National College of Ireland

Configuration Manual

Academic Internship
MSc Cyber Security

Dinesh Kumar Jagadeesan
Student ID: x18170064

School of Computing
National College of Ireland

Supervisor: Imran Khan

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Dinesh Kumar Jagadeesan
Student ID:	x18170064
Programme:	MSc Cyber Security
Year:	2019
Module:	Academic Internship
Supervisor:	Imran Khan
Submission Due Date:	12/12/2019
Project Title:	Configuration Manual
Word Count:	580
Page Count:	6

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

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

Dinesh Kumar Jagadeesan
x18170064

1 Environmental Set-Up

Base system type	x64-based PC
Processor	Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz
Memory	16000 MB RAM
Hypervisor used	Virtual Box
Virtual OS	Ubuntu 14.04-amd64
Virtual memory	2048 MB RAM
Processors allocated	2

Table 1: System details and configurations

2 Requirements and Installation

- Download and install Ubuntu OS 14.04 [1]
- After booting the OS, Install NS2 simulator 2.34 [2]
- Install supporting tools for NS2 simulator[3] and the following dependencies
  - $ sudo apt-get install build-essential autoconf automake libxmu-dev
  - $ sudo apt-get install gcc-4.4
- Extract the contents downloaded .zip file.
- Run the below commands in terminal and extract it to home folder
  - $ tar -xvzf ns-allinone-2.34.tar.gz
- From home folder, enter the path using the command:$ cd ns-allinone-2.34/ns-2.34
- Need to change the file path in .bashrc to make sure the supporting library files are linked.
  - $ gedit /home/dinesh/.bashrc

[1] http://releases.ubuntu.com/14.04/
[2] https://sourceforge.net/projects/nsnam/files/latest/download/
[3] https://www.howtoforge.com/tutorial/ns2-network-simulator-on-ubuntu-14.04/

- Add the below file path in .bashrc file and execute the project file.
  - $ export PATH=$PATH:/home/dinesh/ns-allinone-2.34/bin:/home/dinesh/ns-allinone-2.34/tcl8.5.10/unix:/home/dinesh/ns-allinone-2.34/tk8.5.10/unix
  - $ export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/home/dinesh/ns-allinone-2.34/otcl-1.14:/home/dinesh/ns-allinone-2.34/lib
- Install NS2 and library files by using below command in this path: /home/dinesh/ns-allinone-2.34/ns-2.34
  - ./install

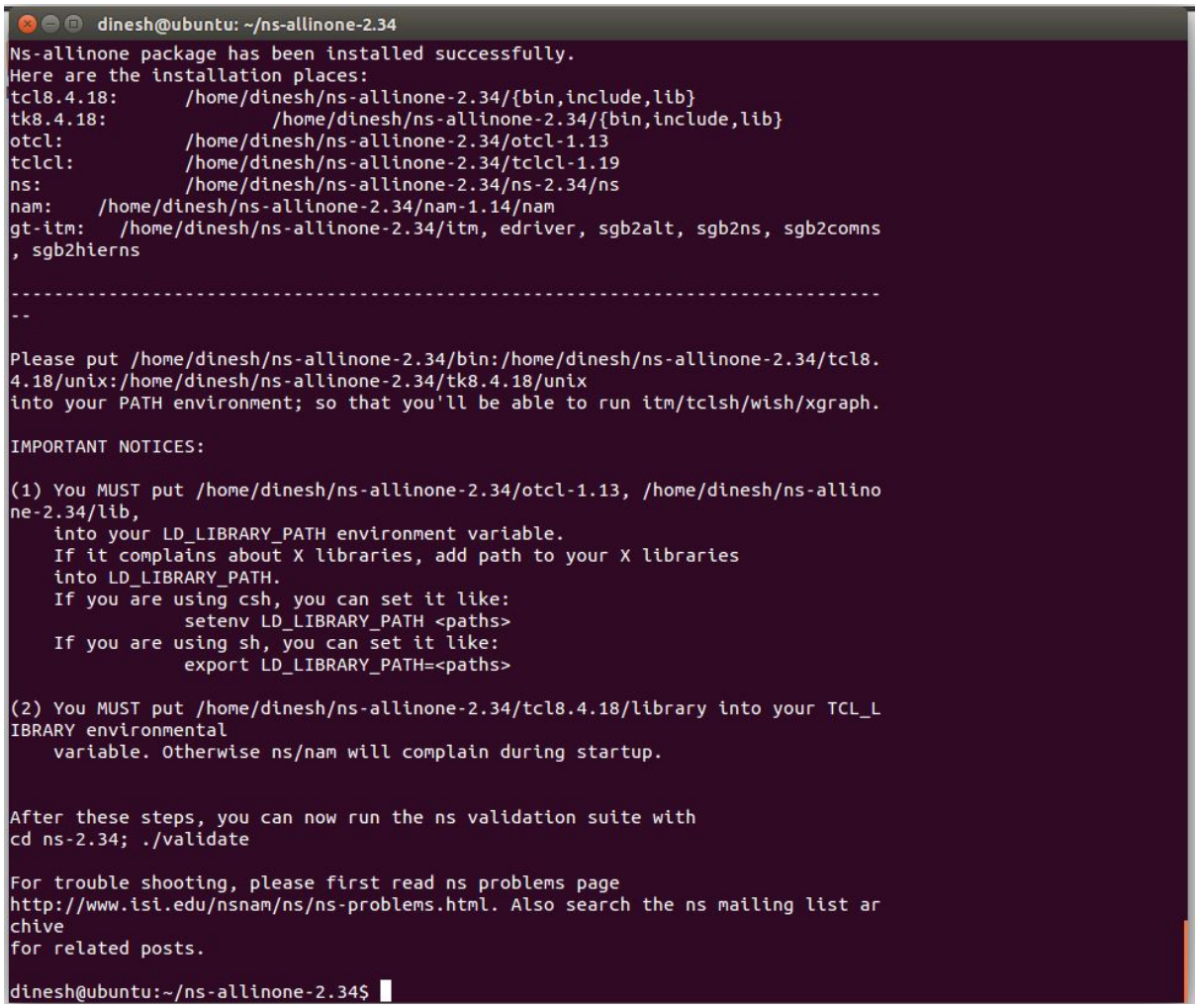

Figure 1: Screenshot of NS2 installation

3 Executing the Project

- Upon completion of installing all library files, Run the command as below in the terminal: /home/dinesh/ns-allinone-2.34
  - ./ns data.tcl

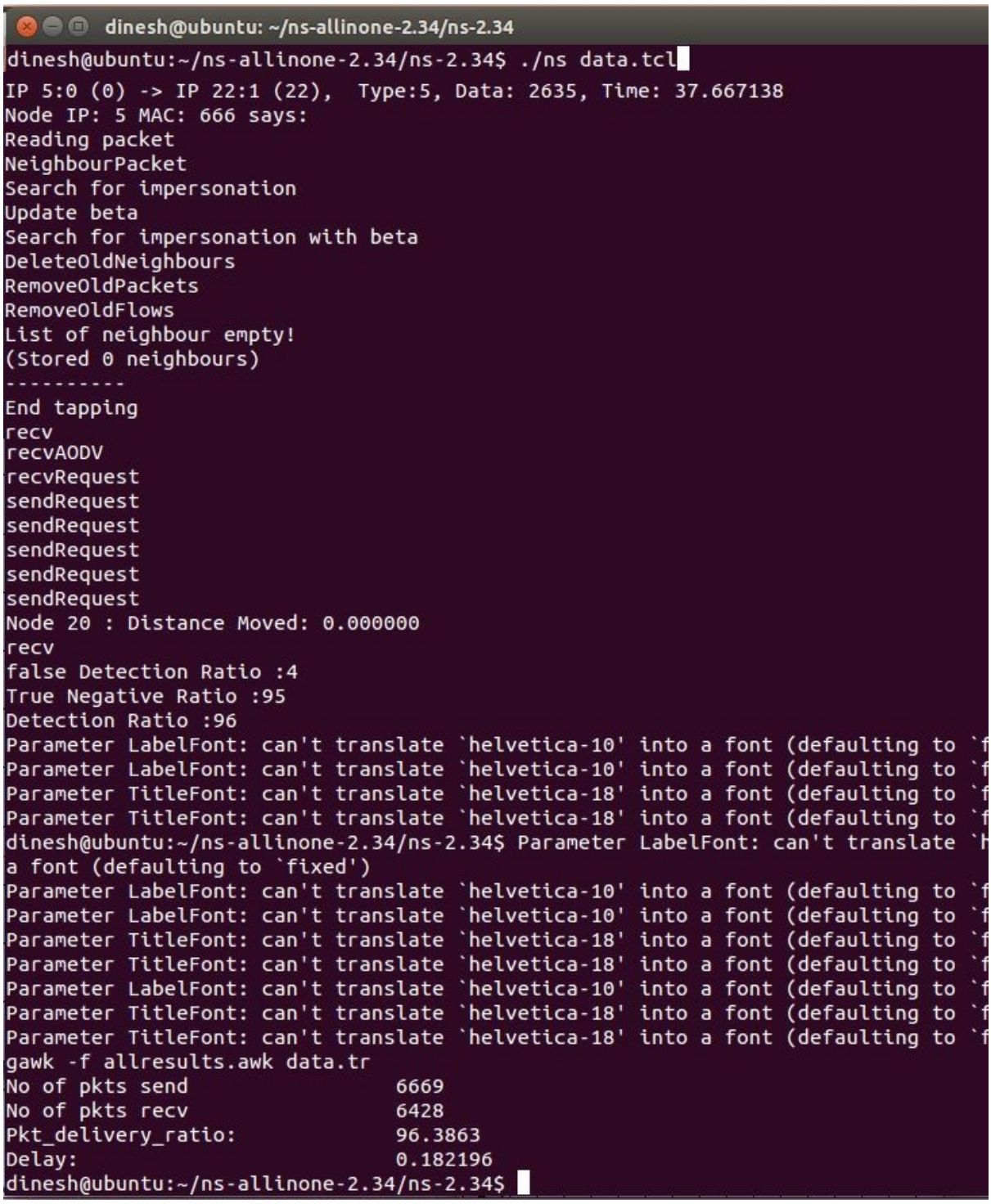

Figure 2: Screenshot of output

The above screenshot is shown with 4 attacker node and the output are shown for PDR, TDR, FDR, detection ratio an delay. By increasing the attacker node, the results are analysed in main report.

For installation visit this YouTube page: Engineering Clinic -Installation of NS2 in Ubuntu [4]

4 Appendix - Code Screenshots

4.1 Scenario Setup

The scenario file has all the parameters, link setup, node position setup, output file, trace file, graph generator.

File Path:ns-allinone-2.34/ns-2.34/data.tcl

data.tcl

```
Agent/AODV set sufix_ 1
set impersonation 4
Agent/AODV set numbermali_  $impersonation
# Creating simulator
set ns            [new Simulator]       ;#ns simulator
$ns use-newtrace

# creation of trace file
set tracefd       [open data.tr w]      ;#trace file
set namtrace       [open data.nam w]    ;#nam trace animation file

# Creating network file animations

$ns trace-all $tracefd
$ns namtrace-all-wireless $namtrace $val(x) $val(y) ;# To show the value of x and y distance in nam file

# Creating topography
set topo       [new Topography]     ;#topography object
$topo load_flatgrid $val(x) $val(y)

# Creation of GOD- General operation director
create-god $val(nn)
#setting Channel
set chan0 [new $val(chan)]
#configuring node
$ns node-config -adhocRouting $val(rp) \
                -llType $val(ll) \
                -macType $val(mac) \
                -ifqType $val(ifq) \
                -ifqLen $val(ifqlen) \
                -antType $val(ant) \
                -propType $val(prop) \
                -phyType $val(netif) \
                -channel $chan0 \
                -topoInstance $topo \
                -agentTrace ON \
                -routerTrace ON \
                -macTrace ON \
                -movementTrace OFF

# node position setup and initializing coordinates

set node_(0) [$ns node]
$node_(0) set X_ 1430
$node_(0) set Y_ 2200
```

The code in the screenshots is about increasing impersonation node in 2nd line "set impersonation". Creating a trace file to record the packet transfer in document named as data.tr. Namtrace is to produces the output as an animation file.

[4] https://www.youtube.com/watch?v=FXm8i1K-6jI&t=586s&pbjreload=10/

4.2 Impersonation attack code

In this file, there are few in-built conditions as link-layer detection, route handling functions. We added an impersonation attack setup with conditions and watermark encoder. The screenshot attached was the main function of the impersonation attack.

File Path: ns-allinone-2.34/ns-2.34/aodv/aodv.cc

```
/*
 * If the route is up, forward the packet
 */

//if (rt->rt_flags == RTF_UP){
if ((rt->rt_flags == RTF_UP) &&
        //Start T2ar Code
        (impersonation != 1)) {
        //End T2ar Code
        assert(rt->rt_hops != INFINITY2);
        forward(rt, p, NO_DELAY);
}
//Start T2ar Code
else if(impersonation == 1){
        if(impersonationd == 0 && packets_dropped >= PACKETS_TO_CONSIDER_AN_ATTACK){
                impersonationd=1;
                printAttackMessage(p);
        }
        packets_dropped++;
        drop(p, DROP_IFQ_FILTER);
        return;
}
//End T2ar Code
```

The code in about creating a scenario for impersonation attack with the condition if impersonation == 0 and packets loss happens, then it passes the condition to impersonation=1 where it has a pre-defined message to drop the packets and it is shown in scenario file with a red dotted number as attacker node.

4.3 Kalman Filter Implementation

In this file, Kalman filter has functions like detecting malicious node, transferring packets between node, new packet detection, updating node position in tracefile.

File Path: ns-allinone-2.34/ns-2.34/aodv/t2ar/t2ar.cc

```
float kalman2_filter(kalman2_state *state, float z_measure)
{
    float temp0 = 0.0f;
    float temp1 = 0.0f;
    float temp = 0.0f;

    /* Step1: Predict */
    state->x[0] = state->A[0][0] * state->x[0] + state->A[0][1] * state->x[1];
    state->x[1] = state->A[1][0] * state->x[0] + state->A[1][1] * state->x[1];
    /* p(n|n-1)=A^2*p(n-1|n-1)+q */
    state->p[0][0] = state->A[0][0] * state->p[0][0] + state->A[0][1] * state->p[1][0] + state->q[0];
    state->p[0][1] = state->A[0][0] * state->p[0][1] + state->A[1][1] * state->p[1][1];
    state->p[1][0] = state->A[1][0] * state->p[0][0] + state->A[0][1] * state->p[1][0];
    state->p[1][1] = state->A[1][0] * state->p[0][1] + state->A[1][1] * state->p[1][1] + state->q[1];

    /* Step2: Measurement */
    /* gain = p * H^T * [r + H * p * H^T]^(-1), H^T means transpose. */
    temp0 = state->p[0][0] * state->H[0] + state->p[0][1] * state->H[1];
    temp1 = state->p[1][0] * state->H[0] + state->p[1][1] * state->H[1];
    temp  = state->r + state->H[0] * temp0 + state->H[1] * temp1;
    state->gain[0] = temp0 / temp;
    state->gain[1] = temp1 / temp;
    /* x(n|n) = x(n|n-1) + gain(n) * [z_measure - H(n)*x(n|n-1)]*/
    temp = state->H[0] * state->x[0] + state->H[1] * state->x[1];
    state->x[0] = state->x[0] + state->gain[0] * (z_measure - temp);
    state->x[1] = state->x[1] + state->gain[1] * (z_measure - temp);

    /* Update @p: p(n|n) = [I - gain * H] * p(n|n-1) */
    state->p[0][0] = (1 - state->gain[0] * state->H[0]) * state->p[0][0];
    state->p[0][1] = (1 - state->gain[0] * state->H[1]) * state->p[0][1];
    state->p[1][0] = (1 - state->gain[1] * state->H[0]) * state->p[1][0];
    state->p[1][1] = (1 - state->gain[1] * state->H[1]) * state->p[1][1];

    return state->x[0];
}

bool
T2AR::file_exists(const char *filename) {
        if (FILE *file = fopen(filename, "r")) { //I'm sure, you meant for READING =)
                fclose(file);
                return true;
        }
        return false;
}
```

Here the Kalman filter works on matrix method with the help of formula defined in main report. The values are measured, updated with gain and predict the measurement.

The screenshots attached are not the entire code in the file, these are important functions in the program file to execute.