mobility	15.0 m/sec
Pause Time	3.0 s
Traffic Model	CBR
Routing Protocol	AODV
Simulation Time	100

OUR CODE PEREMETERS

```
Packet::EnablePrinting ();
m_nSinks = nSinks;
m_txp = txp;
m_CSVfileName = CSVfileName;

int nWifis = 20;

double TotalTime = 100.0;
std::string rate ("2048bps");
std::string phyMode ("DsssRate11Mbps");
std::string tr_name ("manet-aodv-1705087");
int nodeSpeed = 20; //in m/s
int nodePause = 0; //in s
m_protocolName = "protocol";
```

OLDMEAD SECONDARY SCHOOL

SCIENCE CLASS

CONFIGARATIONS

Term Assignment on ns-3

1. Simulate the following networks as per assignment:

Std_id % 6	Wired	Wireless high-rate (e.g., 802.11) (static)	Wireless high-rate (e.g., 802.11) (mobile)	Wireless low-rate (e.g., 802.15.4) (static)	Wireless low-rate (e.g., 802.15.4) (mobile)
0		√		√	
1			√		√
2			√	√	
3	√				√
4		√			√
5	√			√	

EXPECTATIONS AND OUTCOMES

01 . Packet Delivery Ratio

A decrease in PDR is seen at the same time that is a black hole attack on AODV . we can see which is a successful increment in the PDR of modified AODV

02. Average Throughput

Throughtput is decreases due to black hole attack but without black hole it is increases in modified routing AODVsee which is a successful increment in the PDR of modified AODV

EXPECTATIONS AND OUTCOMES

03 . Average End-to-End delay

Average End to End Delay with black hole attack is much higher than without black hole attack in AODV routing protocol.

04. Packet Drop Ratio

It is the ratio of the data lost at destination to those generated by the CBR sources . It increases as data travel much distance than before

6.2 Average Throughput

It is the average rate of successful message delivery over a communication channel. It is measured in data packets per second.

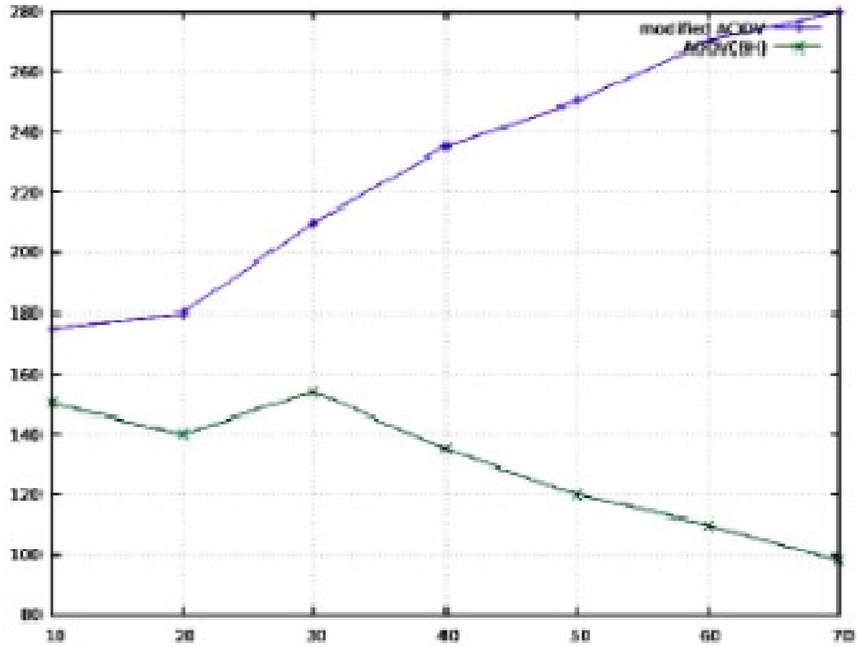


Fig 4: Average Throughput Vs No. of Nodes

WAY 1 - IN CODE

```
RoutingExperiment::ReceivePacket (Ptr<Socket> socket)
{
  Ptr<Packet> packet;
  Address senderAddress;
  while ((packet = socket->RecvFrom (senderAddress)))
    {
      bytesTotal += packet->GetSize ();
      packetsReceived += 1;
      NS_LOG_UNCOND (PrintReceivedPacket (socket, packet, senderAddress));
    }
}

void
RoutingExperiment::CheckThroughput ()
{
  double kbs = (bytesTotal * 8.0) / 1000;
  bytesTotal = 0;

  std::ofstream out (m_CSVfileName.c_str (), std::ios::app);

  out << (Simulator::Now ()).GetSeconds () << ","
      << kbs << ","
      << packetsReceived << ","
      << m_nSinks << ","
      << m_protocolName << ","
      << m_txp << ""
      << std::endl;

  out.close ();
```

WAY 1 - IN CODE -> CSV (20 NODE)

```
0 0 0 5 AODV 8.5
1 0 0 5 AODV 8.5
2 0 0 5 AODV 8.5
3 0 0 5 AODV 8.5
4 0 0 5 AODV 8.5
5 0 0 5 AODV 8.5
6 0 0 5 AODV 8.5
7 0 0 5 AODV 8.5
8 0 0 5 AODV 8.5
9 0 0 5 AODV 8.5
10 0 0 5 AODV 8.5
11 0 0 5 AODV 8.5
12 0 0 5 AODV 8.5
13 0 0 5 AODV 8.5
14 0 0 5 AODV 8.5
15 0 0 5 AODV 8.5
16 0 0 5 AODV 8.5
17 0 0 5 AODV 8.5
18 0 0 5 AODV 8.5
19 0 0 5 AODV 8.5
```

-SIMULATION SEC

-RECIEVED RATE

-PACKET NO

-ROUTING

-POWER

OLDMEAD SECONDARY SCHOOL

SCIENCE CLASS

WAY 1 - IN CODE -> CSV (80-100 SEC)

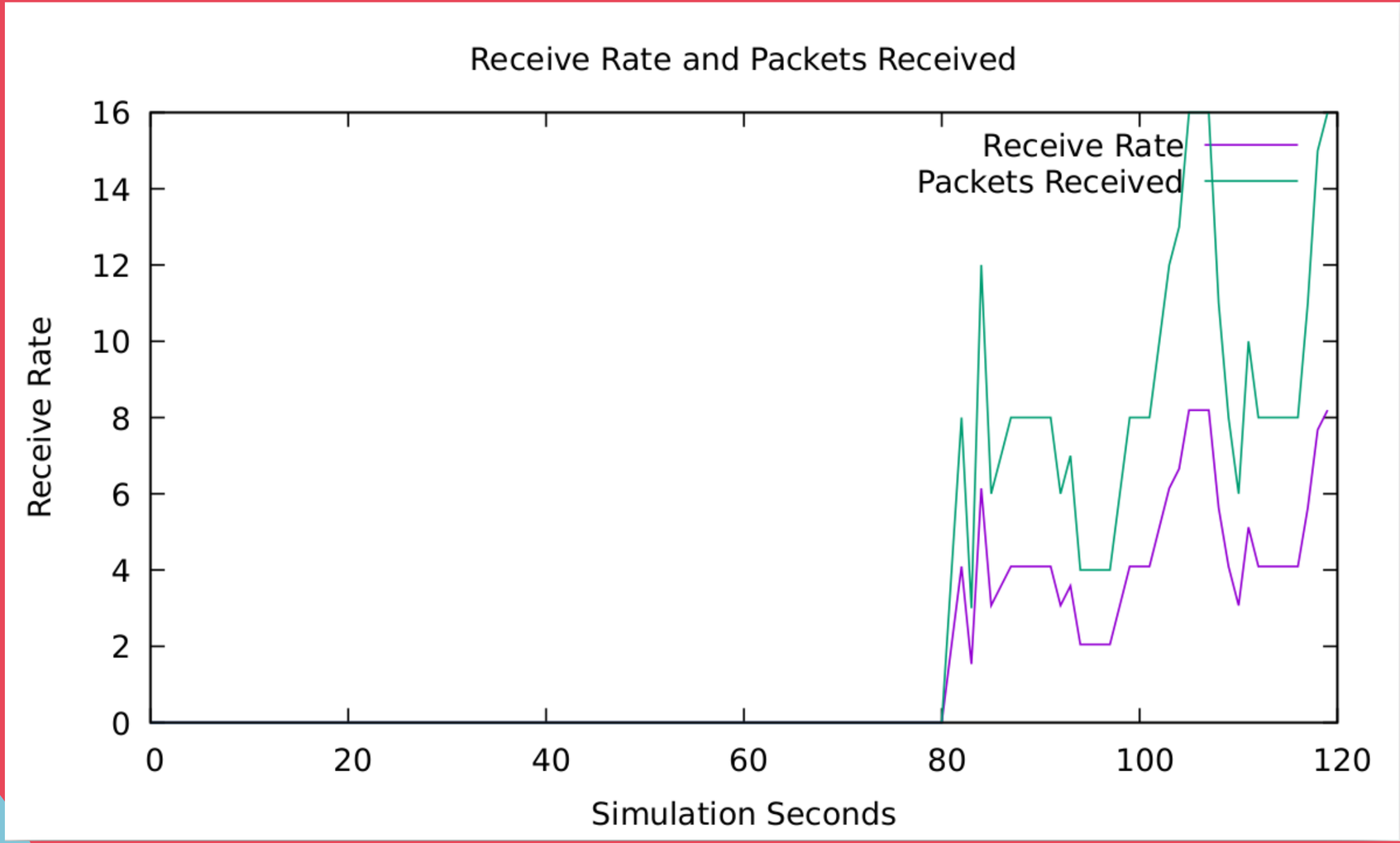
```
79 0 0 5 AODV 8.5
80 0 0 5 AODV 8.5
81 2.048 4 5 AODV 8.5
82 4.096 8 5 AODV 8.5
83 1.536 3 5 AODV 8.5
84 6.144 12 5 AODV 8.5
85 3.072 6 5 AODV 8.5
86 3.584 7 5 AODV 8.5
87 4.096 8 5 AODV 8.5
88 4.096 8 5 AODV 8.5
89 4.096 8 5 AODV 8.5
90 4.096 8 5 AODV 8.5
91 4.096 8 5 AODV 8.5
92 3.072 6 5 AODV 8.5
93 3.584 7 5 AODV 8.5
94 2.048 4 5 AODV 8.5
95 2.048 4 5 AODV 8.5
96 2.048 4 5 AODV 8.5
97 2.048 4 5 AODV 8.5
98 3.072 6 5 AODV 8.5
99 4.096 8 5 AODV 8.5
100 4.096 8 5 AODV 8.5
101 4.096 8 5 AODV 8.5
102 5.12 10 5 AODV 8.5
103 6.144 12 5 AODV 8.5
104 6.656 13 5 AODV 8.5
105 8.192 16 5 AODV 8.5
106 8.192 16 5 AODV 8.5
107 8.192 16 5 AODV 8.5
108 5.632 11 5 AODV 8.5
109 4.096 8 5 AODV 8.5
110 3.072 6 5 AODV 8.5
```

- SIMULATION SEC
- RECIEVED RATE
- PACKET NO
- ROUTING
- POWER

OLDMEAD SECONDARY SCHOOL

SCIENCE CLASS

WAY 1 - IN CODE -> CSV -> GNUPLOT



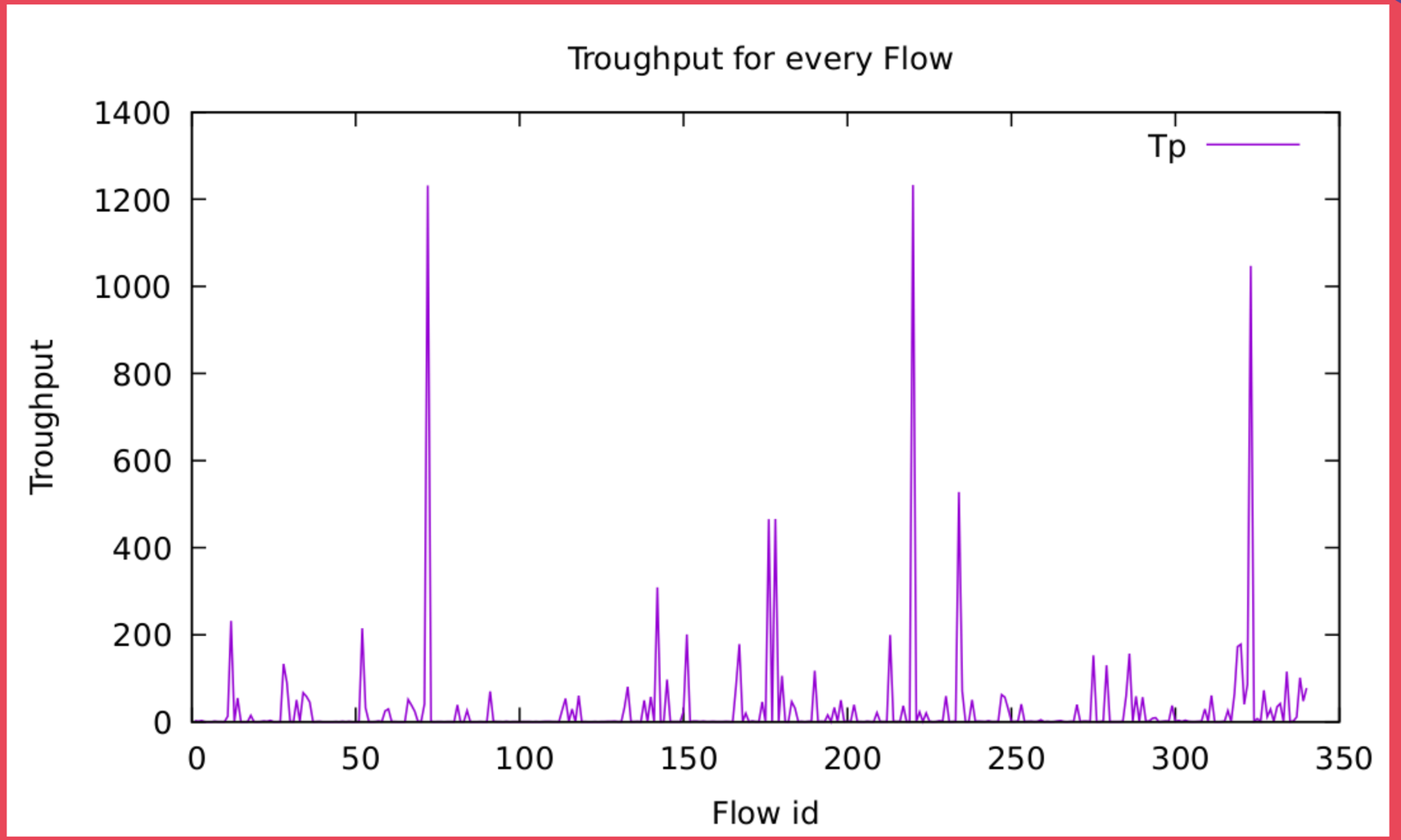
WAY 2 - FLOW MONITOR CODE

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


NS_LOG_INFO ("Run Simulation.");

CheckThroughput ();

AnimationInterface anim ("manet-aodv-1705087.xml");

Simulator::Stop (Seconds (TotalTime));

Simulator::Run ();

flowmon->SerializeToXmlFile ((tr_name + ".flowmon").c_str(), false, false);
```

WAY 2 - FLOW MONITOR CODE -XML

```
<?xml version="1.0" ?>
<FlowMonitor>
  <FlowStats>
    <Flow flowId="1" timeFirstTxPacket="+9.72711e+10ns" timeFirstRxPacket="+9.89154e+10ns"
timeLastTxPacket="+9.97711e+10ns" timeLastRxPacket="+9.97731e+10ns" delaySum="+3.0476e+09ns" jitterSum="+1.64232e+09ns"
lastDelay="+1.98448e+06ns" txBytes="1012" rxBytes="552" txPackets="11" rxPackets="6" lostPackets="0"
timesForwarded="20">
    </Flow>
    <Flow flowId="2" timeFirstTxPacket="+9.7314e+10ns" timeFirstRxPacket="+9.76051e+10ns"
timeLastTxPacket="+9.9814e+10ns" timeLastRxPacket="+9.93201e+10ns" delaySum="+3.51333e+08ns" jitterSum="+2.93116e+08ns"
lastDelay="+6.06412e+06ns" txBytes="1012" rxBytes="828" txPackets="11" rxPackets="9" lostPackets="0"
timesForwarded="30">
    </Flow>
    <Flow flowId="3" timeFirstTxPacket="+9.73211e+10ns" timeFirstRxPacket="+0ns" timeLastTxPacket="+9.98211e+10ns"
timeLastRxPacket="+0ns" delaySum="+0ns" jitterSum="+0ns" lastDelay="+0ns" txBytes="1012" rxBytes="0" txPackets="11"
rxPackets="0" lostPackets="0" timesForwarded="0">
    </Flow>
    <Flow flowId="4" timeFirstTxPacket="+9.73274e+10ns" timeFirstRxPacket="+9.83332e+10ns"
timeLastTxPacket="+9.73274e+10ns" timeLastRxPacket="+9.83332e+10ns" delaySum="+1.00573e+09ns" jitterSum="+0ns"
lastDelay="+1.00573e+09ns" txBytes="48" rxBytes="48" txPackets="1" rxPackets="1" lostPackets="0" timesForwarded="0">
    </Flow>
    <Flow flowId="5" timeFirstTxPacket="+9.73274e+10ns" timeFirstRxPacket="+0ns" timeLastTxPacket="+9.73274e+10ns"
timeLastRxPacket="+0ns" delaySum="+0ns" jitterSum="+0ns" lastDelay="+0ns" txBytes="48" rxBytes="0" txPackets="1"
rxPackets="0" lostPackets="0" timesForwarded="0">
    </Flow>
    <Flow flowId="6" timeFirstTxPacket="+9.73274e+10ns" timeFirstRxPacket="+9.83299e+10ns"
timeLastTxPacket="+9.73274e+10ns" timeLastRxPacket="+9.83299e+10ns" delaySum="+1.00246e+09ns" jitterSum="+0ns"
lastDelay="+1.00246e+09ns" txBytes="48" rxBytes="48" txPackets="1" rxPackets="1" lostPackets="0" timesForwarded="0">
    </Flow>
    <Flow flowId="7" timeFirstTxPacket="+9.73274e+10ns" timeFirstRxPacket="+0ns" timeLastTxPacket="+9.73274e+10ns"
timeLastRxPacket="+0ns" delaySum="+0ns" jitterSum="+0ns" lastDelay="+0ns" txBytes="48" rxBytes="0" txPackets="1"
rxPackets="0" lostPackets="0" timesForwarded="0">
    </Flow>
    <Flow flowId="8" timeFirstTxPacket="+9.73274e+10ns" timeFirstRxPacket="+9.83306e+10ns"
timeLastTxPacket="+9.73274e+10ns" timeLastRxPacket="+9.83306e+10ns" delaySum="+1.00313e+09ns" jitterSum="+0ns"
lastDelay="+1.00313e+09ns" txBytes="48" rxBytes="48" txPackets="1" rxPackets="1" lostPackets="0" timesForwarded="0">
```

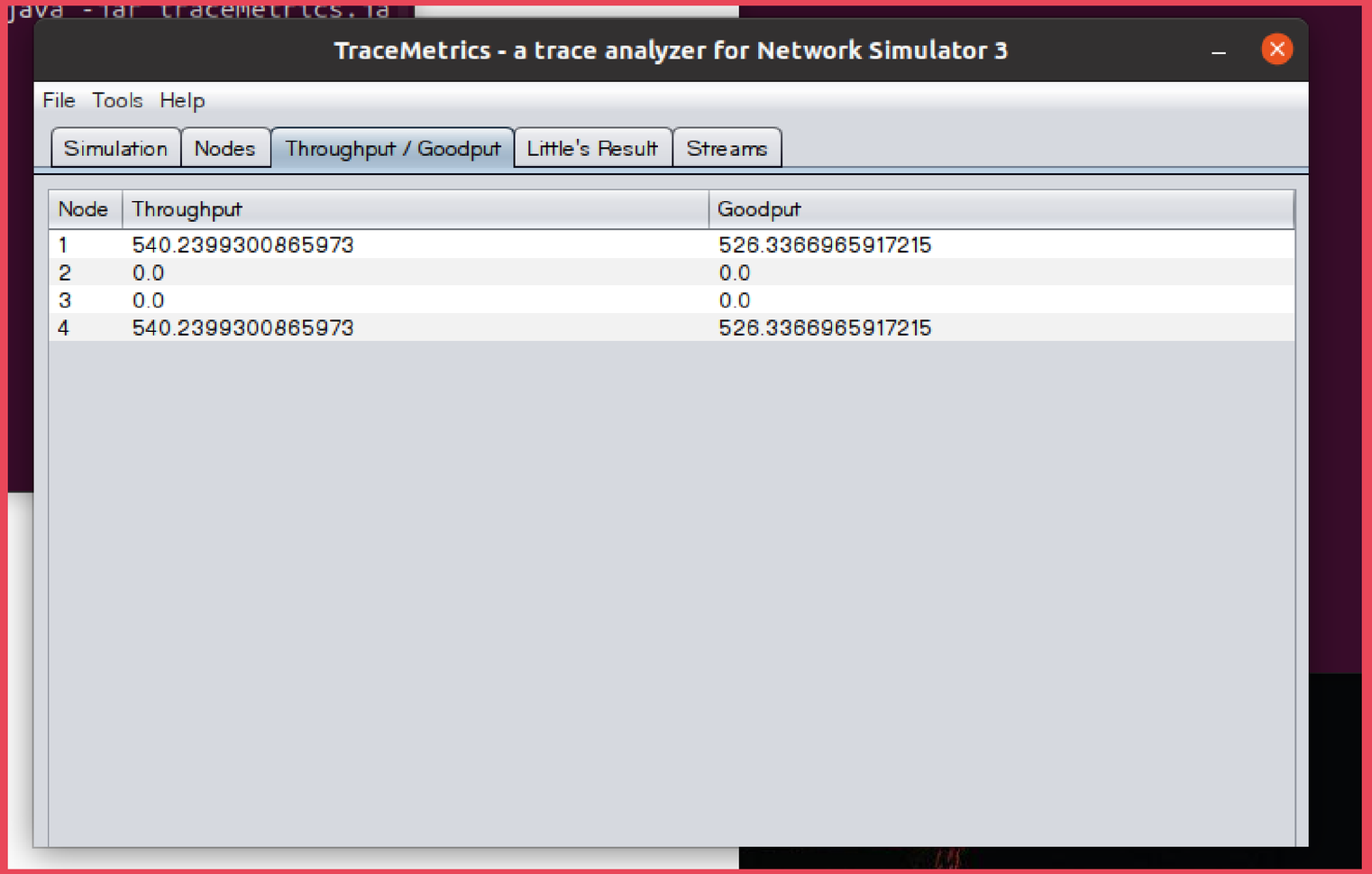
WAY 2 - FLOW MONITOR FILE ->XML ->PARSE WITH PYTHON



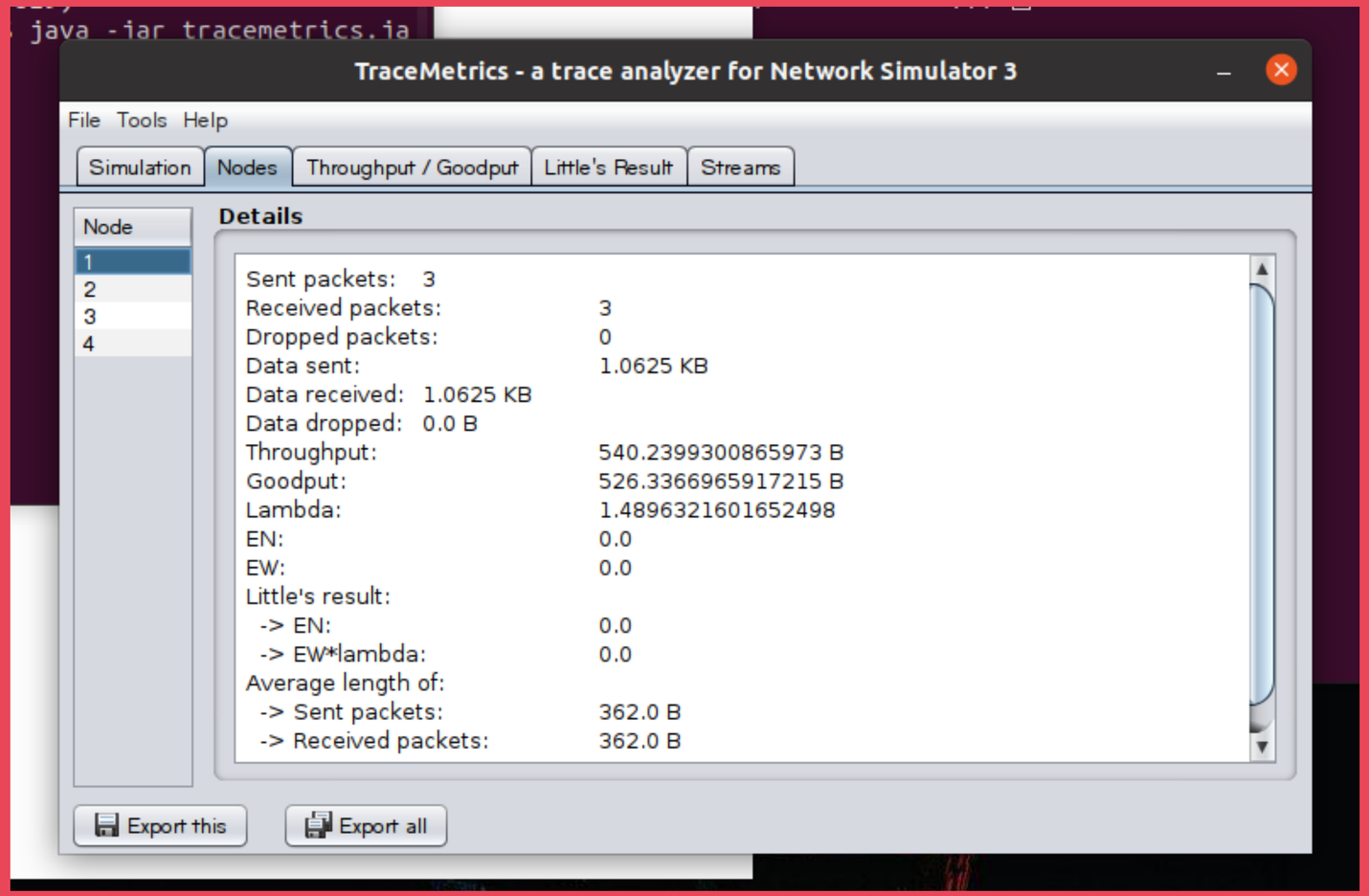
WAY 3 - TRACE METRIC -PROB INADHOC ONLY

```
NS_LOG_INFO ("Configure Tracing.");
tr_name = tr_name + "_tr_";

AsciiTraceHelper ascii;
Ptr<OutputStreamWrapper> osw = ascii.CreateFileStream ( (tr_name + ".tr").c_str());
wifiPhy.EnableAsciiAll (osw);
//AsciiTraceHelper ascii;
//MobilityHelper::EnableAsciiAll (ascii.CreateFileStream (tr_name + ".mob"));
```

WAY 3 - TRACE METRIC -PROB INADHOC ONLY



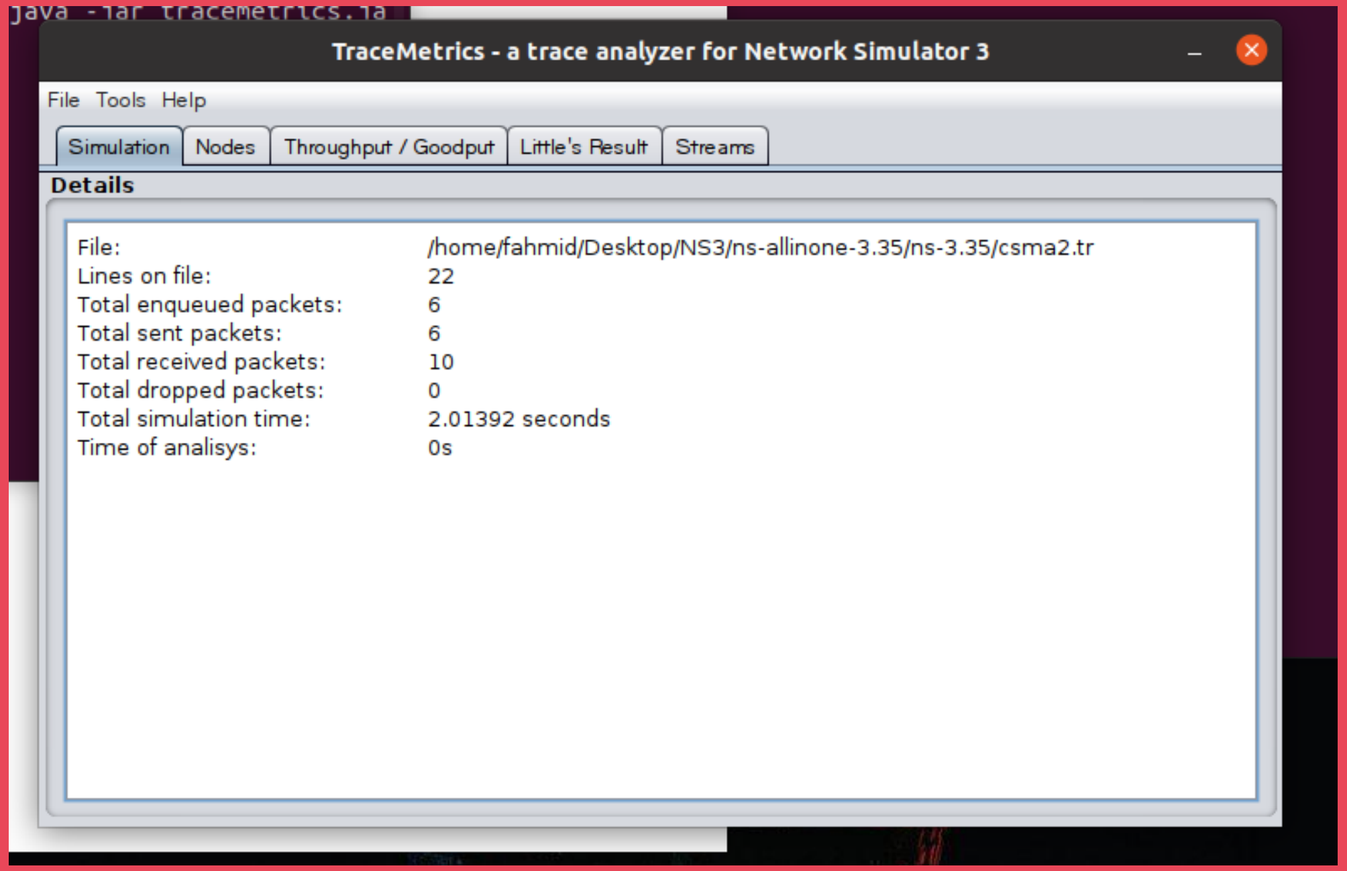
WAY 3 - TRACE METRIC -PROB INADHOC ONLY

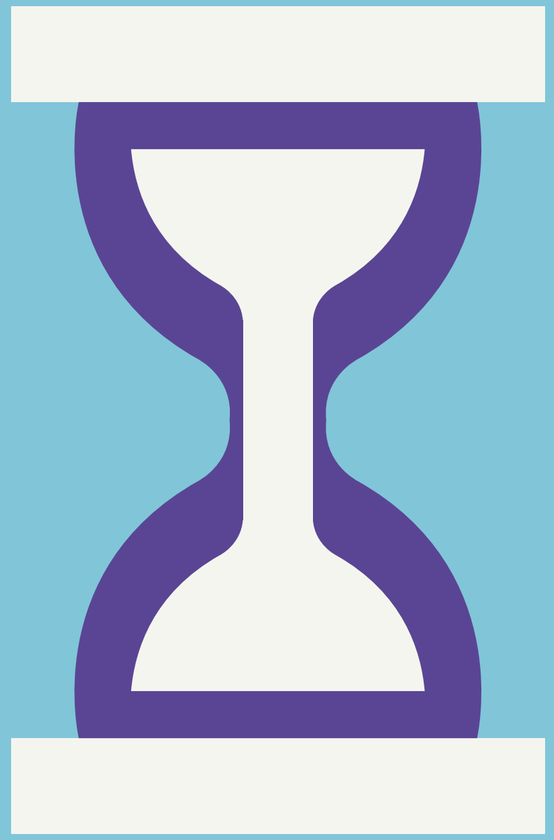


OLDMEAD SECONDARY SCHOOL

SCIENCE CLASS

WAY 3 - TRACE METRIC -PROB INADHOC ONLY





6.1 Packet Delivery Ratio

It can be defined as the ratio of total number of data packets delivered to the destination to the total number of data packets generated by the source. It is figured as –

$$P = \frac{number\ of\ packets\ sent}{number\ of\ packets\ received} \times 100$$

A decrease in PDR is seen at the same time that is a black hole attack on AODV. In the outcomes, we can see which is a successful increment in the PDR of modified AODV.

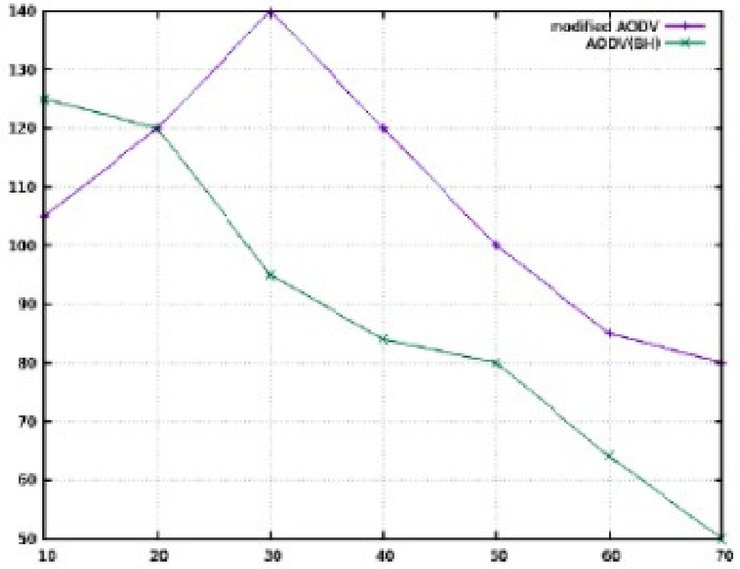
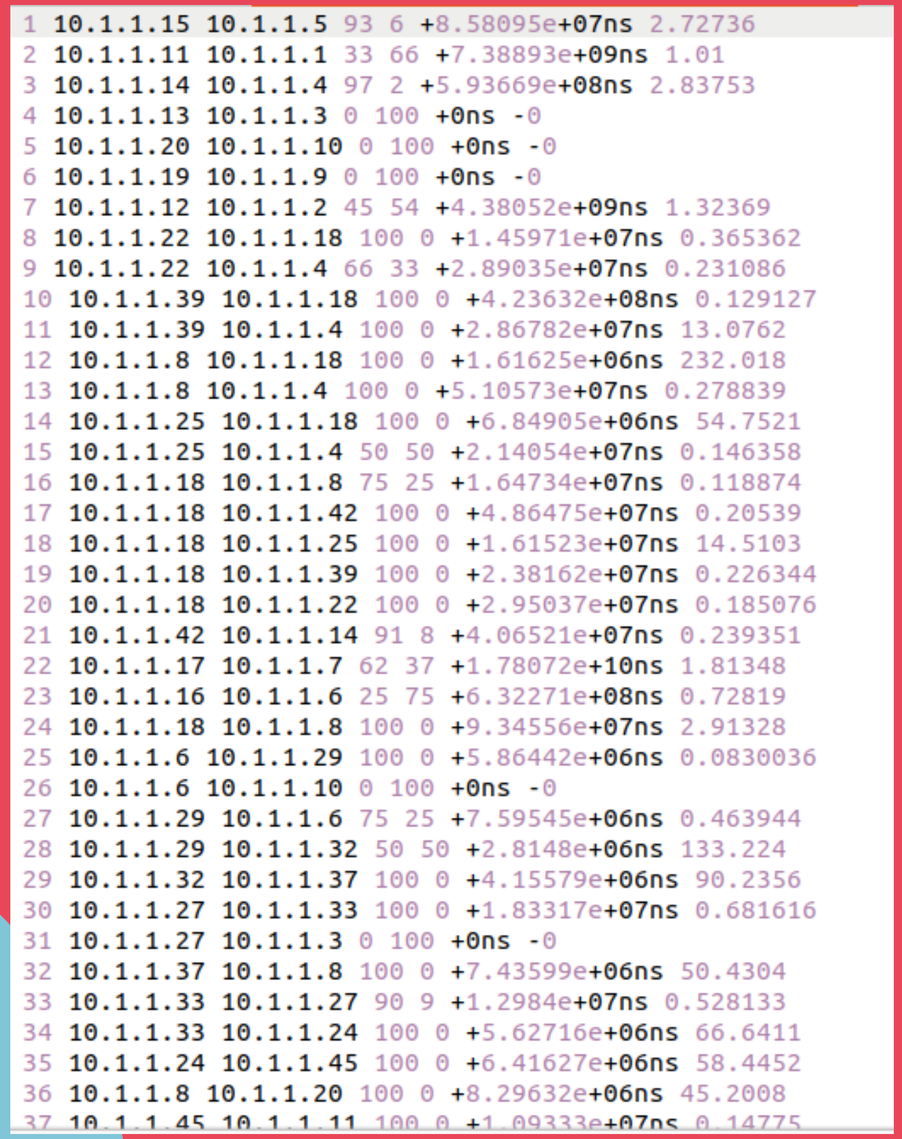


Fig.3. Packet Delivery Ratio vs No. of nodes

FLOWMOINTOR->CSV (DIRECT)

```
std::string m_CSVfileName2="Performance_manet_aodv_1705087_2.csv";

std::ofstream out2 (m_CSVfileName2.c_str (), std::ios::app);

Ptr<Ipv4FlowClassifier> classifier = DynamicCast<Ipv4FlowClassifier> (flowmon.GetClassifier ());
  std::map<FlowId, FlowMonitor::FlowStats> stats = monitor->GetFlowStats ();

  for (std::map<FlowId, FlowMonitor::FlowStats>::const_iterator iter = stats.begin (); iter != stats.end (); ++iter)
    {
    Ipv4FlowClassifier::FiveTuple t = classifier->FindFlow (iter->first);

    out2 << (Simulator::Now ()).GetSeconds () << ","
      << iter->first << ","  // Flow ID
      << t.sourceAddress << "," // Src Addr
      << t.destinationAddress << ","// des addr
      << iter->second.rxPackets*100/iter->second.txPackets << "," // Packet delivery ratio
      << (iter->second.txPackets-iter->second.rxPackets)*100/iter->second.txPackets << "," // Packet loss ratio
      << iter->second.delaySum << "," // Delay
      << iter->second.rxBytes * 8.0/(iter->second.timeLastRxPacket.GetSeconds()-iter->second.timeFirstTxPacket.GetSeconds())/1024 << "" // Throughput
      << std::endl;

SentPackets = SentPackets +(iter->second.txPackets);
ReceivedPackets = ReceivedPackets + (iter->second.rxPackets);
LostPackets = LostPackets + (iter->second.txPackets-iter->second.rxPackets);
AvgThroughput = AvgThroughput + (iter->second.rxBytes * 8.0/(iter->second.timeLastRxPacket.GetSeconds()-iter->second.timeFirstTxPacket.GetSeconds())/1024
Delay = Delay + (iter->second.delaySum);
Jitter = Jitter + (iter->second.jitterSum);
```

FLOWMOINTOR->CSV (DIRECT)



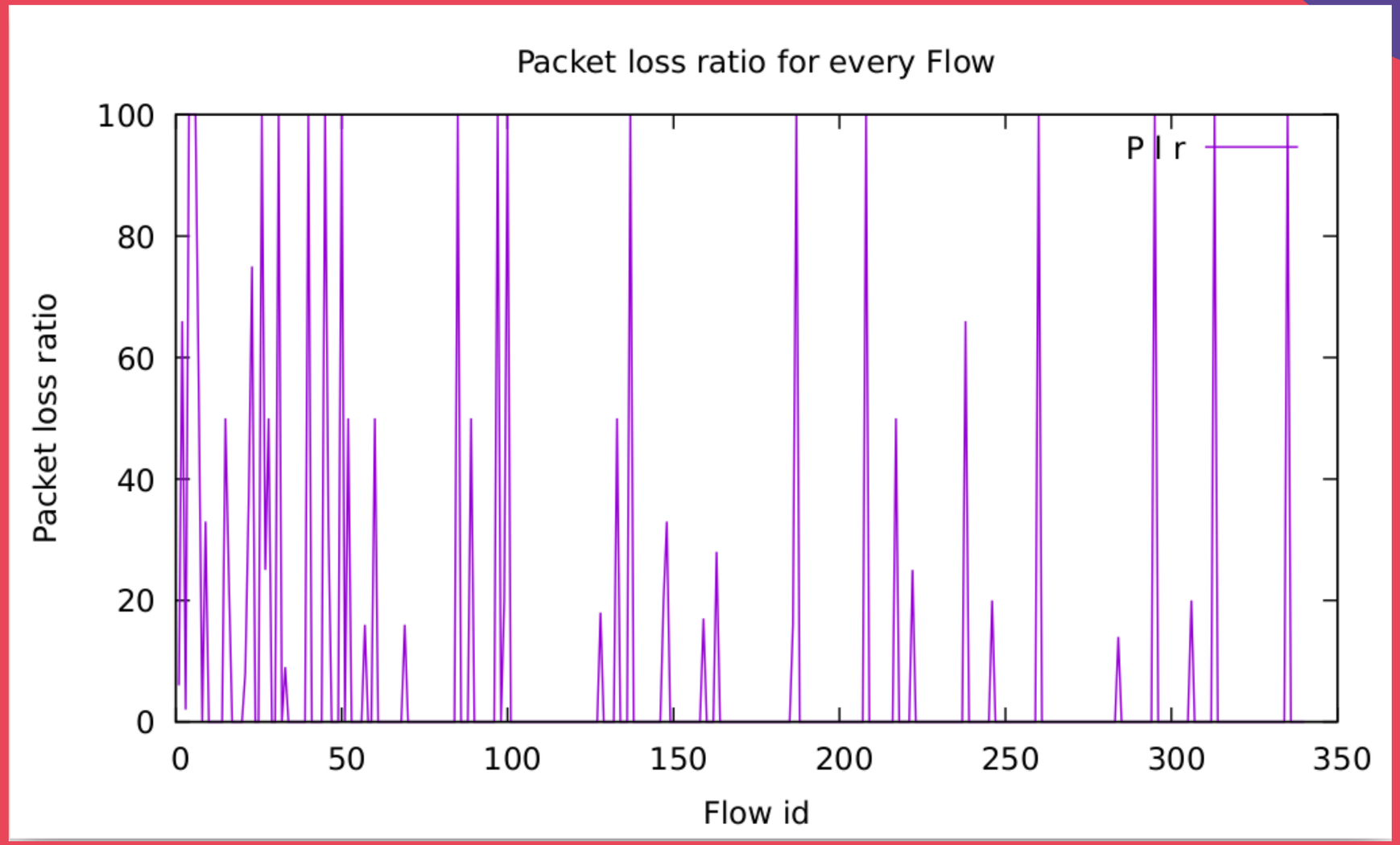
```
1 10.1.1.15 10.1.1.5 93 6 +8.58095e+07ns 2.72736
2 10.1.1.11 10.1.1.1 33 66 +7.38893e+09ns 1.01
3 10.1.1.14 10.1.1.4 97 2 +5.93669e+08ns 2.83753
4 10.1.1.13 10.1.1.3 0 100 +0ns -0
5 10.1.1.20 10.1.1.10 0 100 +0ns -0
6 10.1.1.19 10.1.1.9 0 100 +0ns -0
7 10.1.1.12 10.1.1.2 45 54 +4.38052e+09ns 1.32369
8 10.1.1.22 10.1.1.18 100 0 +1.45971e+07ns 0.365362
9 10.1.1.22 10.1.1.4 66 33 +2.89035e+07ns 0.231086
10 10.1.1.39 10.1.1.18 100 0 +4.23632e+08ns 0.129127
11 10.1.1.39 10.1.1.4 100 0 +2.86782e+07ns 13.0762
12 10.1.1.8 10.1.1.18 100 0 +1.61625e+06ns 232.018
13 10.1.1.8 10.1.1.4 100 0 +5.10573e+07ns 0.278839
14 10.1.1.25 10.1.1.18 100 0 +6.84905e+06ns 54.7521
15 10.1.1.25 10.1.1.4 50 50 +2.14054e+07ns 0.146358
16 10.1.1.18 10.1.1.8 75 25 +1.64734e+07ns 0.118874
17 10.1.1.18 10.1.1.42 100 0 +4.86475e+07ns 0.20539
18 10.1.1.18 10.1.1.25 100 0 +1.61523e+07ns 14.5103
19 10.1.1.18 10.1.1.39 100 0 +2.38162e+07ns 0.226344
20 10.1.1.18 10.1.1.22 100 0 +2.95037e+07ns 0.185076
21 10.1.1.42 10.1.1.14 91 8 +4.06521e+07ns 0.239351
22 10.1.1.17 10.1.1.7 62 37 +1.78072e+10ns 1.81348
23 10.1.1.16 10.1.1.6 25 75 +6.32271e+08ns 0.72819
24 10.1.1.18 10.1.1.8 100 0 +9.34556e+07ns 2.91328
25 10.1.1.6 10.1.1.29 100 0 +5.86442e+06ns 0.0830036
26 10.1.1.6 10.1.1.10 0 100 +0ns -0
27 10.1.1.29 10.1.1.6 75 25 +7.59545e+06ns 0.463944
28 10.1.1.29 10.1.1.32 50 50 +2.8148e+06ns 133.224
29 10.1.1.32 10.1.1.37 100 0 +4.15579e+06ns 90.2356
30 10.1.1.27 10.1.1.33 100 0 +1.83317e+07ns 0.681616
31 10.1.1.27 10.1.1.3 0 100 +0ns -0
32 10.1.1.37 10.1.1.8 100 0 +7.43599e+06ns 50.4304
33 10.1.1.33 10.1.1.27 90 9 +1.2984e+07ns 0.528133
34 10.1.1.33 10.1.1.24 100 0 +5.62716e+06ns 66.6411
35 10.1.1.24 10.1.1.45 100 0 +6.41627e+06ns 58.4452
36 10.1.1.8 10.1.1.20 100 0 +8.29632e+06ns 45.2008
37 10.1.1.45 10.1.1.11 100 0 +1.09333e+07ns 0.14775
```

- FLOW ID
- SRC DST
- DROP RATIO
- LOSS RATIO
- DELAY

OLDMEAD SECONDARY SCHOOL

SCIENCE CLASS

FLOWMOINTOR->CSV->GNU PLOT



OLDMEAD SECONDARY SCHOOL

SCIENCE CLASS

6.4 Packet Drop Ratio

It is the ratio of the data lo st at destination to those generated by the CBR sources. The packets are dropped when the node is not able to find the valid route to the node specified as an intermediate node in the route to reach the destination node.

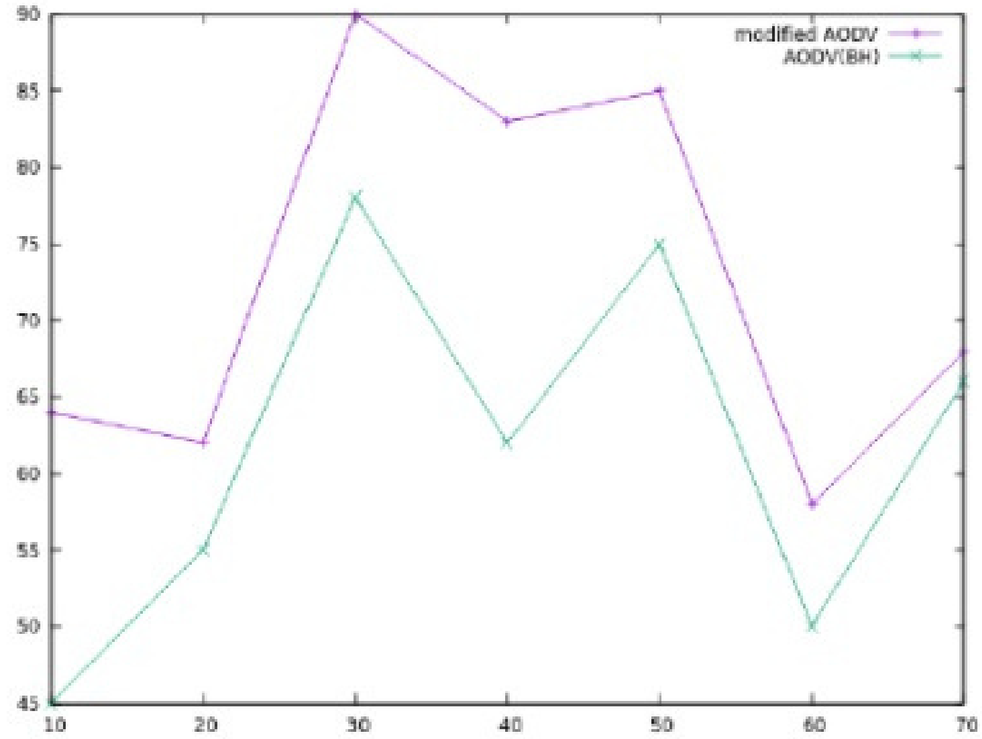
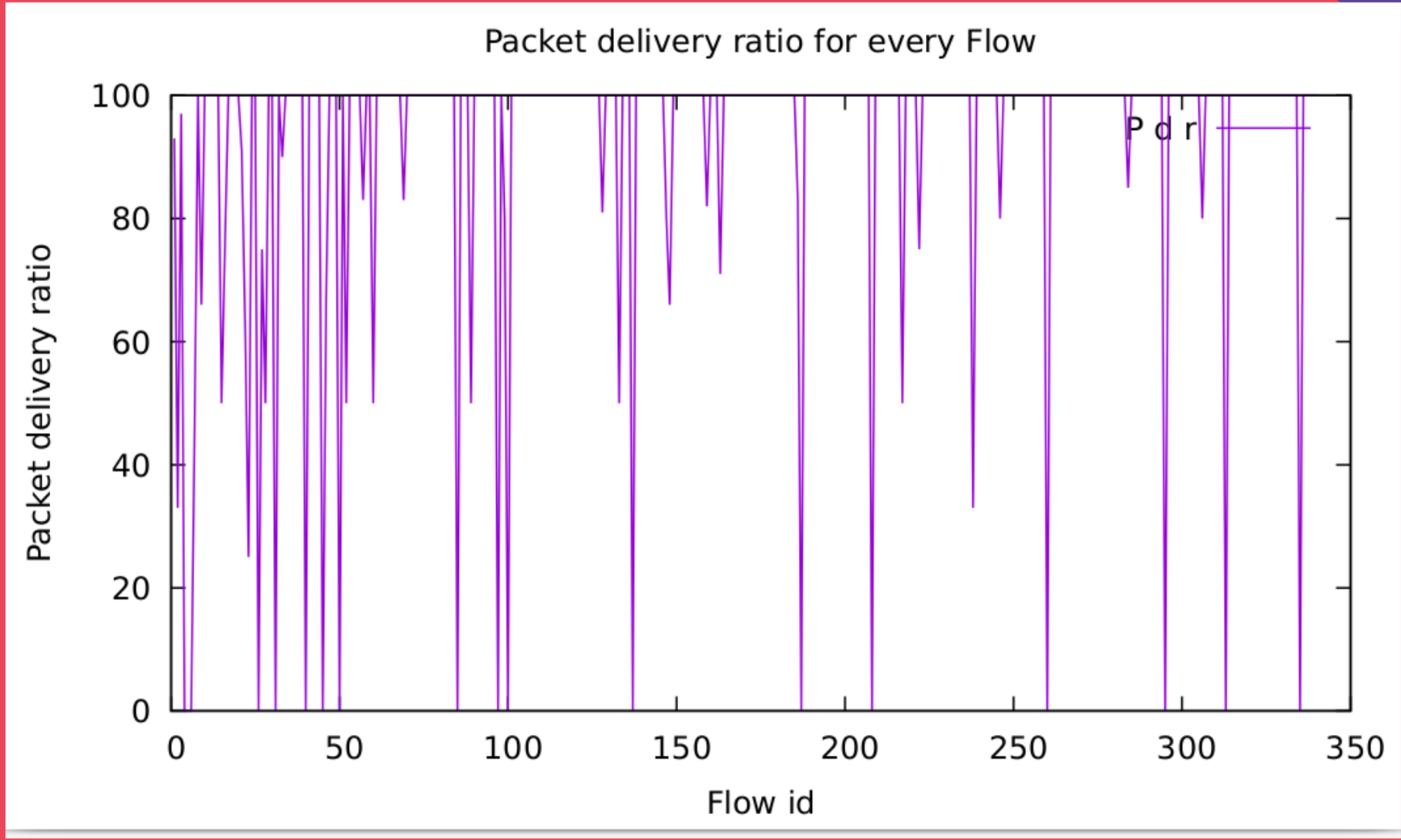


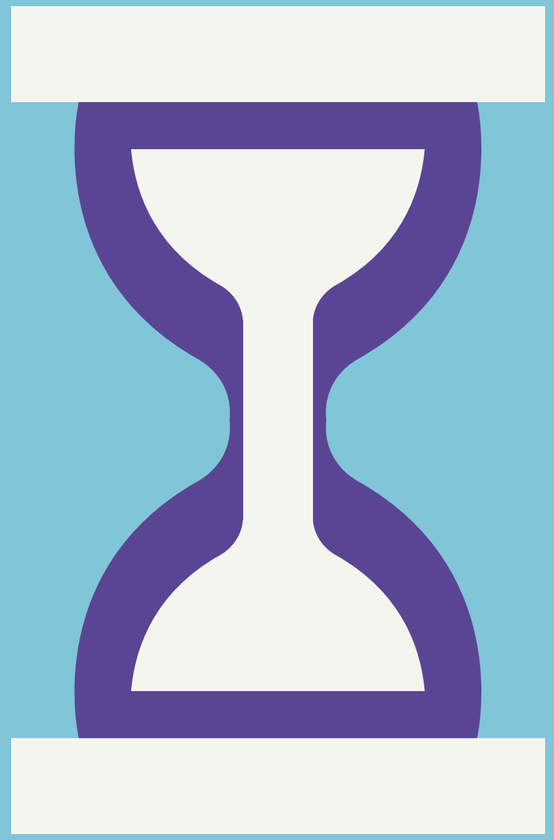
Fig 6: Packet Drop Ratio vs No. of Nodes

FLOWMOINTOR->CSV->GNU PLOT



OLDMEAD SECONDARY SCHOOL

SCIENCE CLASS



6.3 Average End-to-End delay

It is the average delay between the sending of packets by the source and its receipt by the receiver.

Fig..4. shows that the Average End to End Delay with black hole attack is much higher than without black hole attack in AODV routing protocol.

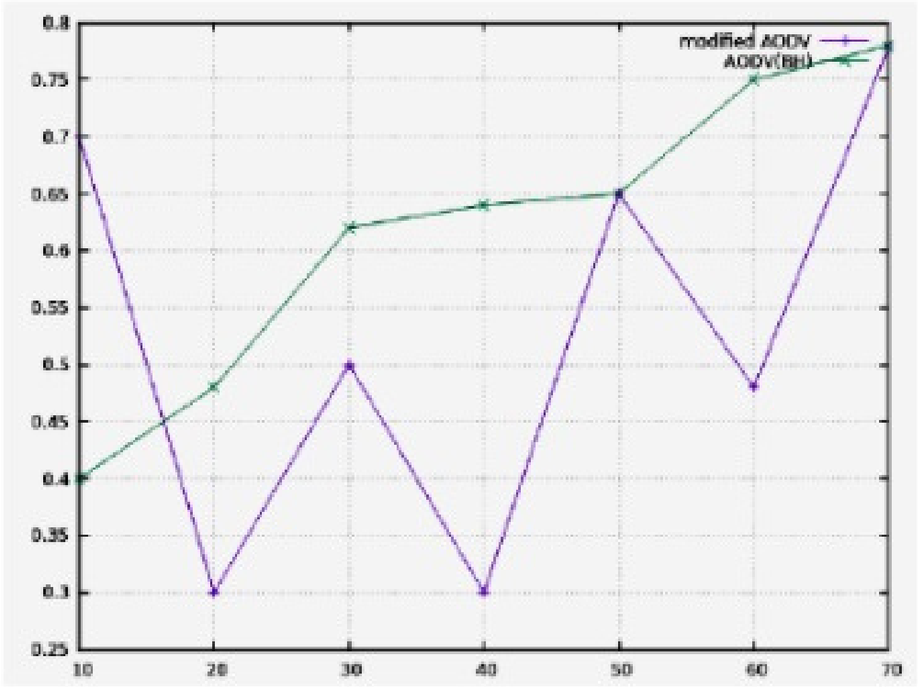
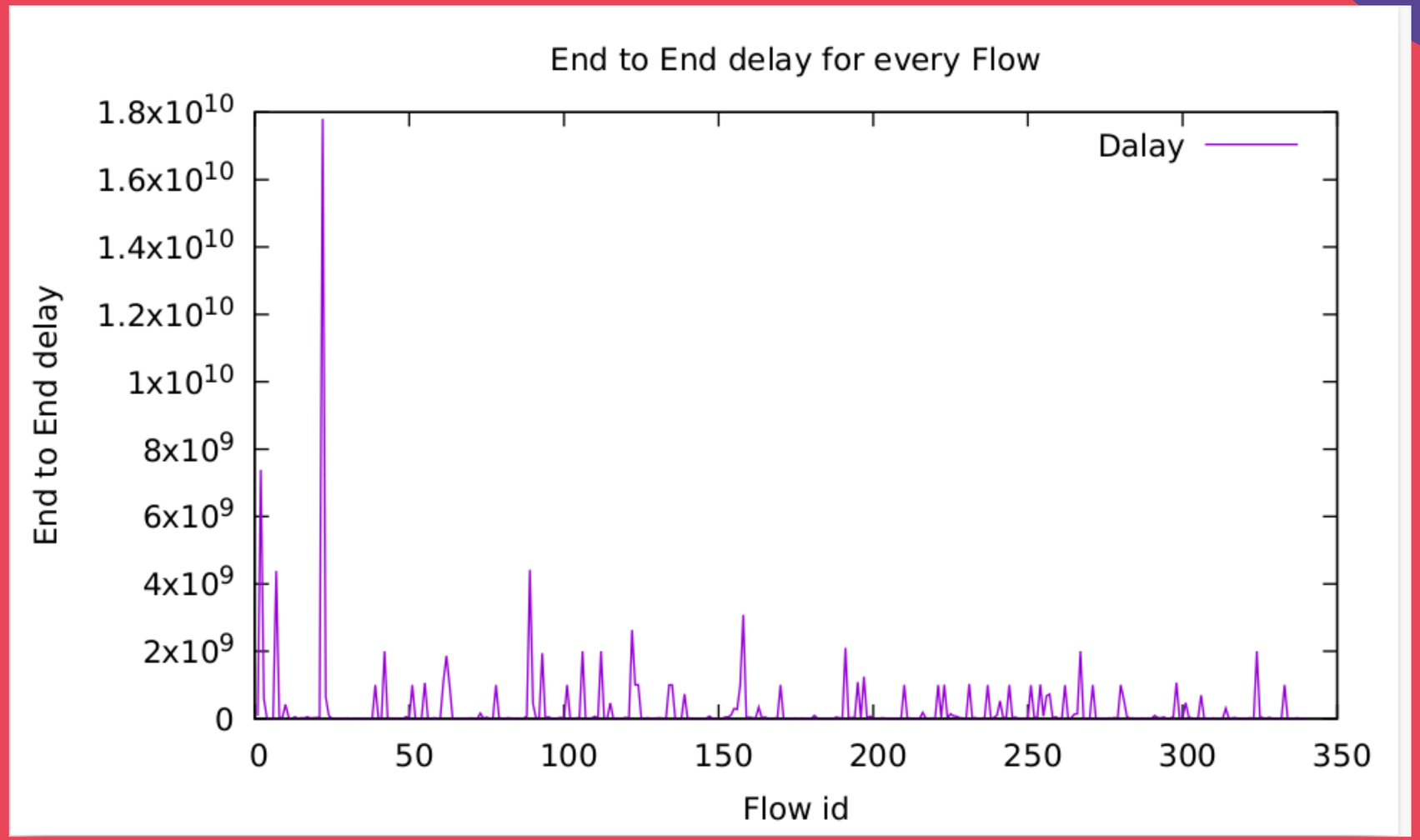


Fig 5: Average End to End Delay vs Number of Nodes

FLOWMOINTOR->CSV->GNU PLOT



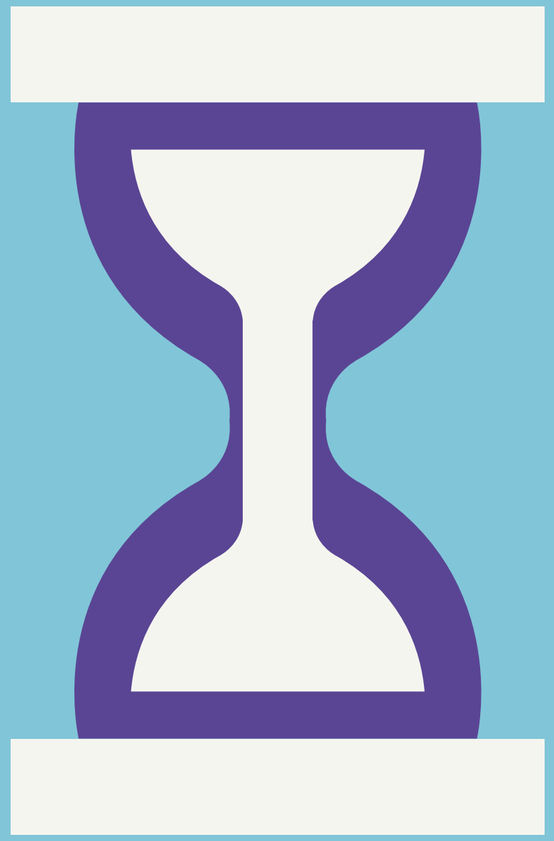
OLDMEAD SECONDARY SCHOOL

SCIENCE CLASS

HOW WE CODE FLOW (SINK +APP+ NODE)

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

OnOffHelper onoff1 ("ns3::UdpSocketFactory",Address ());
onoff1.SetAttribute ("OnTime", StringValue ("ns3::ConstantRandomVariable[Constant=1.0]"));
onoff1.SetAttribute ("OffTime", StringValue ("ns3::ConstantRandomVariable[Constant=0.0]"));

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<UniformRandomVariable> var = CreateObject<UniformRandomVariable> ();
    ApplicationContainer temp = onoff1.Install (adhocNodes.Get (i + nSinks));
    temp.Start (Seconds (var->GetValue (97.0,98.0)));
    temp.Stop (Seconds (TotalTime));
  }
```

FLOW MON EXTRA

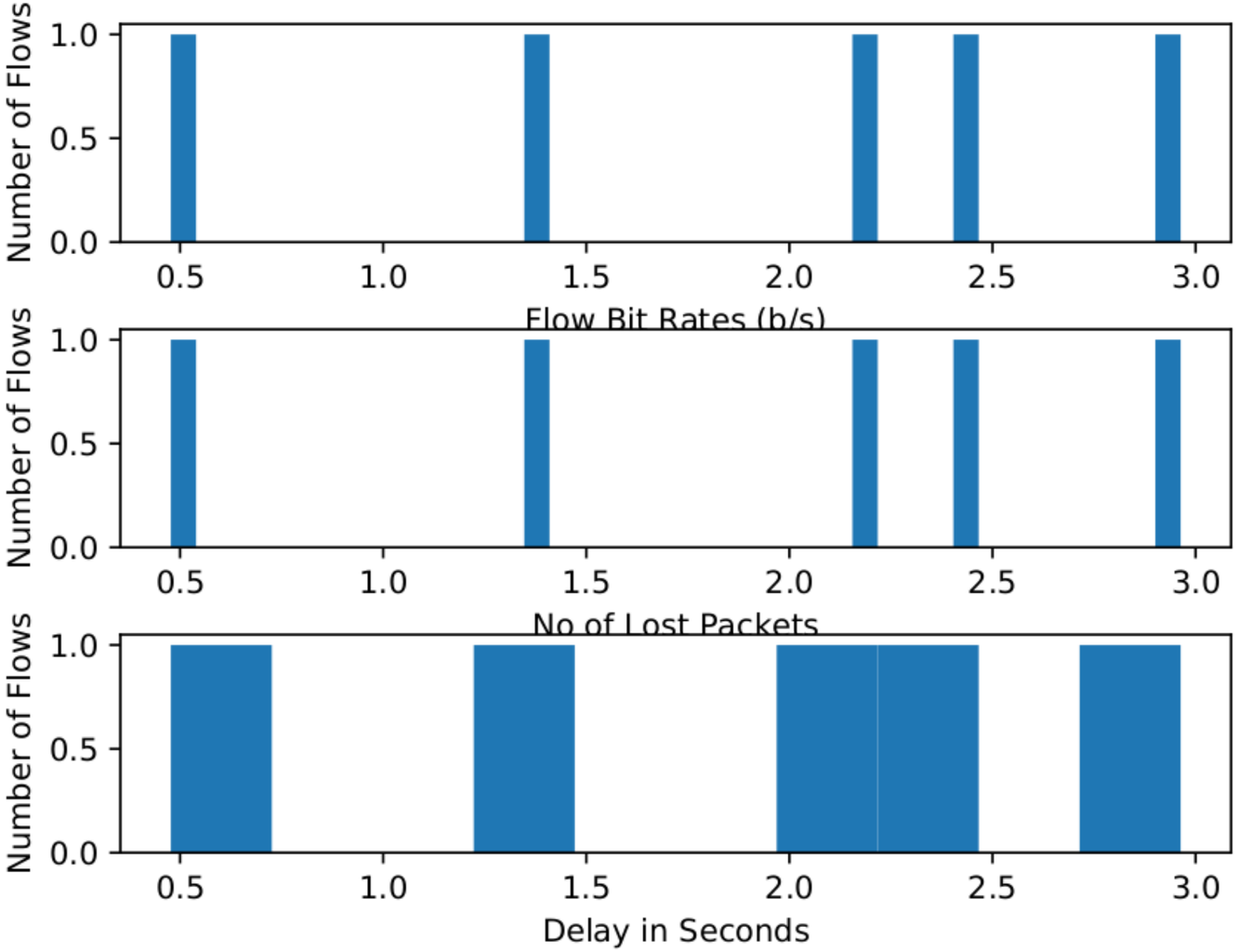


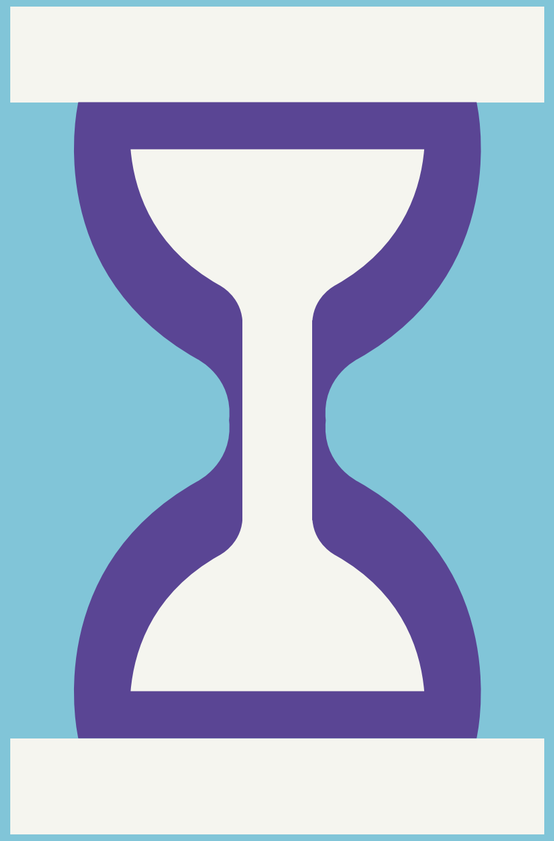
```python
from xml.etree import ElementTree as ET
import sys
import matplotlib.pyplot as pylab
et=ET.parse(sys.argv[1])
bitrates=[]
losses=[]
delays=[]
for flow in et.findall("FlowStats/Flow"):
 for tpl in et.findall("Ipv4FlowClassifier/Flow"):
  if tpl.get('flowId')==flow.get('flowId'):
   break
 if tpl.get('destinationPort')=='654':
  continue
 losses.append(int(flow.get('lostPackets')))

 rxPackets=int(flow.get('rxPackets'))
 if rxPackets==0:
  bitrates.append(0)
 else:
  t0=float(flow.get('timeFirstRxPacket')[:-2])
  t1=float(flow.get("timeLastRxPacket")[:-2])
  duration=(t1-t0)*1e-9
  bitrates.append(8*int(flow.get("rxBytes"))/duration*1e-3)
  delays.append(float(flow.get('delaySum')[:-2])*1e-9/rxPackets)

pylab.subplot(311)
pylab.hist(bitrates,bins=40)
pylab.xlabel("Flow Bit Rates (b/s)")
pylab.ylabel("Number of Flows")


pylab.subplot(312)
pylab.hist(bitrates,bins=40)
pylab.xlabel("No of Lost Packets")
pylab.ylabel("Number of Flows")
```

OLDMEAD SECONDARY SCHOOL

SCIENCE CLASS

XML->PYTHON->MATLAB PLOT





FLOW MON TOTAL

```
--------Total Results of the simulation---

Total sent packets  =2123
Total Received Packets =1605
Total Lost Packets =518
Packet Loss ratio =24%
Packet delivery ratio =75%
Average Throughput =31.8937Kbps
End to End Delay =+9.79147e+10ns
```

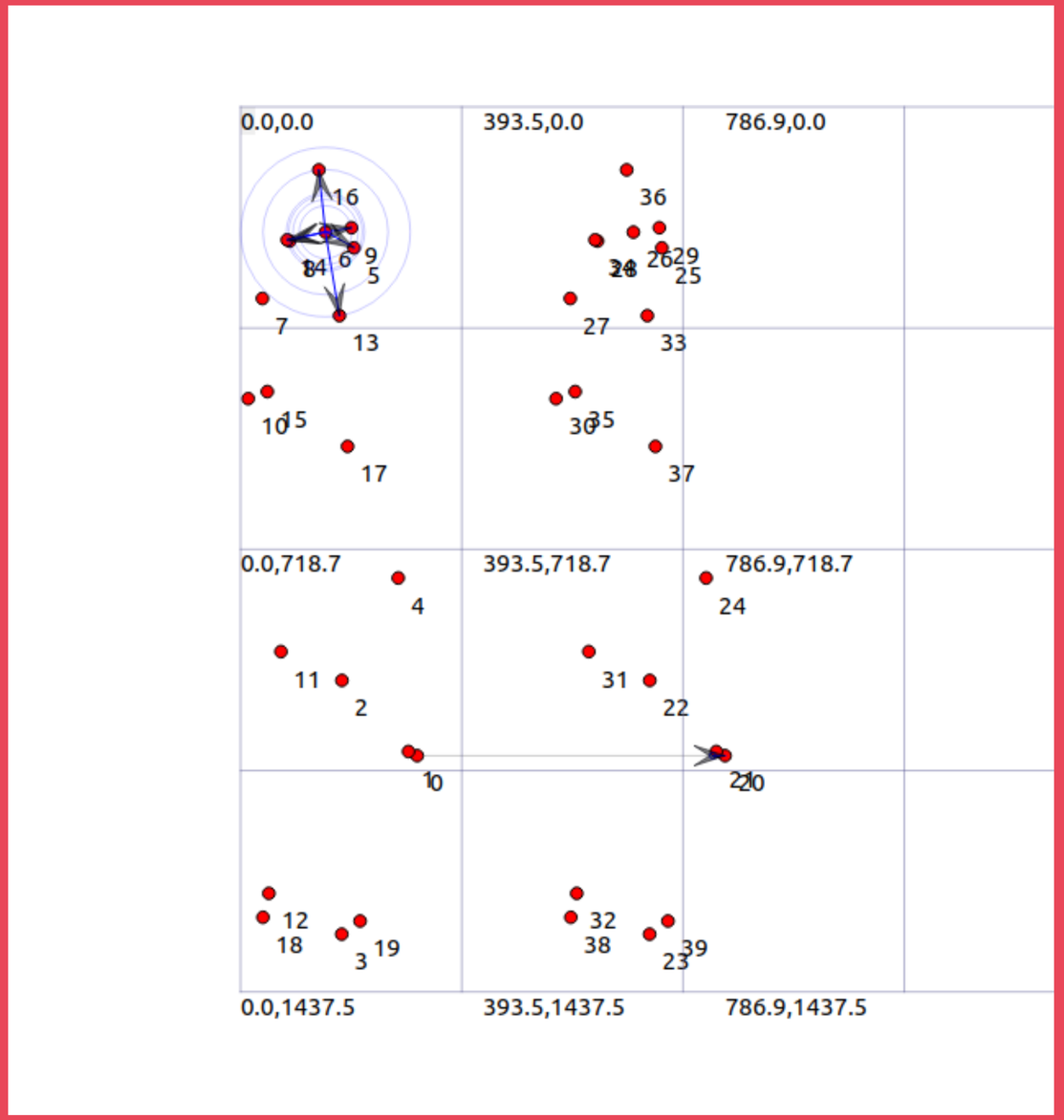
OLDMEAD SECONDARY SCHOOL

SCIENCE CLASS

2 MANET COONECTED -NET IP

```
NS_LOG_INFO ("assigning ip address");



Ipv4AddressHelper addressAdhoc;

addressAdhoc.SetBase ("10.1.3.0", "255.255.255.0");
Ipv4InterfaceContainer p2pInterfaces;
p2pInterfaces = addressAdhoc.Assign (p2pDevices);


addressAdhoc.SetBase ("10.1.1.0", "255.255.255.0");
Ipv4InterfaceContainer adhocApInterfaces,adhocInterfaces;
adhocInterfaces = addressAdhoc.Assign (adhocDevices);
//adhocApInterfaces = addressAdhoc.Assign (wifiApDevices);

addressAdhoc.SetBase ("10.1.2.0", "255.255.255.0");
Ipv4InterfaceContainer adhocApInterfaces2,adhocInterfaces2;
adhocInterfaces2 = addressAdhoc.Assign (adhocDevices2);
//adhocApInterfaces2 = addressAdhoc.Assign (wifiApDevices2);
```

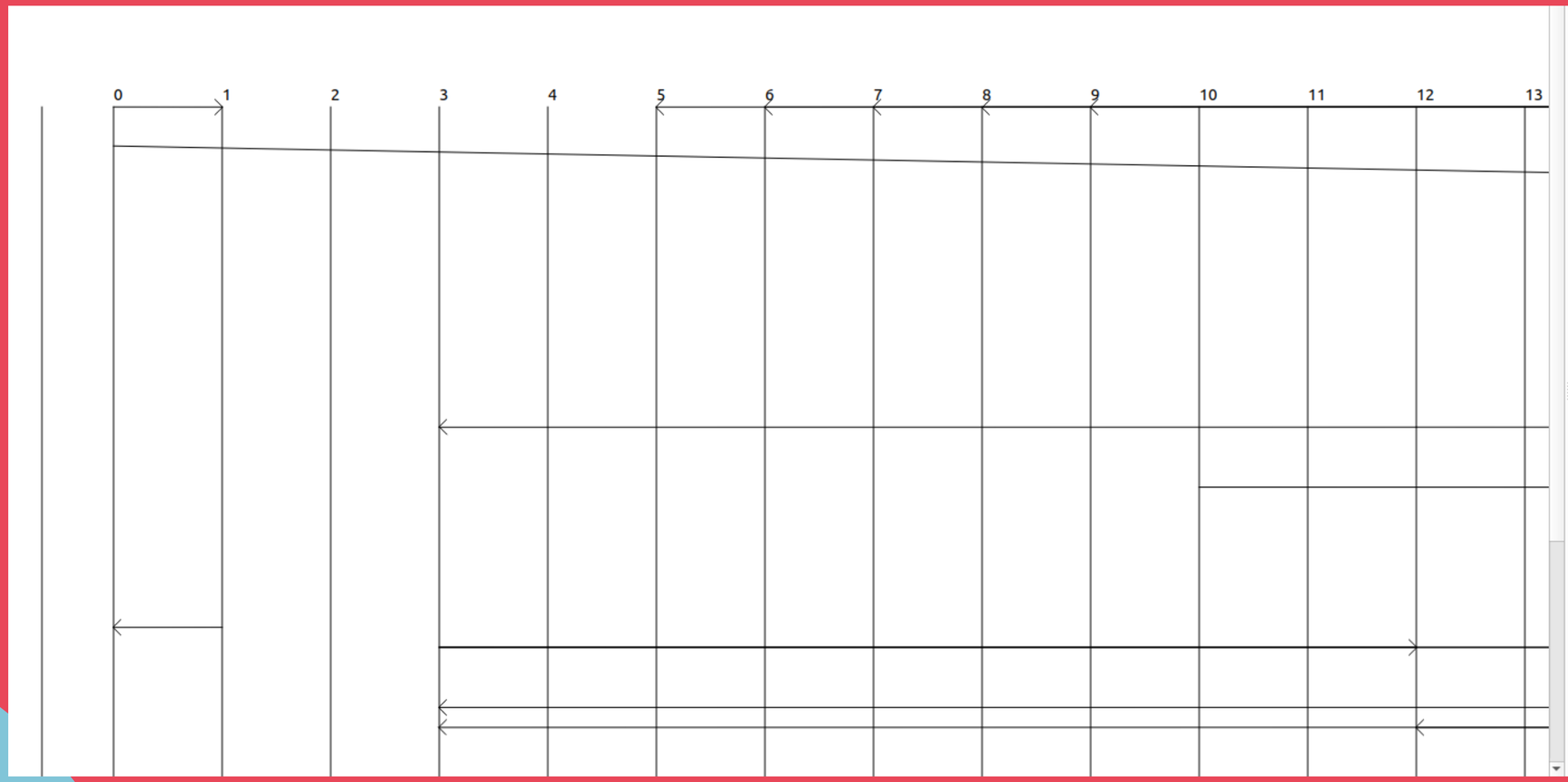
2 MANET COONECTED -NET ANIM



OLDMEAD SECONDARY SCHOOL

SCIENCE CLASS

2 MANET COONECTED -NET ANIM



Export Table

	From Id	To Id	Tx	Meta
1	14	8	0.00605	
2	0	1	0.00605	
3	14	6	0.00605	
4	14	7	0.00605	
5	14	9	0.00605	
6	14	5	0.00605	
7	14	16	0.00605	
8	0	20	0.008	
9	38	32	0.01105	
10	38	23	0.01105	
11	32	38	0.01805	
12	32	23	0.01805	
13	12	18	0.02205	
14	12	3	0.02205	
15	10	15	0.02505	
16	1	0	0.03205	
17	3	19	0.03305	
18	3	18	0.03305	
19	3	12	0.03305	
20	26	29	0.03405	
21	26	25	0.03405	
22	26	28	0.03405	
23	26	34	0.03405	
24	26	36	0.03405	
25	26	33	0.03405	
26	19	3	0.03605	
27	18	12	0.03705	
28	18	3	0.03705	
29	35	30	0.03905	
30	6	9	0.04005	
31	6	5	0.04005	
32	6	8	0.04005	

OLDMEAD SECONDARY SCHOOL

SCIENCE CLASS

SOFTWARE / LANGUAGE WE USED

NET-ANIM

WIRESRK

TRACEME
TRICS

GNU PLOT

FLOW
MONITOR

PYTHON

OLDMEAD SECONDARY SCHOOL

SCIENCE CLASS

THANK YOU !! ☺

OLDMEAD SECONDARY SCHOOL

SCIENCE CLASS