Configuration Manual

Academic Internship
MSc Cyber Security

Uppili Srinivasa Raghavan
Student ID:X18133312

School of Computing
National College of Ireland

Supervisor: Christos Grecos

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Uppili Srinivasa Raghavan
Student ID:	X18133312
Programme:	MSc Cyber Security
Year:	2019
Module:	Academic Internship
Supervisor:	Christos Grecos
Submission Due Date:	12/12/2019
Project Title:	Configuration Manual
Word Count:	457
Page Count:	8

Signature:	
Date:	12th December 2019

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

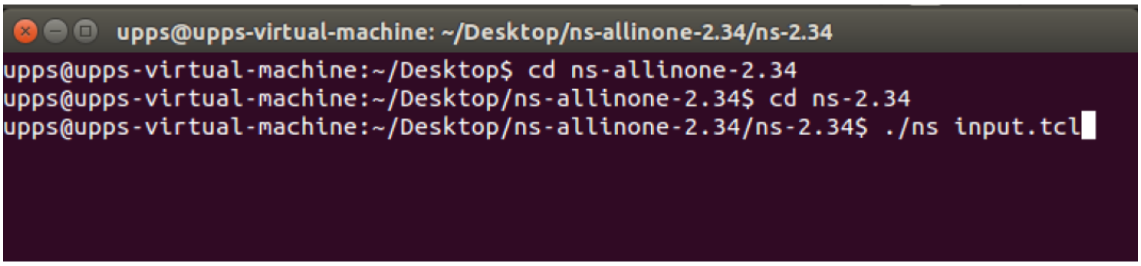
Uppili Srinivasa Raghavan
X1813331

1 Installation of NS 2.34 in Ubuntu 14.04

- Download ns-allinone-2.34.tar.gz
- Extract the tar file in the desktop.
- Install the basic packages necessary for installation use following commands in the terminal:
  - *sudo apt-get update*
  - *sudo apt-get install gcc build-essential autoconf automake tcl8.5-dev tk8.5-dev perl xgraph libxt-dev libx11-dev libxmu-dev*
- In order to install NS 2.34 go to ns-allinone-2.34 using following commands:
  - *cd ns-allinone-2.34*
  - *./install*
- All the necessary packages are installed and NS 2 is ready to run.

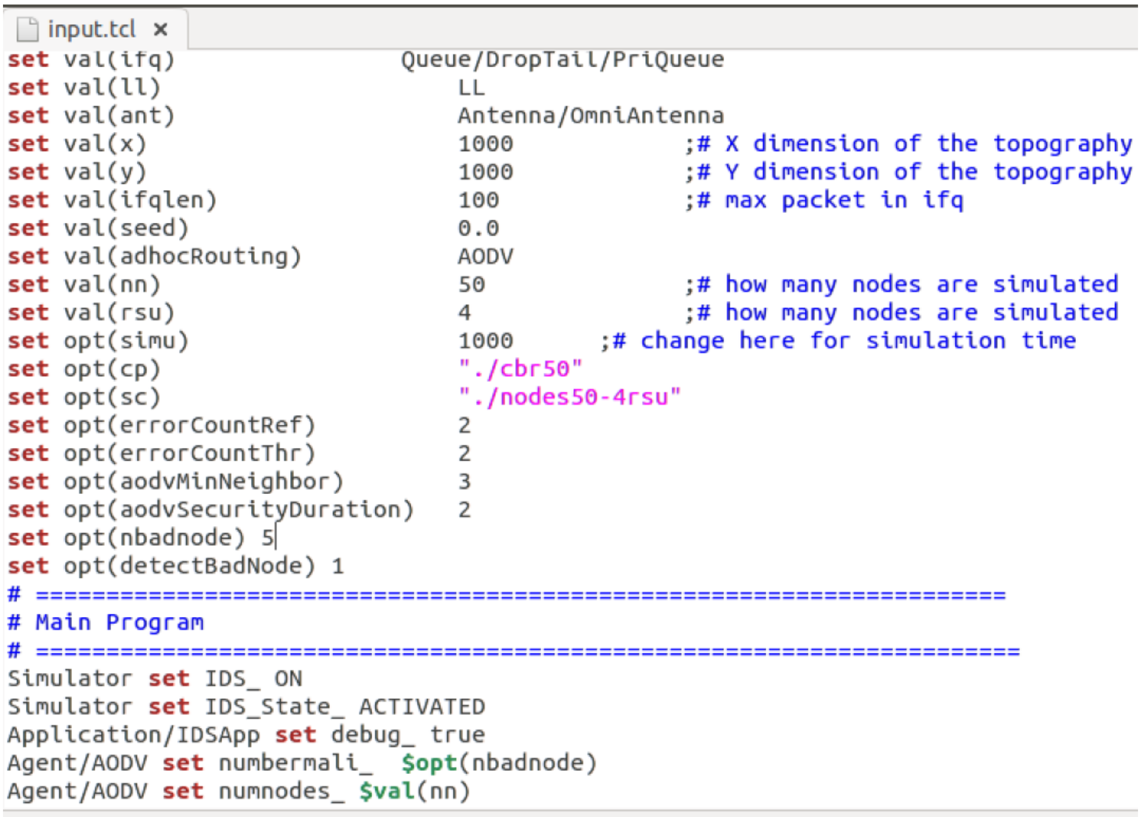
2 How to run the scenario file

- After successful installation we can run the scenario file.
- Go to folder ns-2.34 using the command:
  - *cd ns-2.34*
- The scenario file input.tcl can be executed using the following command:
  - *./ns input.tcl*


upps@upps-virtual-machine: ~/Desktop/ns-allinone-2.34/ns-2.34

```
upps@upps-virtual-machine:~/Desktop$ cd ns-allinone-2.34
upps@upps-virtual-machine:~/Desktop/ns-allinone-2.34$ cd ns-2.34
upps@upps-virtual-machine:~/Desktop/ns-allinone-2.34/ns-2.34$ ./ns input.tcl
```

Figure 1: Scenario file(input.tcl)



```
set val(ifq)                 Queue/DropTail/PriQueue
set val(ll)                      LL
set val(ant)                     Antenna/OmniAntenna
set val(x)                       1000            ;# X dimension of the topography
set val(y)                       1000            ;# Y dimension of the topography
set val(ifqlen)                  100             ;# max packet in ifq
set val(seed)                    0.0
set val(adhocRouting)            AODV
set val(nn)                      50              ;# how many nodes are simulated
set val(rsu)                     4               ;# how many nodes are simulated
set opt(simu)                    1000      ;# change here for simulation time
set opt(cp)                      "./cbr50"
set opt(sc)                      "./nodes50-4rsu"
set opt(errorCountRef)           2
set opt(errorCountThr)           2
set opt(aodvMinNeighbor)         3
set opt(aodvSecurityDuration)    2
set opt(nbadnode) 5
set opt(detectBadNode) 1
# ==================================================================
# Main Program
# ===================================================================
Simulator set IDS_ ON
Simulator set IDS_State_ ACTIVATED
Application/IDSApp set debug_ true
Agent/AODV set numbermali_  $opt(nbadnode)
Agent/AODV set numnodes_ $val(nn)
```

Figure 2: executing input.tcl

- After the successful execution of the scenario file the Network animator(NAM) and the graphs pop out.
- It will also create two outputs output.tr,output.nam.

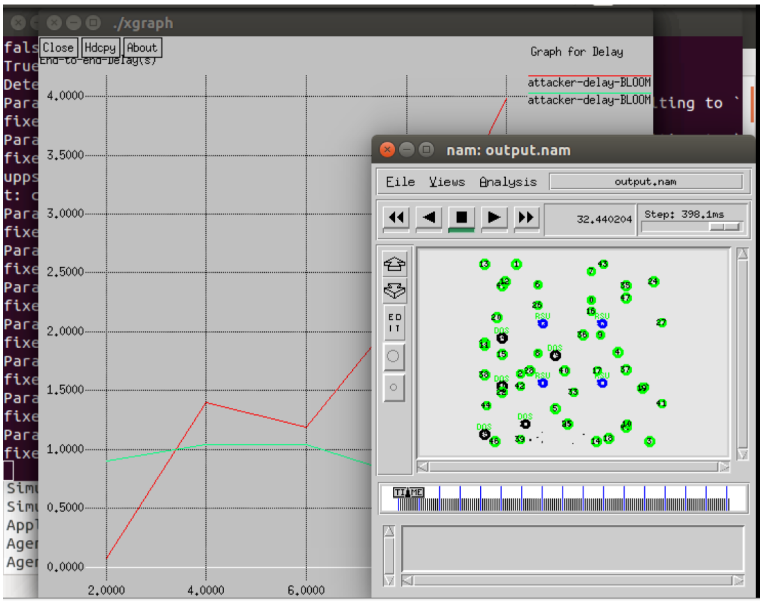


Figure 3: Results

3 Simulation results

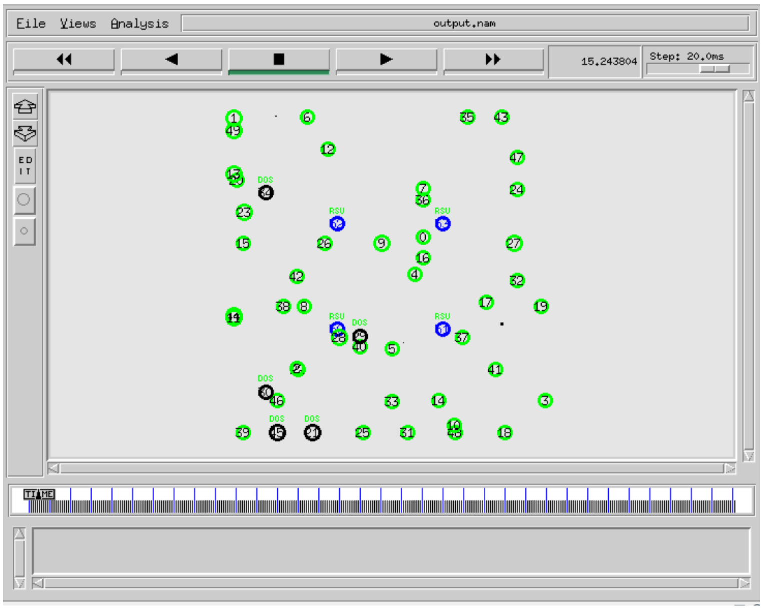
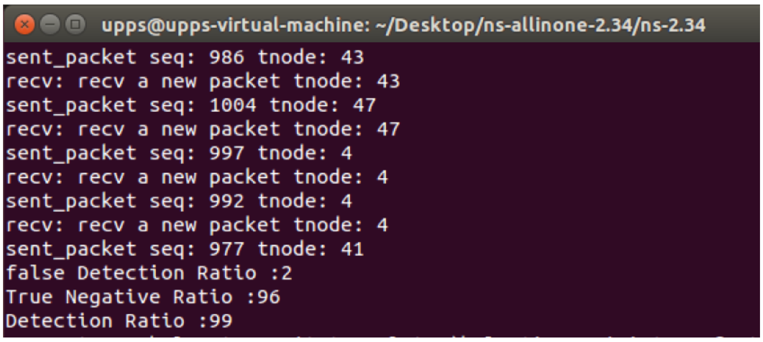


Figure 4: Simulation Results

4 Detection ratio and False positive ratio

The Detection ratio and the False positive rate are calculated during the execution the scenario file.


upps@upps-virtual-machine: ~/Desktop/ns-allinone-2.34/ns-2.34

```
sent_packet seq: 986 tnode: 43
recv: recv a new packet tnode: 43
sent_packet seq: 1004 tnode: 47
recv: recv a new packet tnode: 47
sent_packet seq: 997 tnode: 4
recv: recv a new packet tnode: 4
sent_packet seq: 992 tnode: 4
recv: recv a new packet tnode: 4
sent_packet seq: 977 tnode: 41
false Detection Ratio :2
True Negative Ratio :96
Detection Ratio :99
```

Figure 5: DR and FPR Results

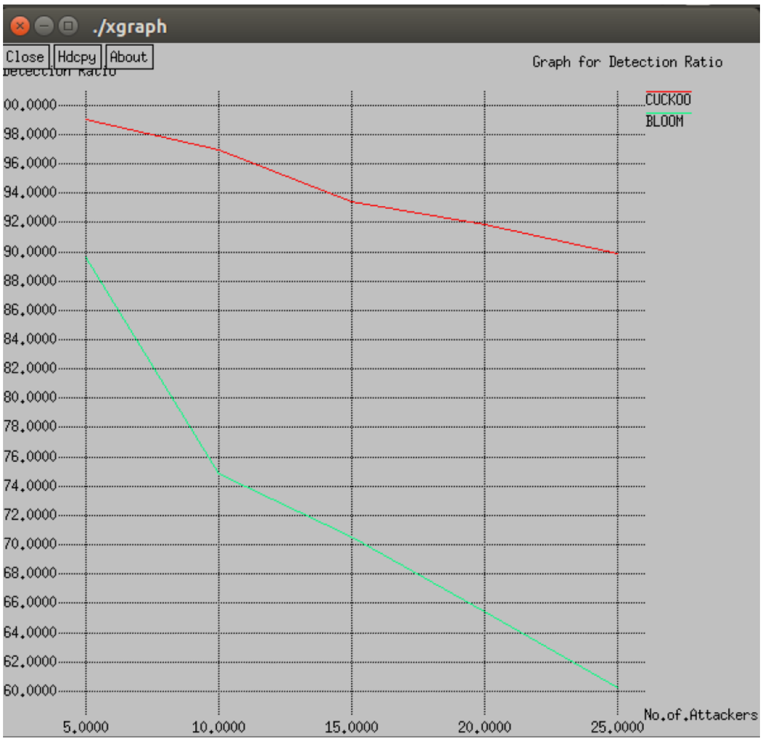


Figure 6: Detection Ratio

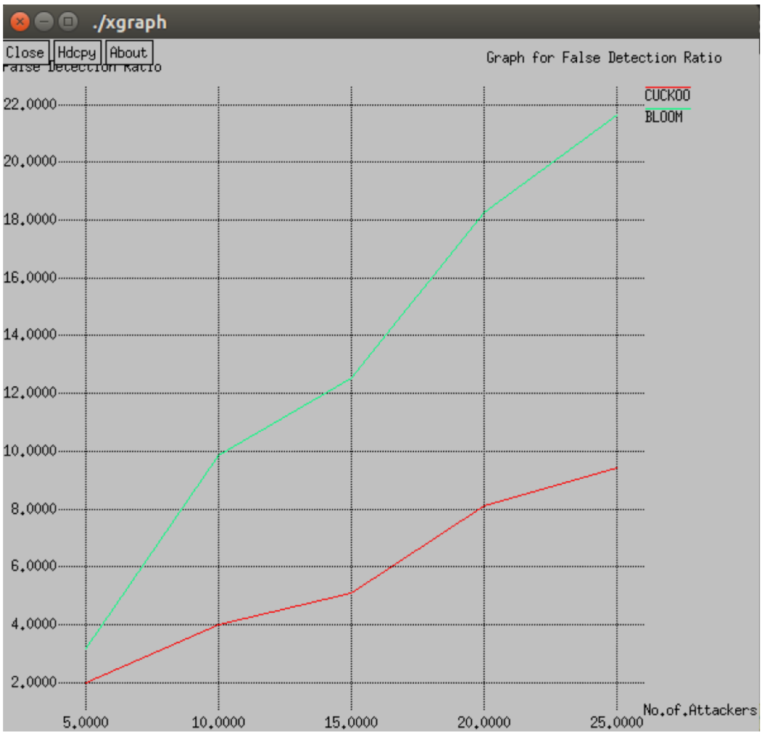
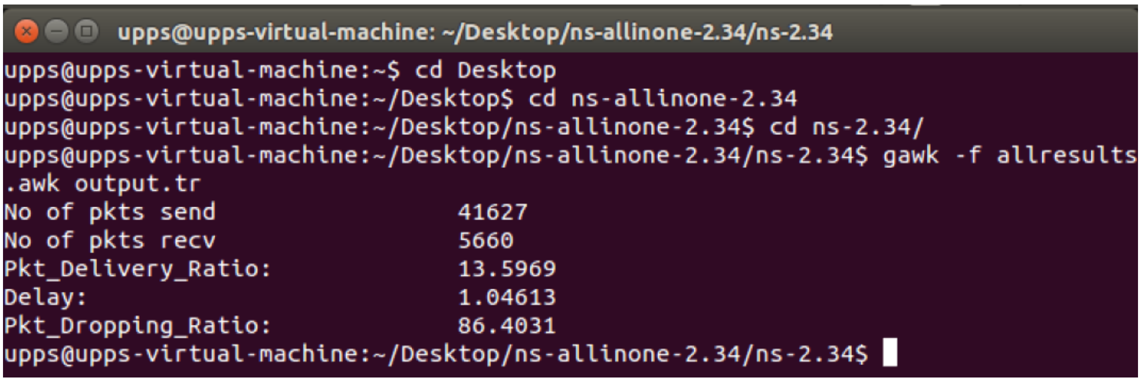


Figure 7: False Positive Ratio

5 Packet Delivery Ratio, Packet-Loss Ratio, End-to-end Delay

The AWK scripts are used to calculate the Packet delivery and packet-loss ratio and End-to-end delay.

- Go to the folder ns-2.34 where you will find allresults.awk using command:
  - *cd ns-2.34*
- To execute the awk scripts use the following command:
  - *gawk -f allresults.awk output.tr*


upps@upps-virtual-machine: ~/Desktop/ns-allinone-2.34/ns-2.34

```
upps@upps-virtual-machine:~$ cd Desktop
upps@upps-virtual-machine:~/Desktop$ cd ns-allinone-2.34
upps@upps-virtual-machine:~/Desktop/ns-allinone-2.34$ cd ns-2.34/
upps@upps-virtual-machine:~/Desktop/ns-allinone-2.34/ns-2.34$ gawk -f allresults
.awk output.tr
No of pkts send                 41627
No of pkts recv                 5660
Pkt_Delivery_Ratio:             13.5969
Delay:                          1.04613
Pkt_Dropping_Ratio:             86.4031
upps@upps-virtual-machine:~/Desktop/ns-allinone-2.34/ns-2.34$ 
```

Figure 8: awk script execution

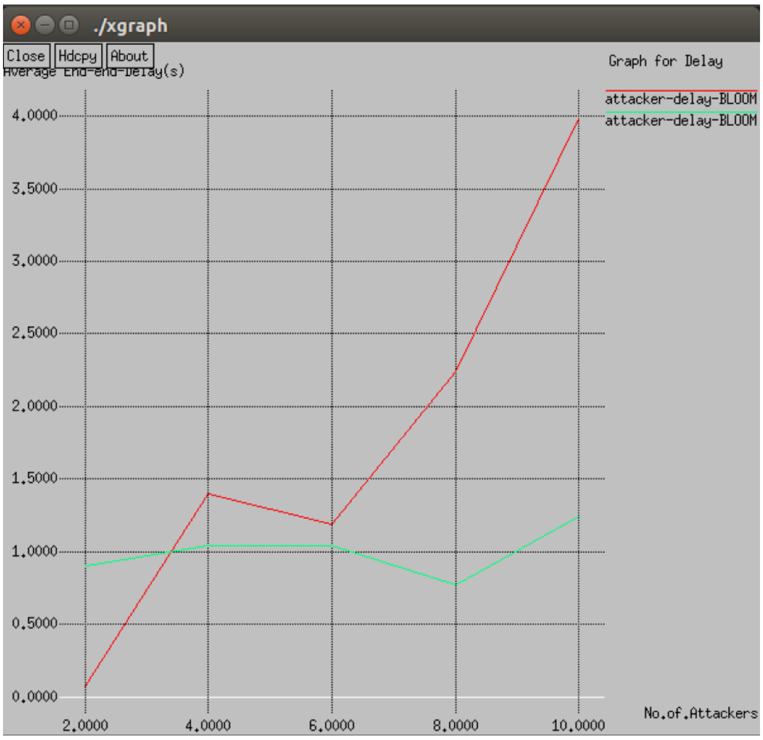


Figure 9: Delay

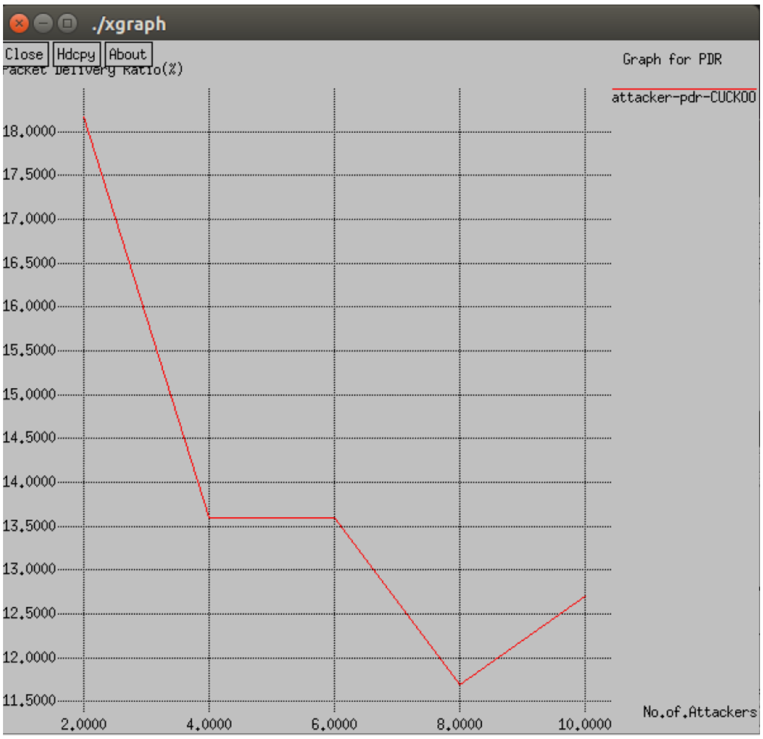


Figure 10: Packet delivery ratio

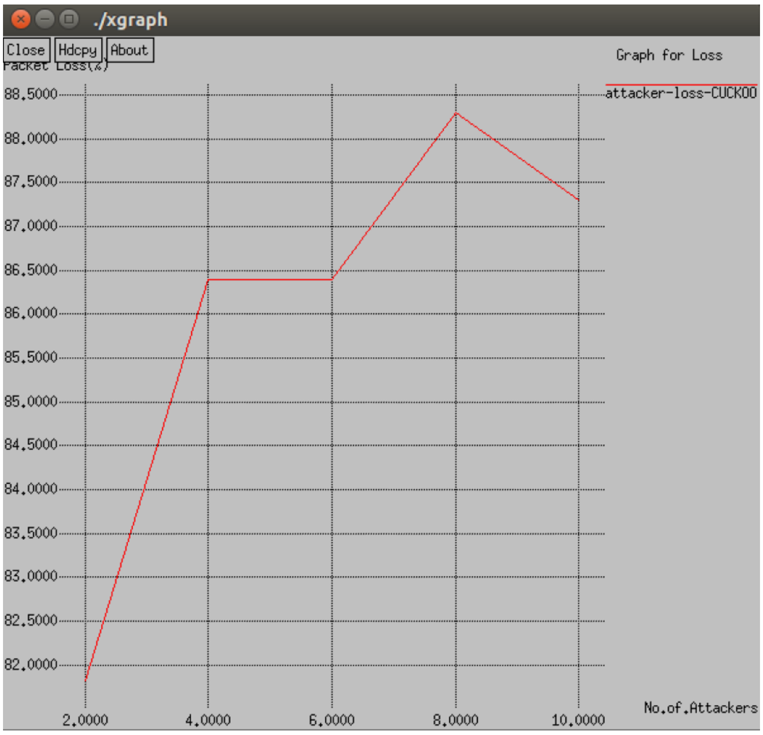
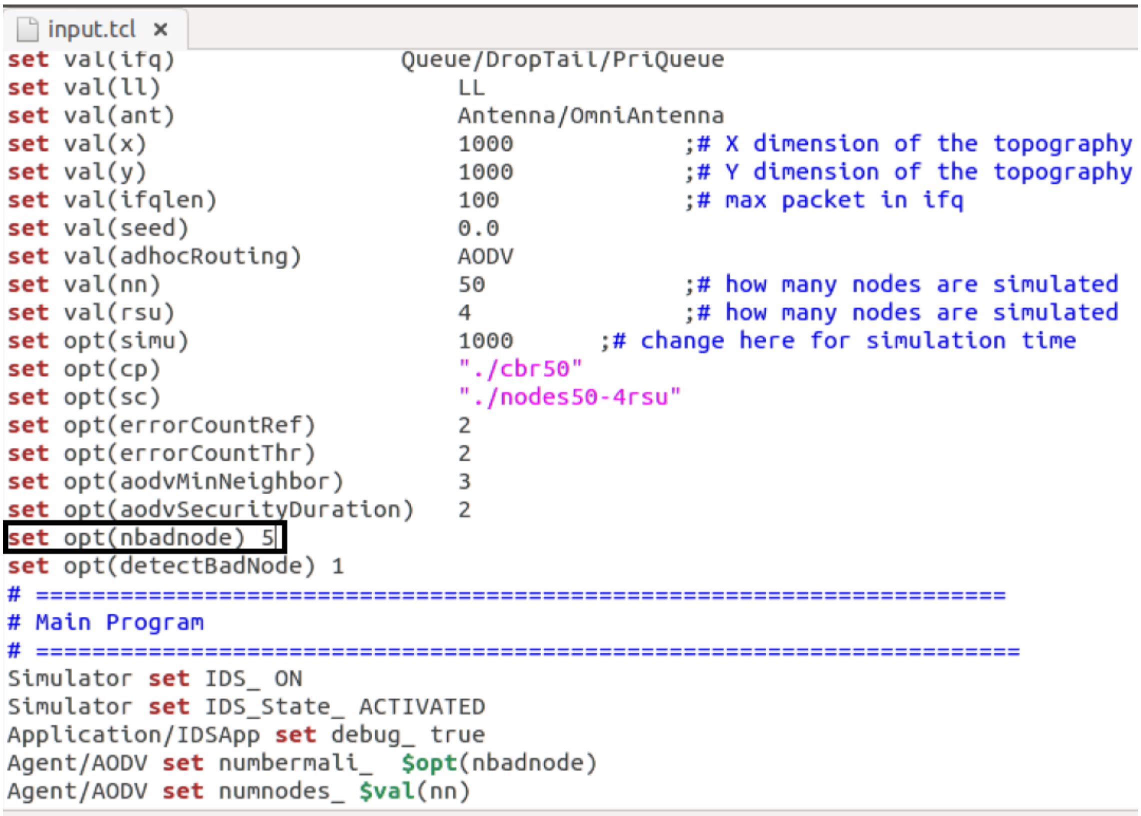


Figure 11: Packet-loss ratio

6 How to vary the scenario

- For the purpose of generating graphs we try out different scenarios by varying the number of malicious nodes in the network.
- This is done in the scenario file input.tcl.



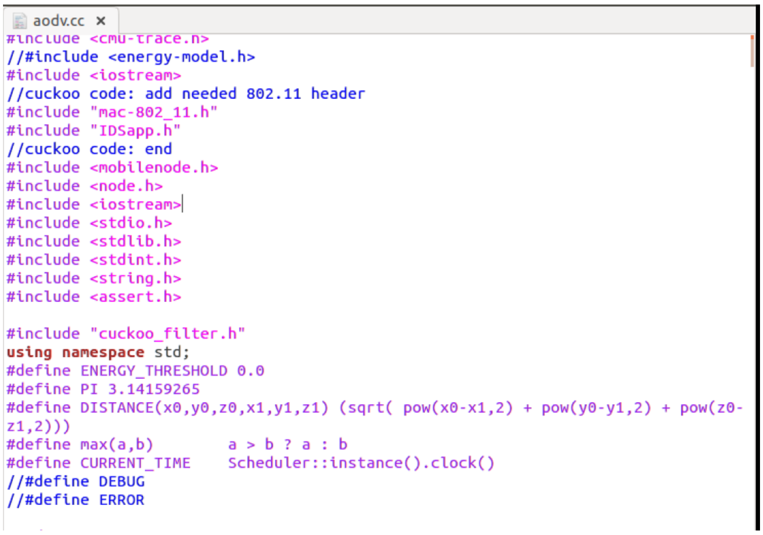
```
set val(ifq)                    Queue/DropTail/PriQueue
set val(ll)                         LL
set val(ant)                        Antenna/OmniAntenna
set val(x)                          1000            ;# X dimension of the topography
set val(y)                          1000            ;# Y dimension of the topography
set val(ifqlen)                     100             ;# max packet in ifq
set val(seed)                       0.0
set val(adhocRouting)               AODV
set val(nn)                         50              ;# how many nodes are simulated
set val(rsu)                        4               ;# how many nodes are simulated
set opt(simu)                       1000      ;# change here for simulation time
set opt(cp)                         "./cbr50"
set opt(sc)                         "./nodes50-4rsu"
set opt(errorCountRef)              2
set opt(errorCountThr)              2
set opt(aodvMinNeighbor)            3
set opt(aodvSecurityDuration)       2
set opt(nbadnode) 5
set opt(detectBadNode) 1
# =====================================================================
# Main Program
# ======================================================================
Simulator set IDS_ ON
Simulator set IDS_State_ ACTIVATED
Application/IDSApp set debug_ true
Agent/AODV set numbermali_  $opt(nbadnode)
Agent/AODV set numnodes_ $val(nn)
```

Figure 12: variation in scenario

- The highlighted field must be varied to get different values and graph performance.
- In our case we have varied the number of malicious nodes from 5 to 25 for Detection ratio,False-positive ratio and Delay.
- For Packet-delivery ratio and Packet-loss ratio we varied the the number of malicious nodes from 2 to 10.

7 Cuckoo code

The entire code for cuckoo is available in aodv.cc file which is can be seen in the aodv folder present in ns-2.34.



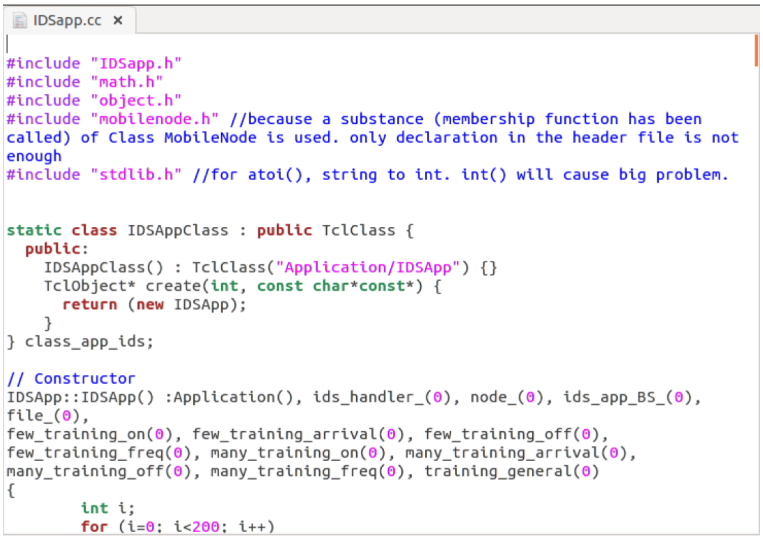
```
#include <cmu-trace.h>
//#include <energy-model.h>
#include <iostream>
//cuckoo code: add needed 802.11 header
#include "mac-802_11.h"
#include "IDSapp.h"
//cuckoo code: end
#include <mobilenode.h>
#include <node.h>
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include <string.h>
#include <assert.h>

#include "cuckoo_filter.h"
using namespace std;
#define ENERGY_THRESHOLD 0.0
#define PI 3.14159265
#define DISTANCE(x0,y0,z0,x1,y1,z1) (sqrt( pow(x0-x1,2) + pow(y0-y1,2) + pow(z0-
z1,2)))
#define max(a,b)        a > b ? a : b
#define CURRENT_TIME    Scheduler::instance().clock()
//#define DEBUG
//#define ERROR
```

Figure 13: cuckoo code

8 IP detection code

The code for IP detection is available in IDSapp.cc which can be seen in aodv folder present in ns-2.34.



```

#include "IDSapp.h"
#include "math.h"
#include "object.h"
#include "mobilenode.h" //because a substance (membership function has been
called) of Class MobileNode is used. only declaration in the header file is not
enough
#include "stdlib.h" //for atoi(), string to int. int() will cause big problem.


static class IDSAppClass : public TclClass {
  public:
    IDSAppClass() : TclClass("Application/IDSApp") {}
    TclObject* create(int, const char*const*) {
      return (new IDSApp);
    }
} class_app_ids;

// Constructor
IDSApp::IDSApp() :Application(), ids_handler_(0), node_(0), ids_app_BS_(0),
file_(0),
few_training_on(0), few_training_arrival(0), few_training_off(0),
few_training_freq(0), many_training_on(0), many_training_arrival(0),
many_training_off(0), many_training_freq(0), training_general(0)
{
        int i;
        for (i=0; i<200; i++)
```

Figure 14: IP detection code